



НЕТОЛОГИЯ  
групп

# ЗАНЯТИЕ 1.4

# КЛАСТЕРИЗАЦИЯ



# Алексей Кузьмин

Директор разработки; Data Scientist

**ДомКлик.ру**



aleksej.kuzmin@gmail.com

---

# ЦЕЛИ ЗАНЯТИЯ

---

## В КОНЦЕ ЗАНЯТИЯ ВЫ НАУЧИТЕСЬ:

- производить **кластеризацию данных**
- выбирать **наиболее подходящий алгоритм** для задачи

---

О ЧЁМ ПОГОВОРИМ И ЧТО  
СДЕЛАЕМ

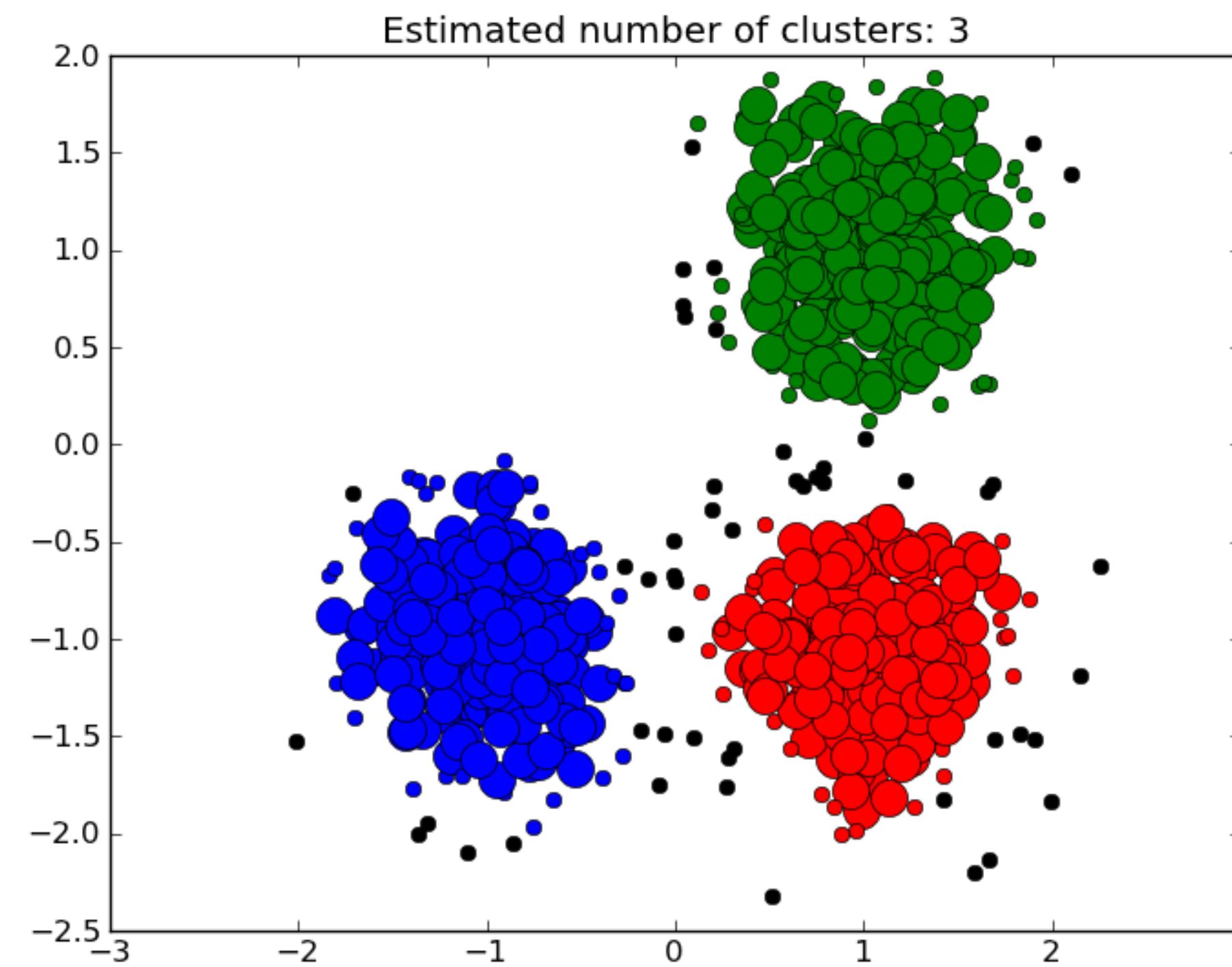
- 
1. Задача кластеризации: постановка и примеры
  2. Основные алгоритмы
  3. Метрики качества кластеризации

---

# 1. ЗАДАЧА КЛАСТЕРИЗАЦИИ

# ТИПЫ ЗАДАЧ

- \* классификация
- \* ранжирование
- \* регрессия
- \* **кластеризация**



## ПРИМЕРЫ ЗАДАЧ КЛАСТЕРИЗАЦИИ

**Пользовательская сегментация.** Как выглядят типичные пользователи? (находим сектора, работаем с ними отдельно)

**Логистика.** Где расположить магазины, чтобы охватить большее кол-во покупателей?

**Новости.** О чём сейчас пишут СМИ? (Я.Новости кластеризуют новости и выдают их отдельными темами)

**EDA.** Есть 100млн обращений пользователей. О чём они пишут?

## ДОПОЛНИТЕЛЬНЫЕ ПРИЛОЖЕНИЯ

**Создание дополнительных фич.** Можно дополнить имеющиеся данные метками принадлежности к одному из классов

**Разметка данных.** Если нет проставленных классов, но нужно сделать классификатор, то в создании разметки для обучающей выборки сильно поможет кластеризация

**Поиск структуры данных** как часть эксплоративного анализа

# ПОСТАНОВКА ЗАДАЧИ

$$X^{n \times k} \Rightarrow y^{n \times 1}$$

$$\rho : X \times X \rightarrow [0, \infty)$$

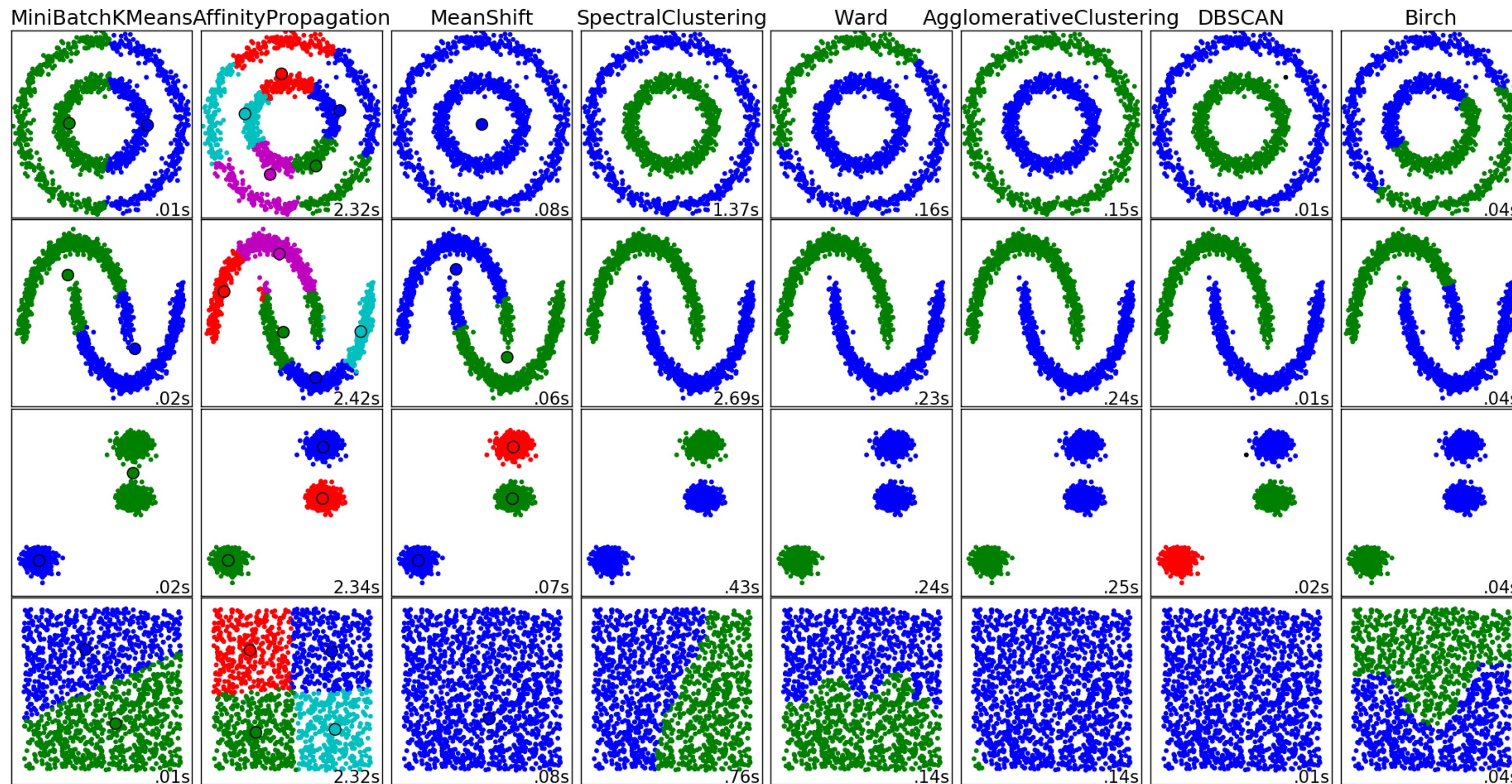
Каждому объекту поставить в пару метку кластера так, чтобы **близкие** объекты лежали в одном кластере, а далёкие - в разных

Это математически некорректная задача, в ней есть неоднозначности и **нет** **правильного ответа**

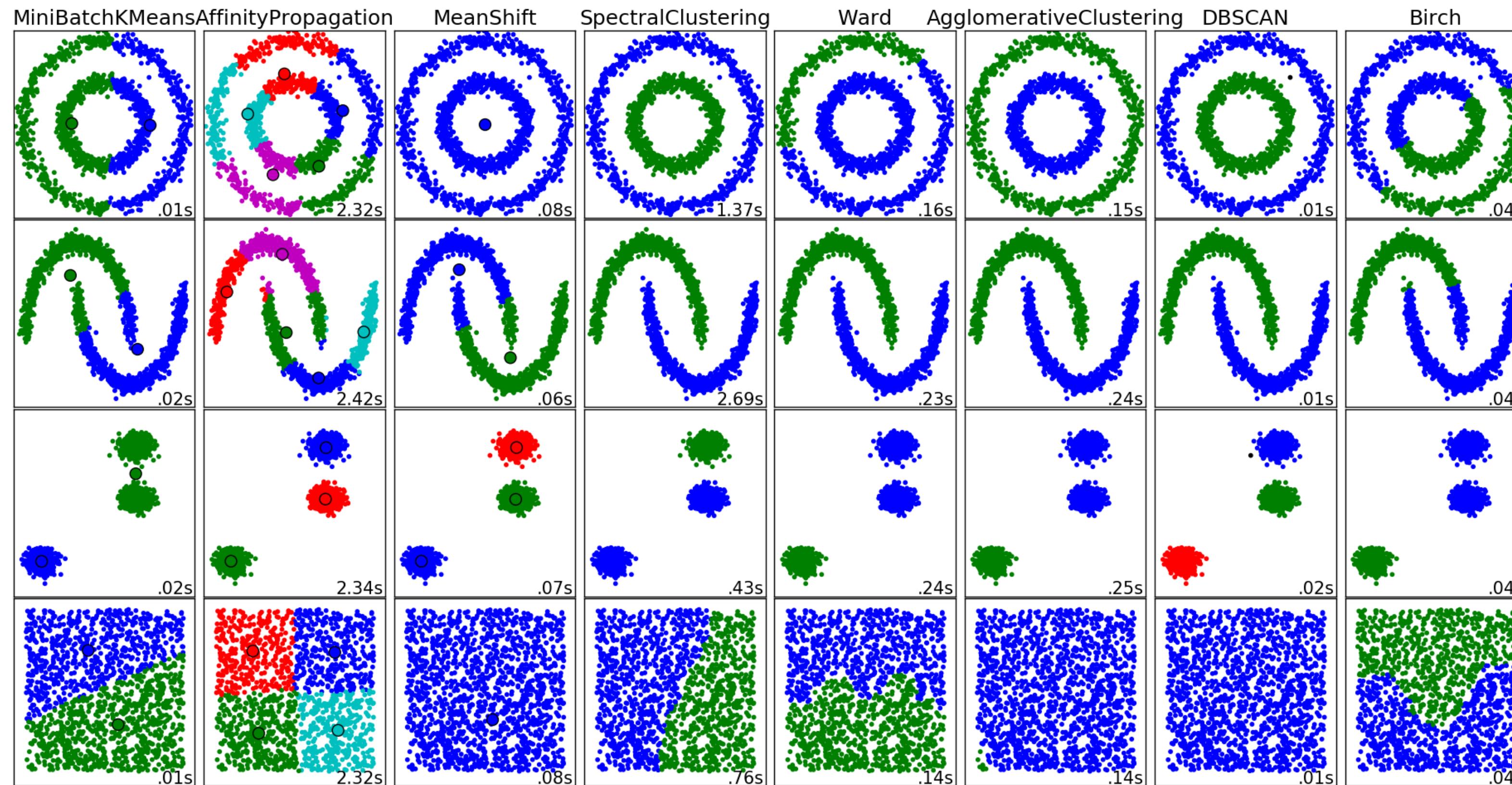
---

## 2. АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ

# АЛГОРИТМОВ - МНОГО. ЗАЧЕМ?



# АЛГОРИТМОВ - МНОГО. ЗАЧЕМ?



Как и в других  
задачах:  
разные алгоритмы  
справляются лучше с  
разными формами  
зависимостей

# ТИПЫ КЛАСТЕРИЗАЦИИ

**Жёсткая** кластеризация

(1 объект - 1 класс)

**Мягкая** (fuzzy) кластеризация

(1 объект - несколько (или 0) классов)

**Иерархическая** кластеризация

(объект внутри кластера 2.1 -> внутри кластера 2)

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. PREPROCESSING

---

# PREPROCESSING

# PREPROCESSING

Все методы кластеризации основываются на метриках и потому крайне чувствительны к одному масштабу данных, поэтому

**StandardScaler** - must have

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. K-MEANS

---

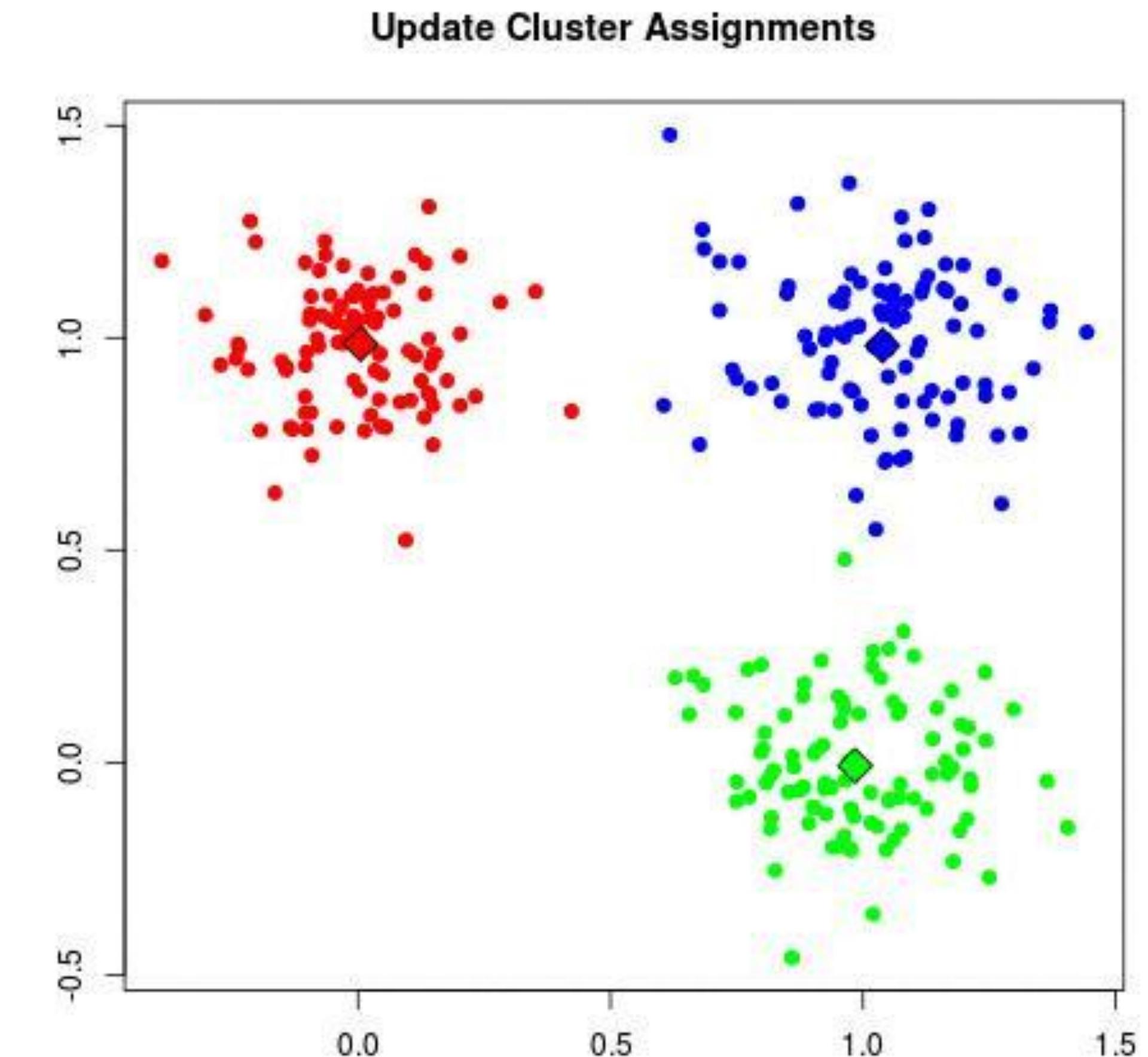
# K-MEANS

# АЛГОРИТМ

Задать начальные значения центроидов кластеров

**Повторять, пока** центроиды смещаются:

- \* присвоить наблюдениям номер кластера с **ближайшим** к ним центром
- \* передвинуть центроиды кластеров к среднему значению координат членов кластера



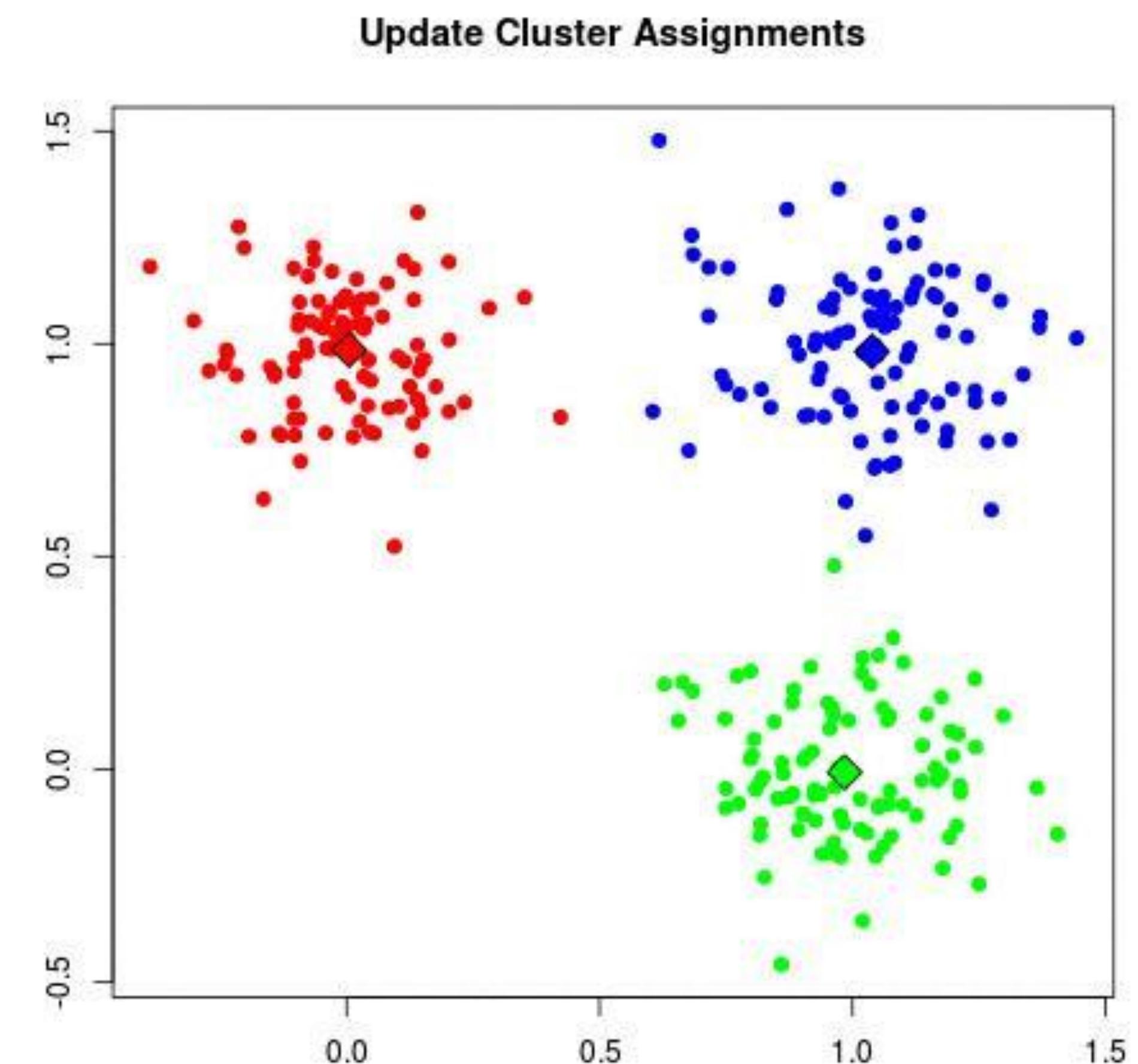
# ЦЕЛЬ

Минимизировать внутриклассовые  
отличия от центроида:

$$\sum_{i=0}^n \min_{\mu_j} (\|x_i - \mu_j\|)^2$$

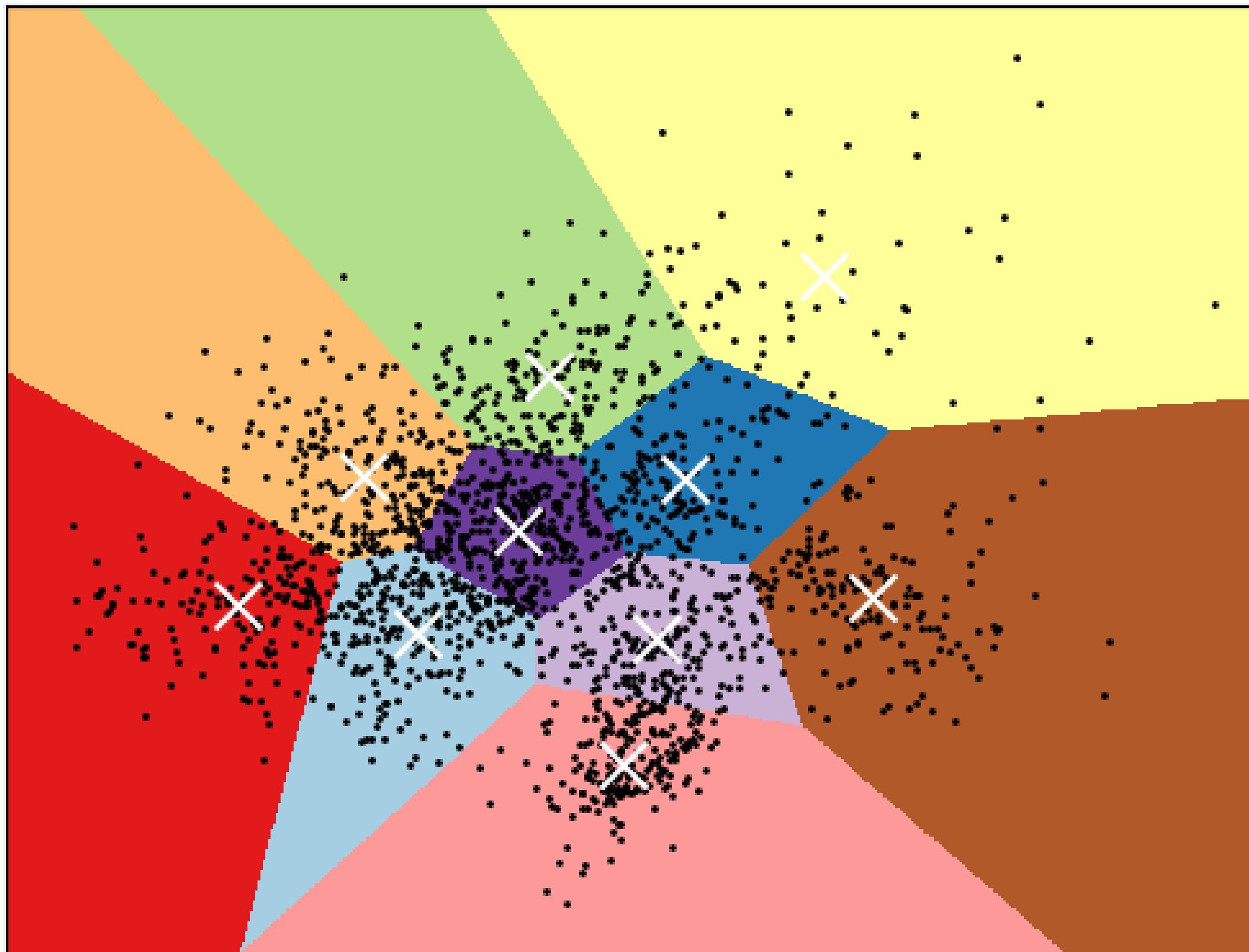
Связанные с этим проблемы:

- \* предположение о выпуклости и однородности кластеров
- \* проклятие размерности



# ИТОГ

K-means clustering on the digits dataset (PCA-reduced data)  
Centroids are marked with white cross

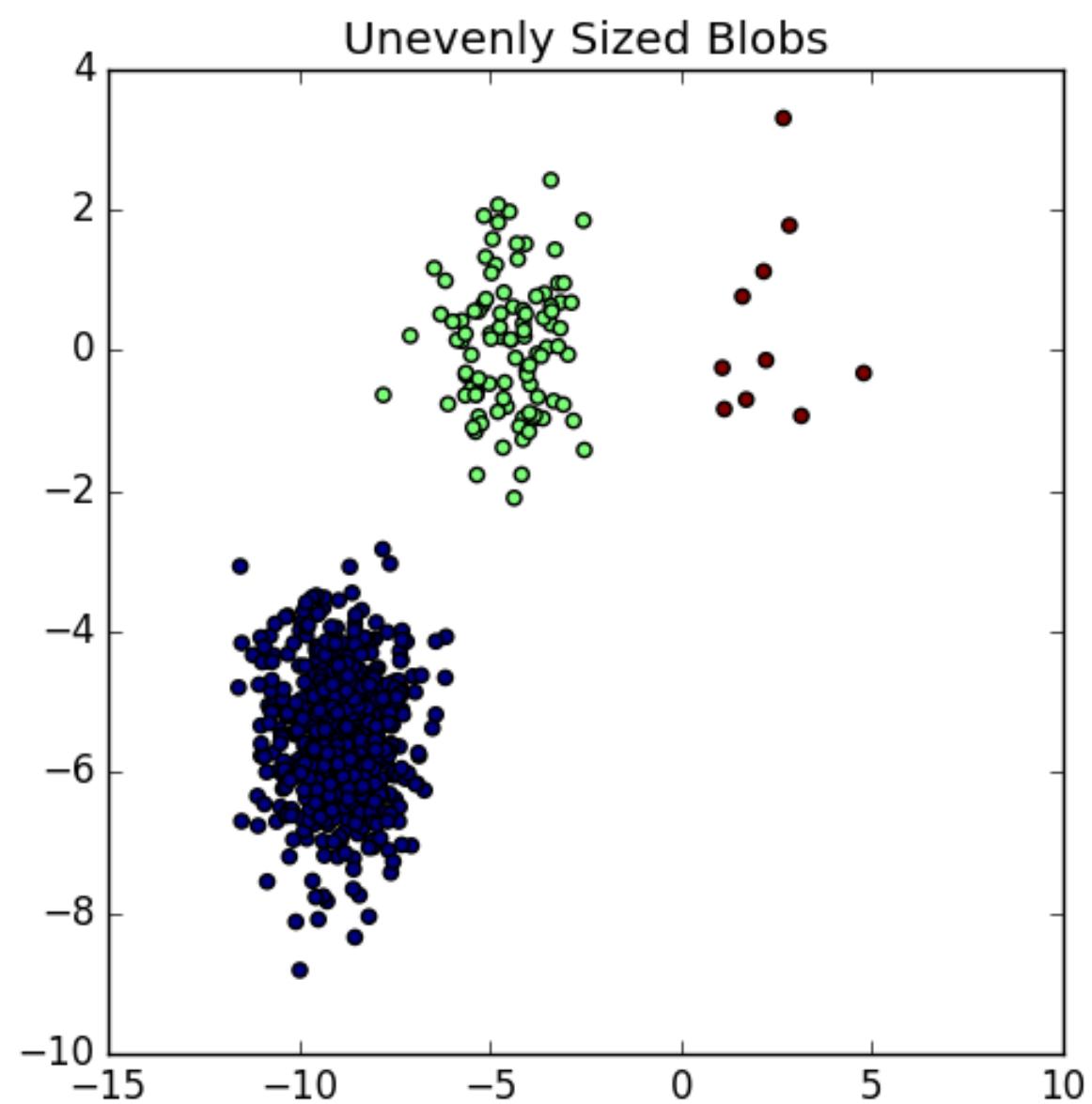
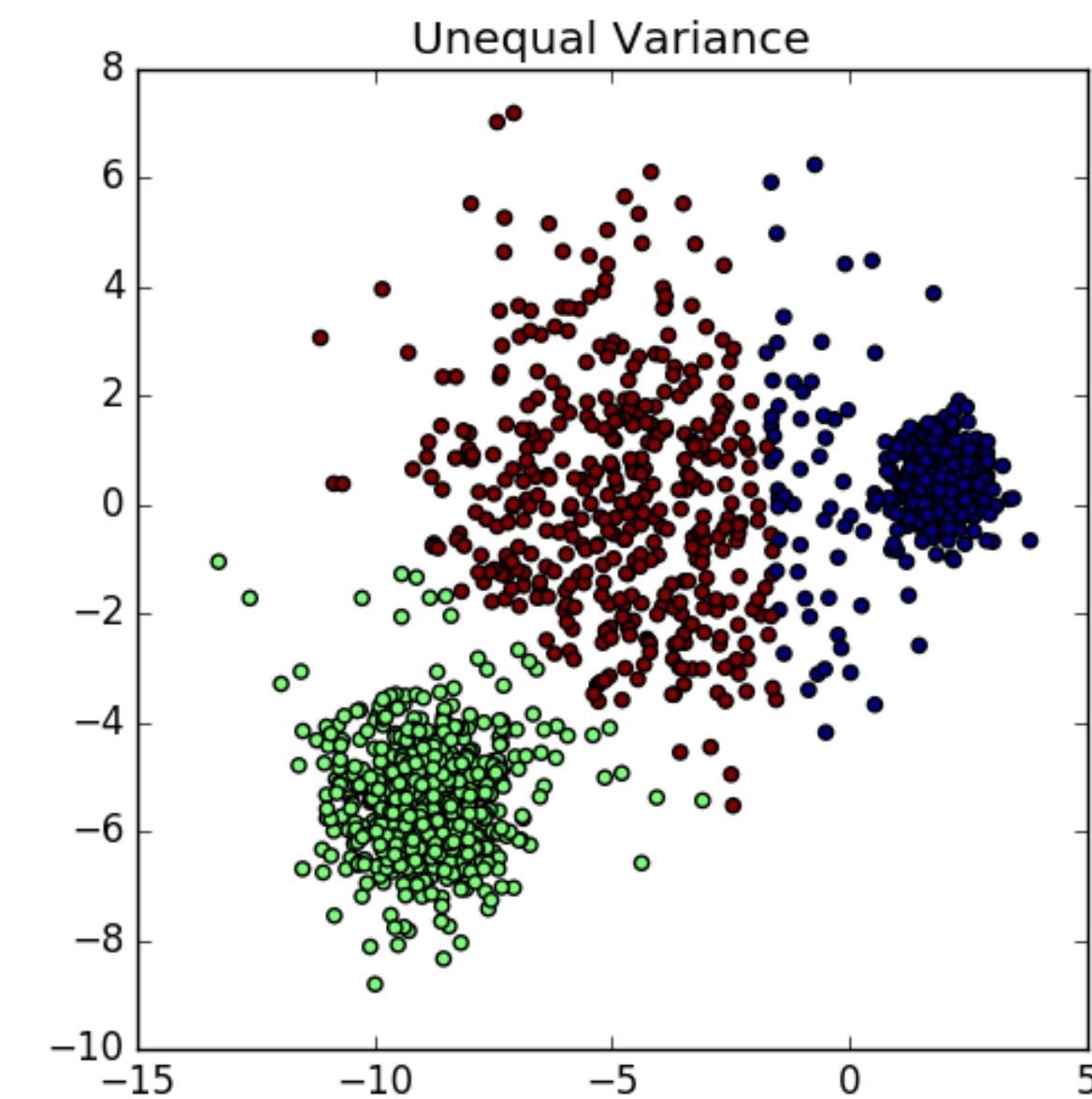
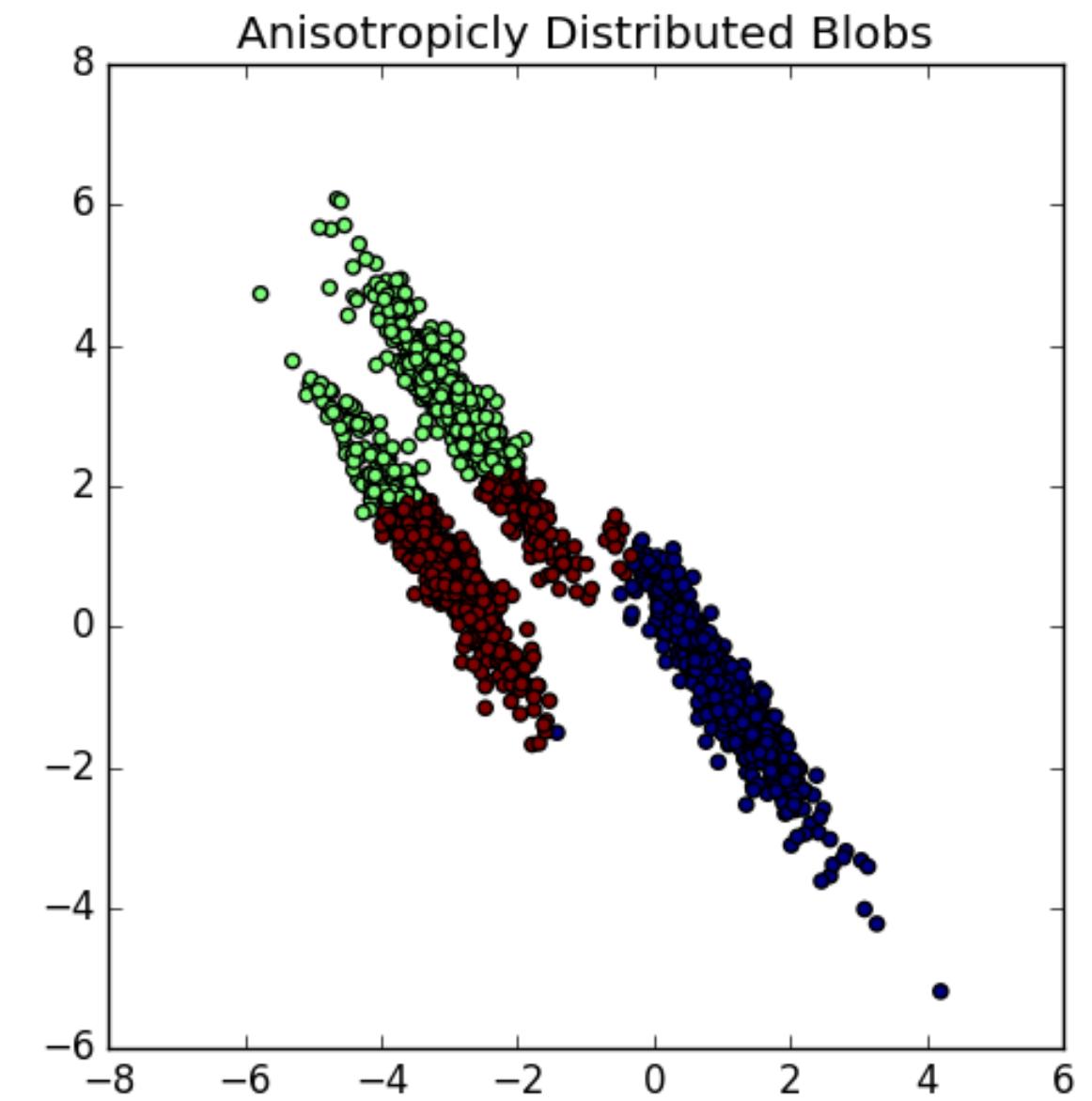
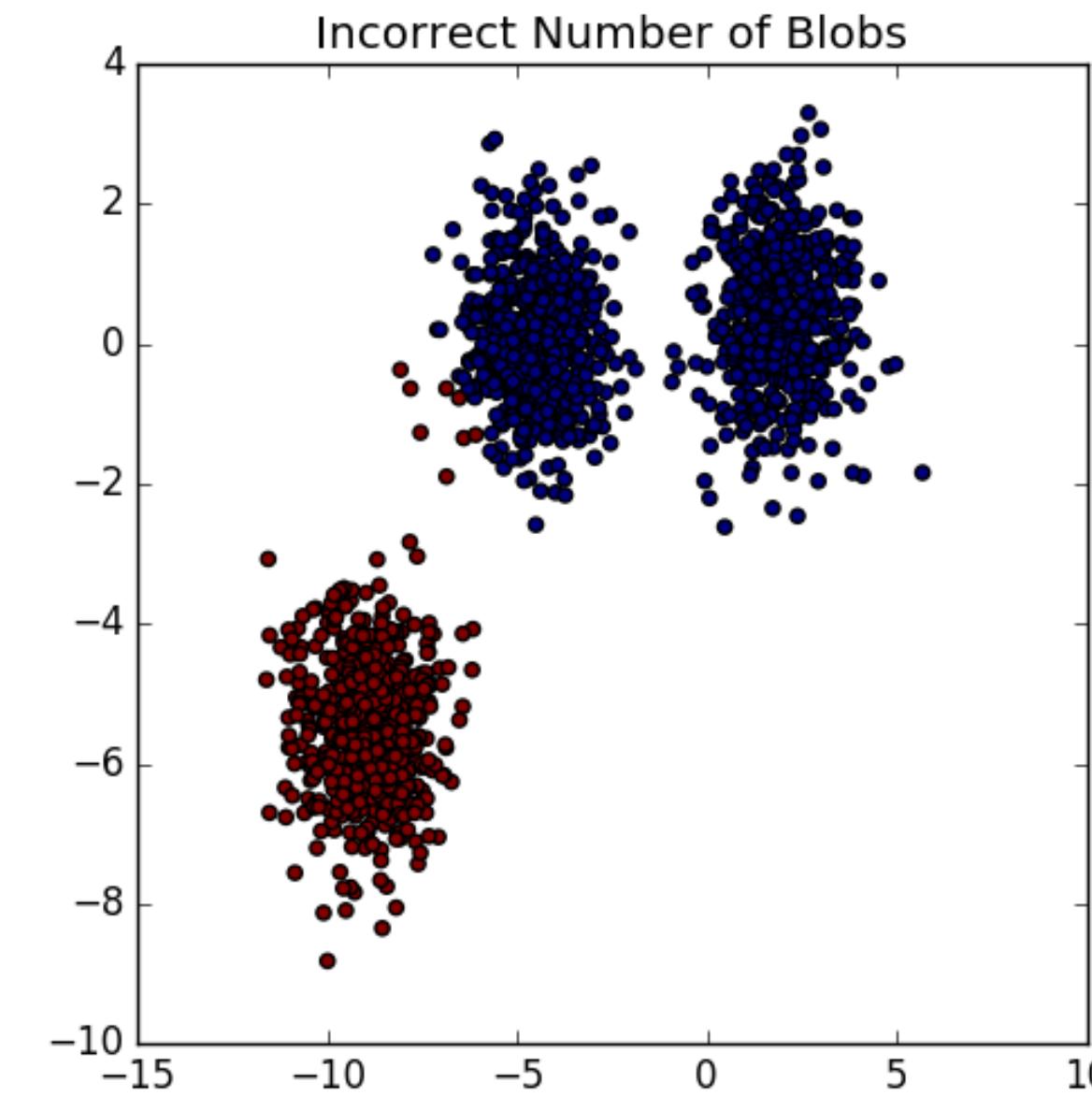


Пространство  
нарезается на лоскуты  
из прямых  
гиперплоскостей

# ОГРАНИЧЕНИЯ

Алгоритм может выдавать  
контринтуитивные результаты:

1. Если указано не то число кластеров
2. Кластеры - не выпуклые и близко расположены
3. Разная дисперсия близких кластеров

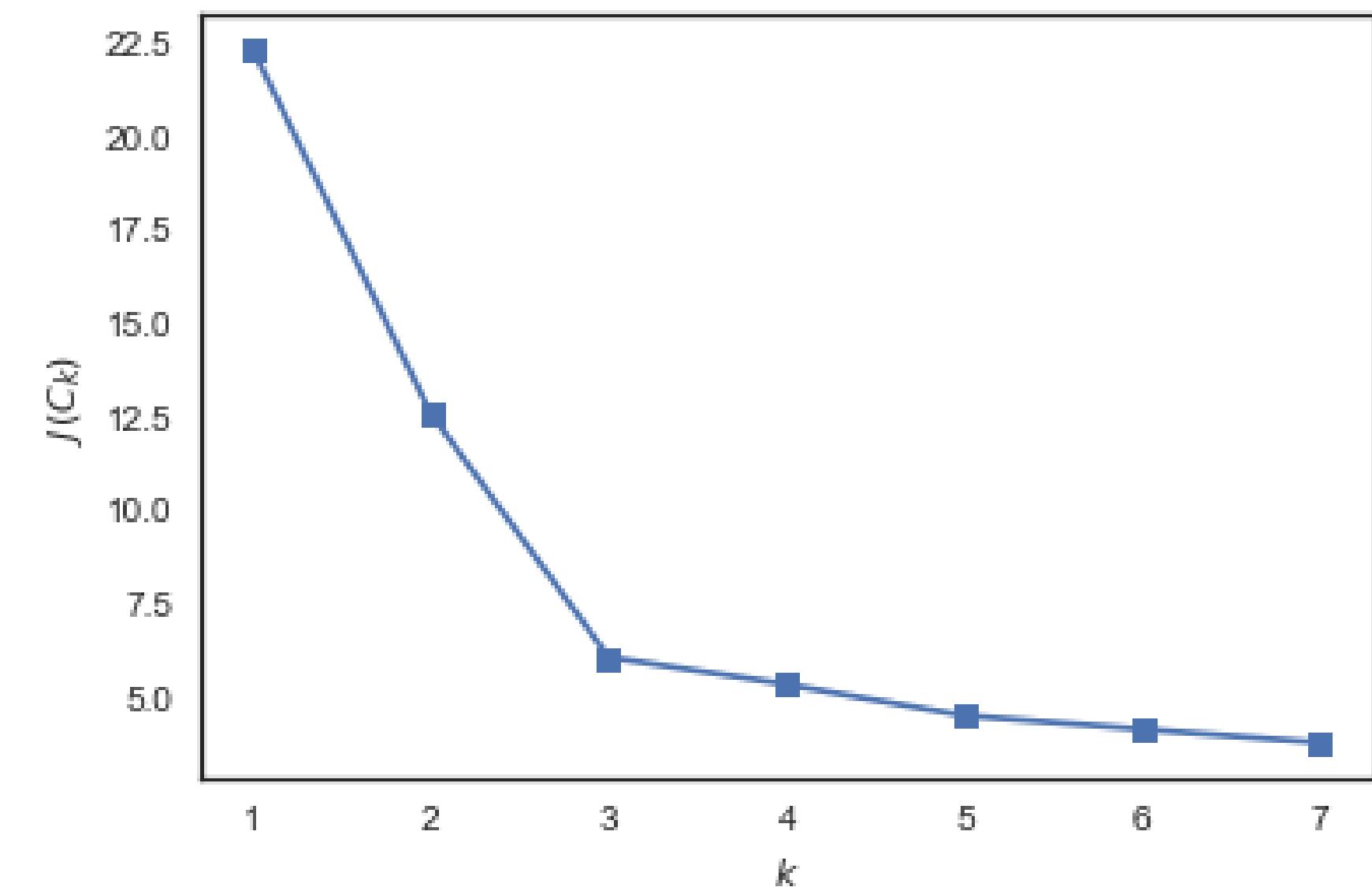


# КОЛИЧЕСТВО КЛАСТЕРОВ

**Идея:** перебирать от 1 до N кластеров, засечь, с какого момента качество перестанет быстро улучшаться

$$(\text{т.е. } k^* = \operatorname{argmin}(\Delta J(C_{k+1}) / \Delta J(C_k)))$$

**Качество** - сумма квадратов расстояний от точек до центроидов кластеров



# НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ

Алгоритм очень зависит от начального приближения: метод сойдётся всегда, но к разным локальным минимумам.

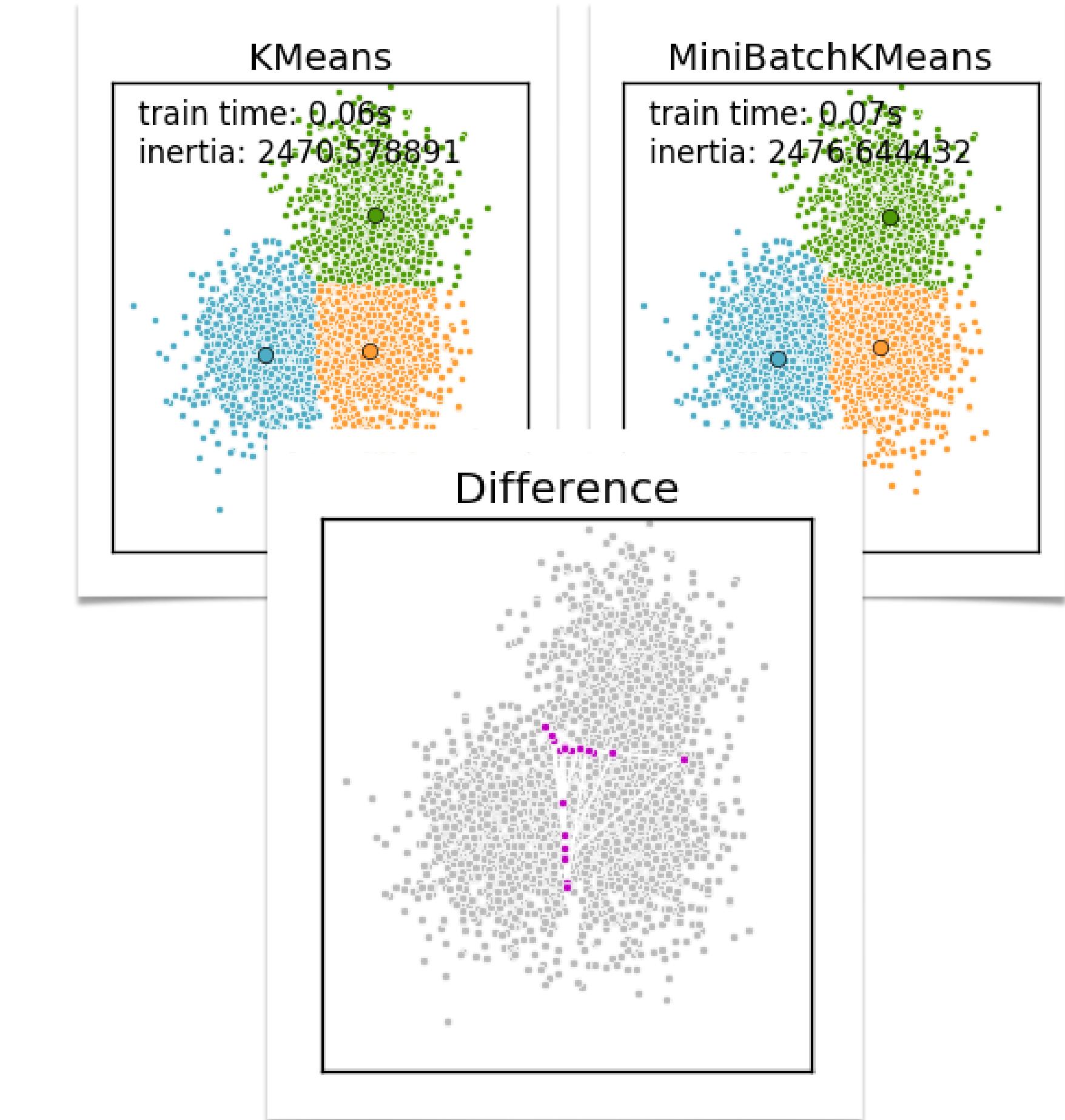
Какие точки выбрать?

- \* **Мультистарт:** N наборов начальных приближений, выбор лучшего
- \* **Наиболее удалённые** друг от друга точки:
  - \* удалить аномалии (посчитать среднее расстояние до q ближайших соседей, отбросить  $\alpha\%$  самых удалённых)
  - \* взять 2 самые дальние друг от друга точки, они составят множество U
  - \* k-2 раз добавлять в U по 1 точке, расстояние которой до ближайшей из старых точек U будет максимально большим

## УСКОРЕНИЕ. MINI BATCH KMEANS

**Способ:** каждый шаг брать не все точки, а лишь подмножества (batch), обновляя центроиды как среднее признаков объектов кластера как текущего, так и всех предыдущих шагов

**Результат:** рост скорости с мизерным падением качества



# РАЗВИТИЕ. МЯГКИЙ ВАРИАНТ (EM)

Более мягкий вариант KMeans: каждому объекту ставить в соответствие не 1 кластер, а вектор близости к каждому кластеру

**Повторять, пока** центроиды смещаются:

- \* оценить близость каждого объекта к каждому центроиду кластеров
- \* присвоить объектам номер кластера с ближайшим к ним центром
- \* передвинуть центроиды к **средневзвешенному** значению координат всех объектов, взвешивая по близости объекта к текущему центроиду кластера

# РЕАЛИЗАЦИЯ В SKLEARN

## sklearn.cluster.KMeans

- \* n\_clusters=8
- \* init='k-means++'
- \* n\_init=10
- \* max\_iter=300
- \* tol=0.0001
- \* precompute\_distances='auto'
- \* verbose=0
- \* random\_state=None
- \* copy\_x=True
- \* n\_jobs=1
- \* algorithm='auto'

### **Основные параметры**

- \* n\_clusters - количество кластеров для разбиения
- \* init: 'k-means+', 'random', ndarray - начальное приближение
- \* max\_iter - кол-во итераций
- \* n\_jobs - кол-во процессоров (-1 - max)

### **Основные характеристики**

- \* 11 параметров
- \* по умолчанию: 10 начальных умных запусков на 1 процессоре, кластеризация на 8 групп

### **Основные методы**

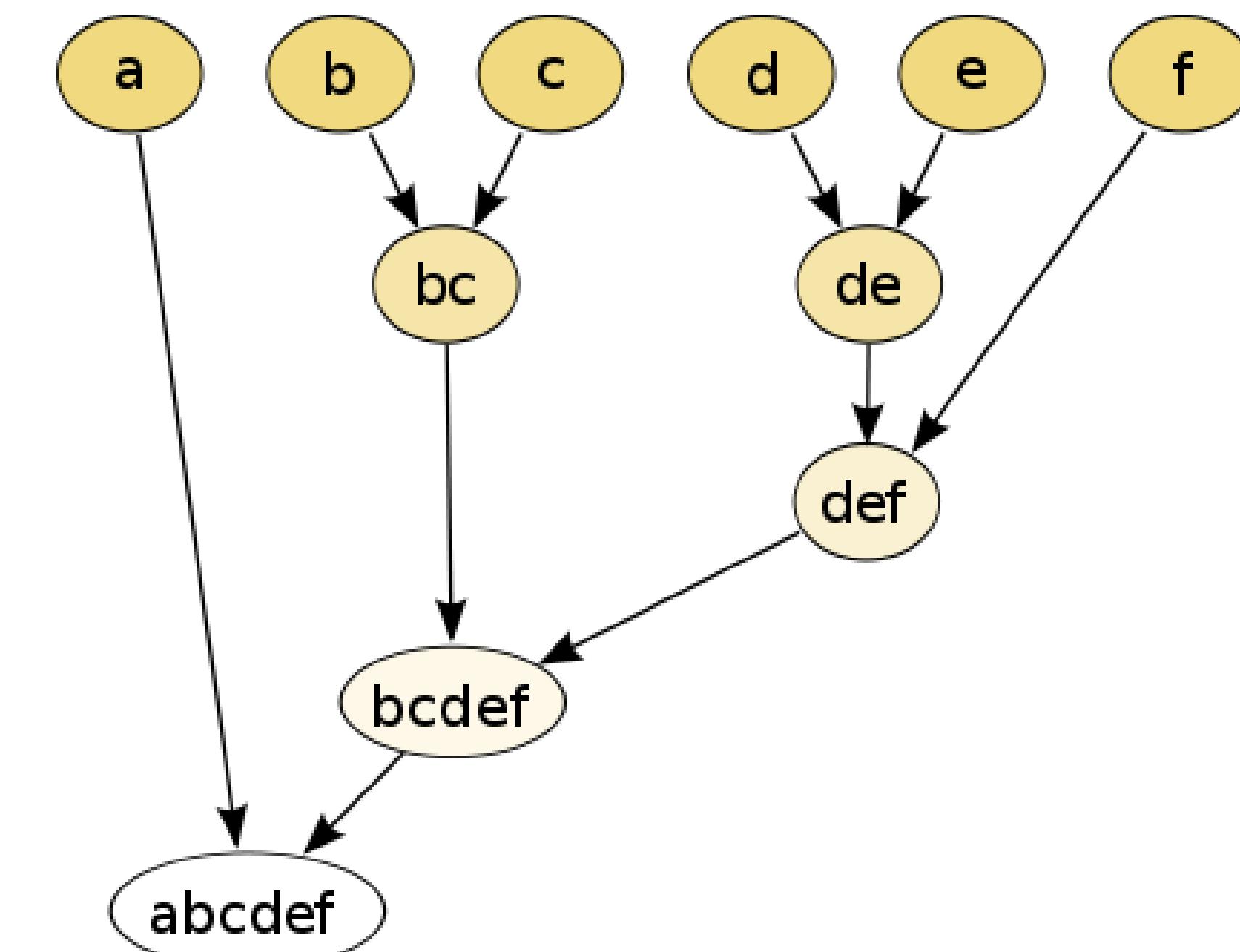
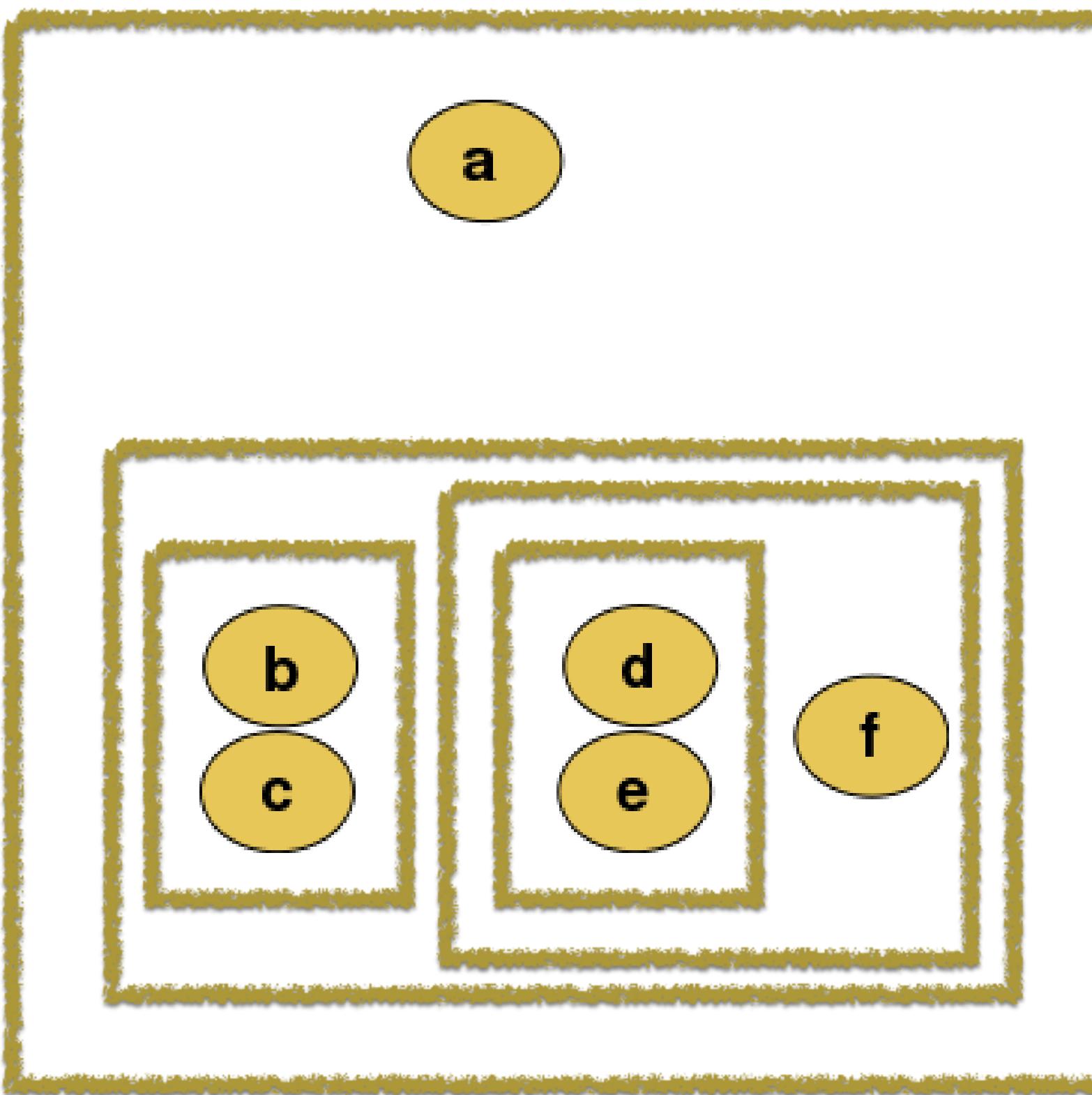
- \* fit, fit\_predict, fit\_transform, transform, predict

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. HIERARCHICAL CLUSTERING

---

# HIERARCHICAL CLUSTERING

# ИДЕЯ

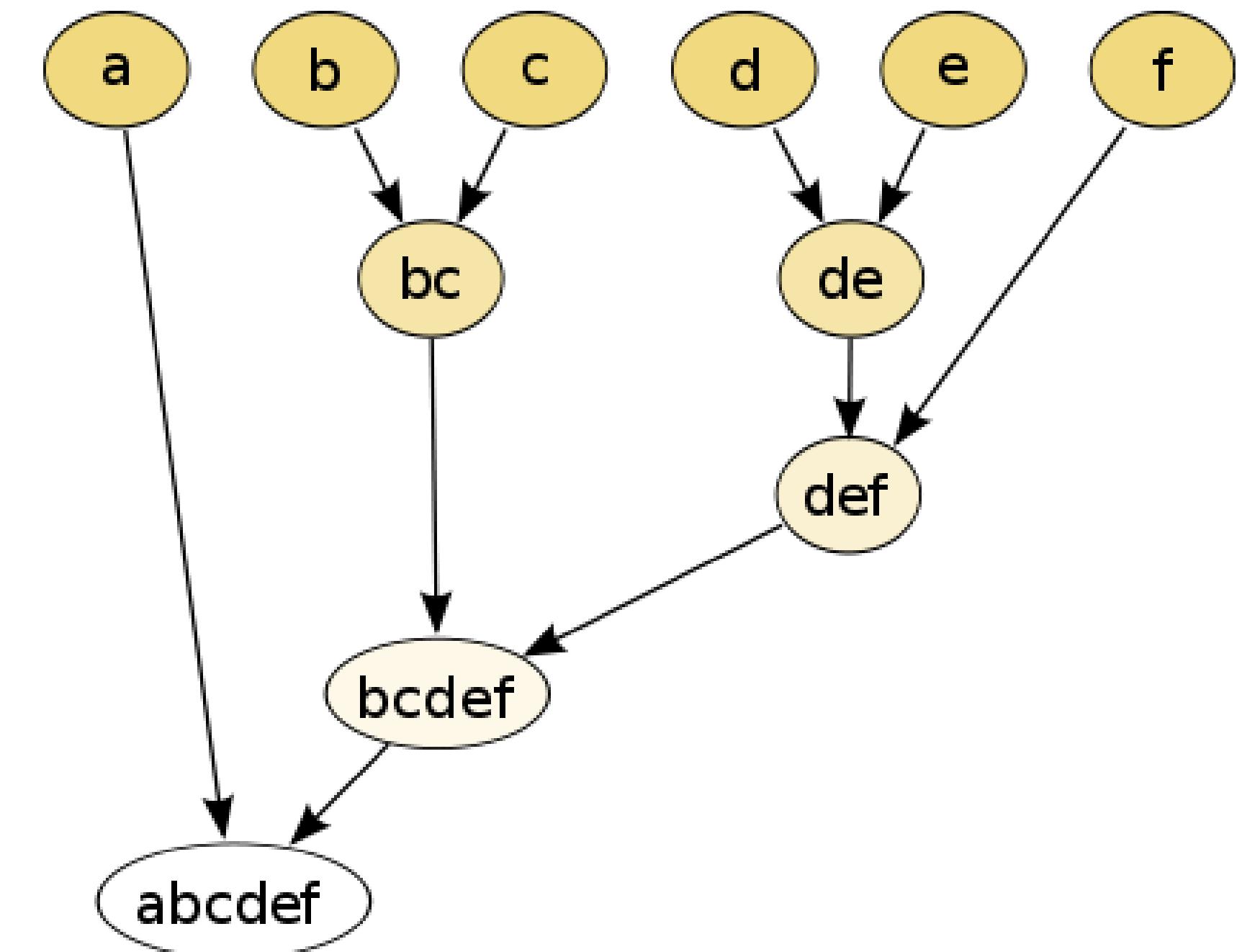


# АЛГОРИТМ

Все объекты - отдельные кластеры

**Повторять, пока** > 1 кластера:

\* соединить 2 **ближайших** кластера



## АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. HIERARCHICAL CLUSTERING

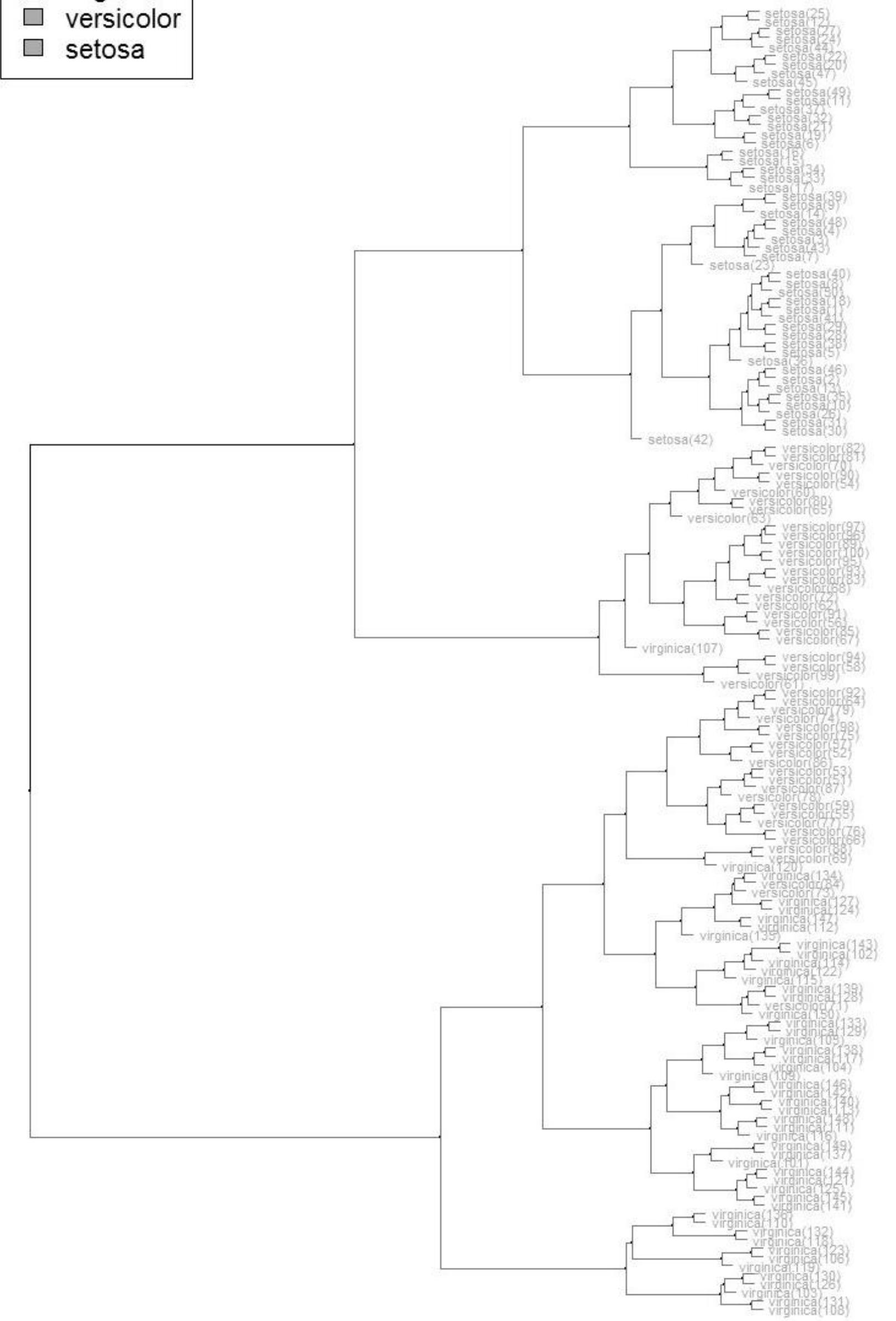
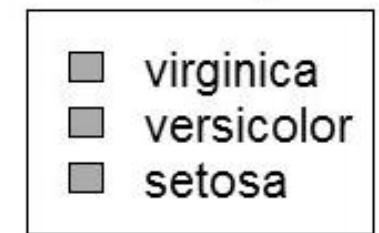
### ПРИМЕР

Дендрограмма кластеризации цветков ириса.

Проведена иерархическая кластеризация,  
визуально отображаемая в виде  
дендрограммы.

На картинке цветом линии отмечены 3  
кластера, а цветом надписи - настоящий вид  
цветка

Clustered Iris data set  
(the labels give the true flower species)



## АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. HIERARCHICAL CLUSTERING

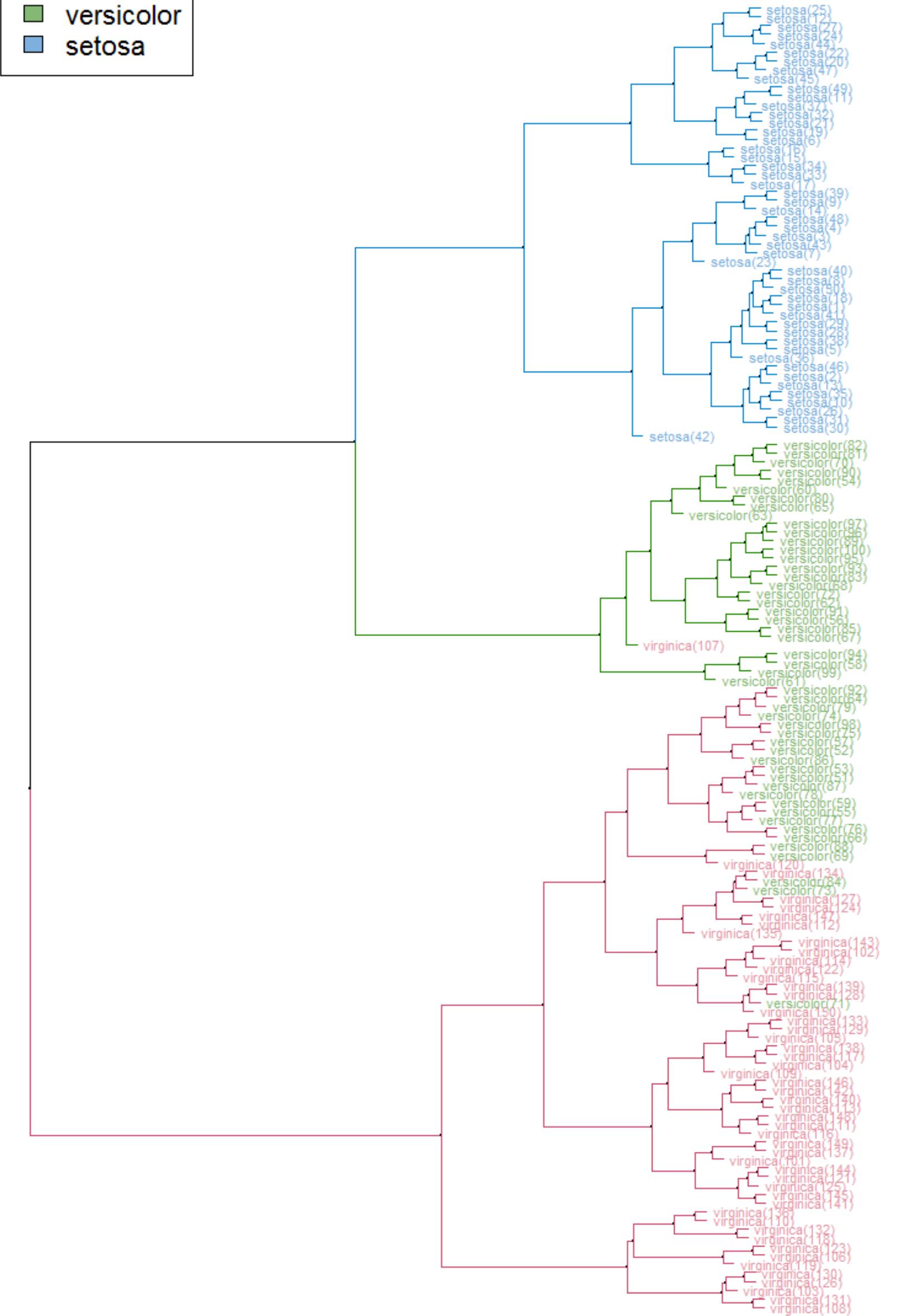
### ПРИМЕР

Дендрограмма кластеризации цветков ириса.

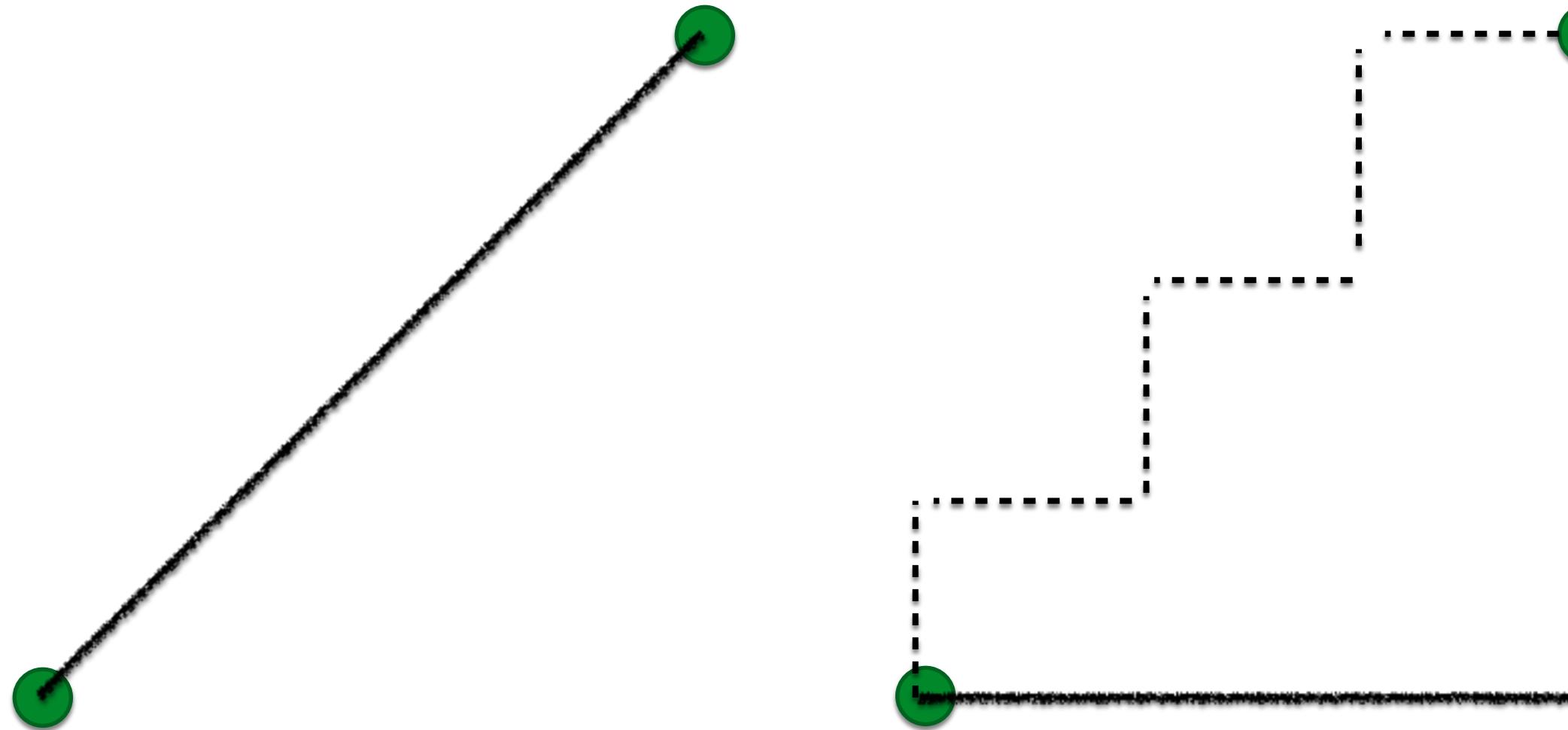
Проведена иерархическая кластеризация, визуально отображаемая в виде дендрограммы.

На картинке цветом линии отмечены 3 кластера, а цветом надписи - настоящий вид цветка

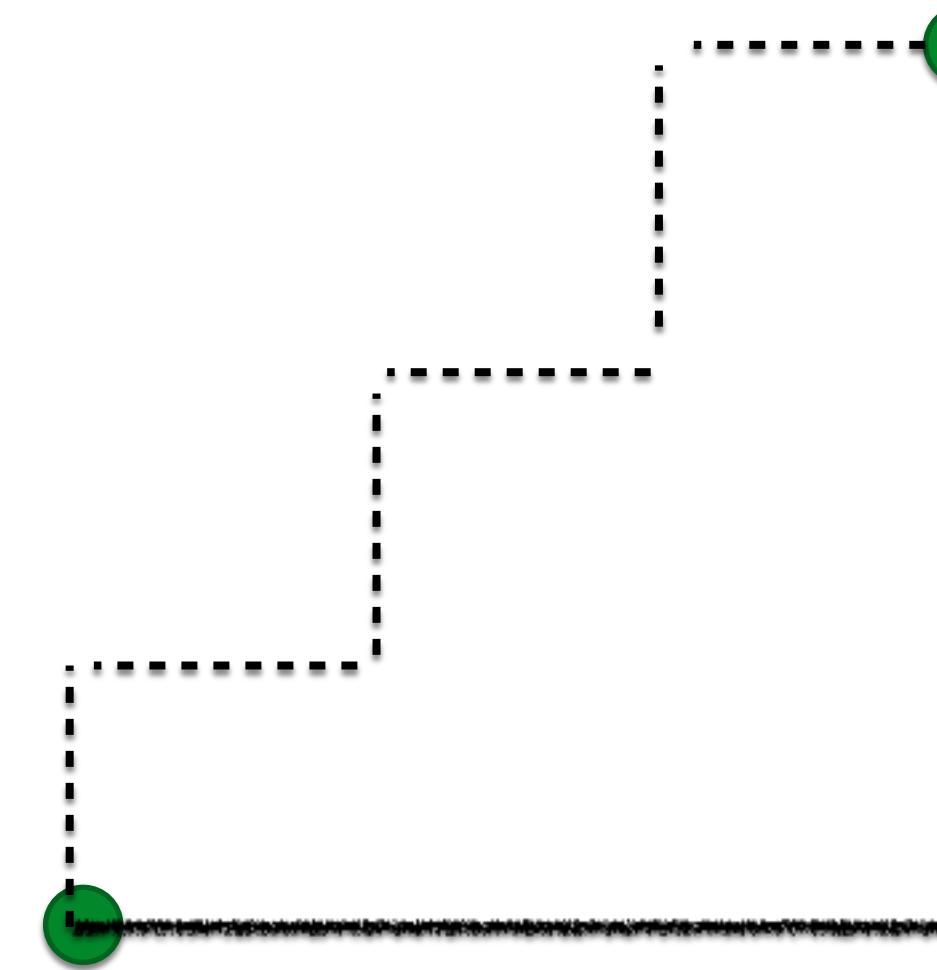
Clustered Iris data set  
(the labels give the true flower species)



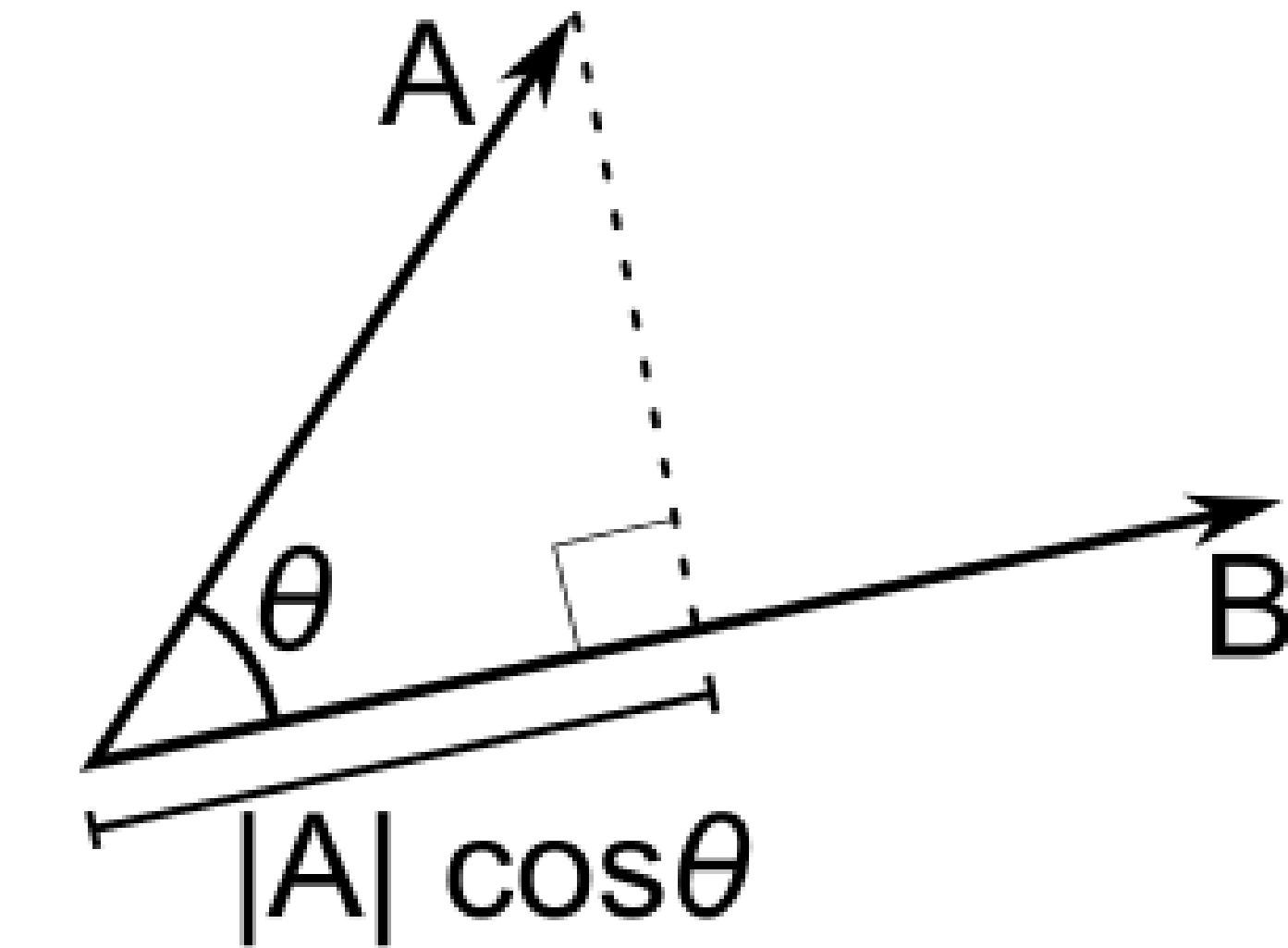
# РАССТОЯНИЕ МЕЖДУ ОБЪЕКТАМИ



euclide (l2)

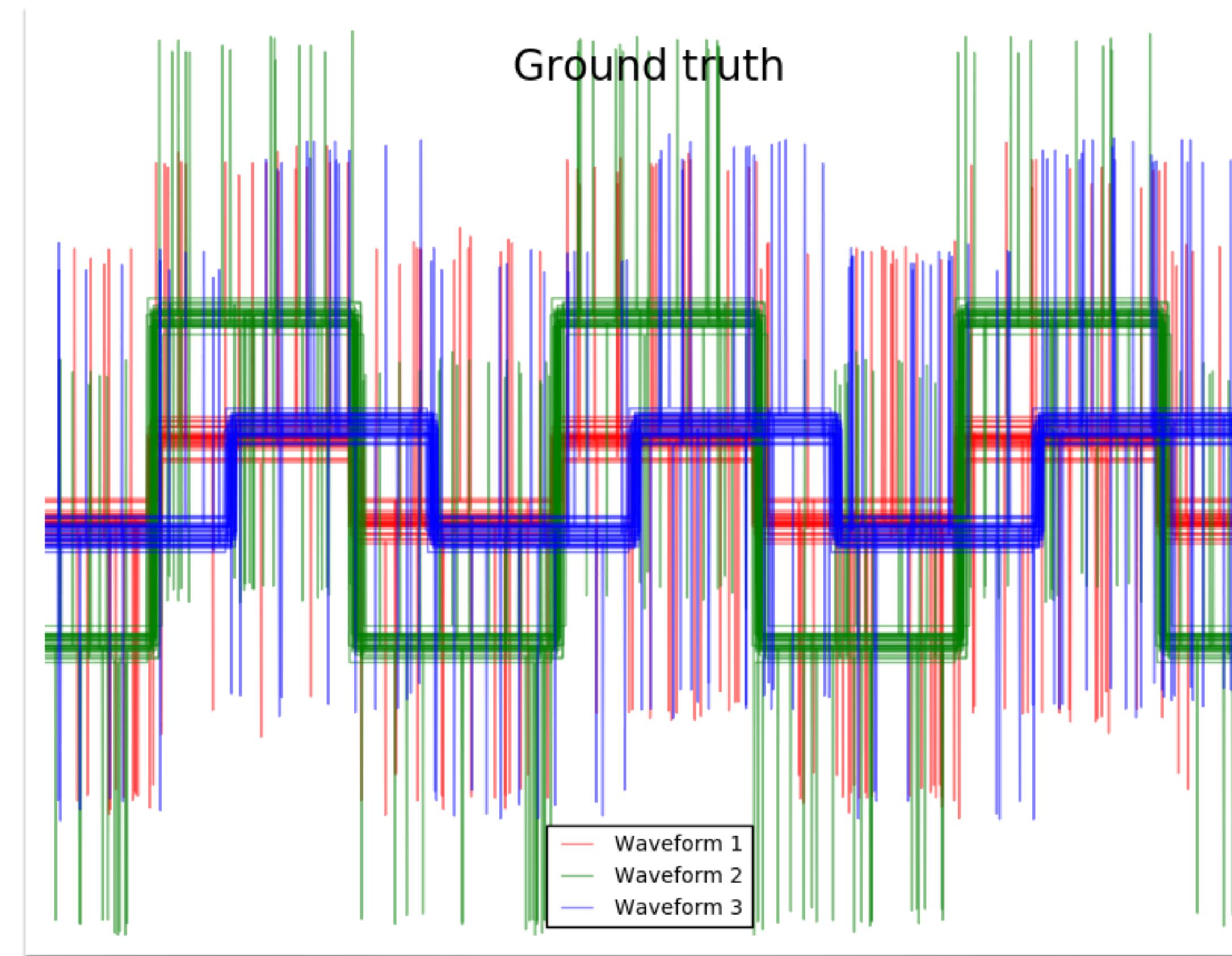


manhattan (l1)



cosine

# РАССТОЯНИЕ МЕЖДУ ОБЪЕКТАМИ

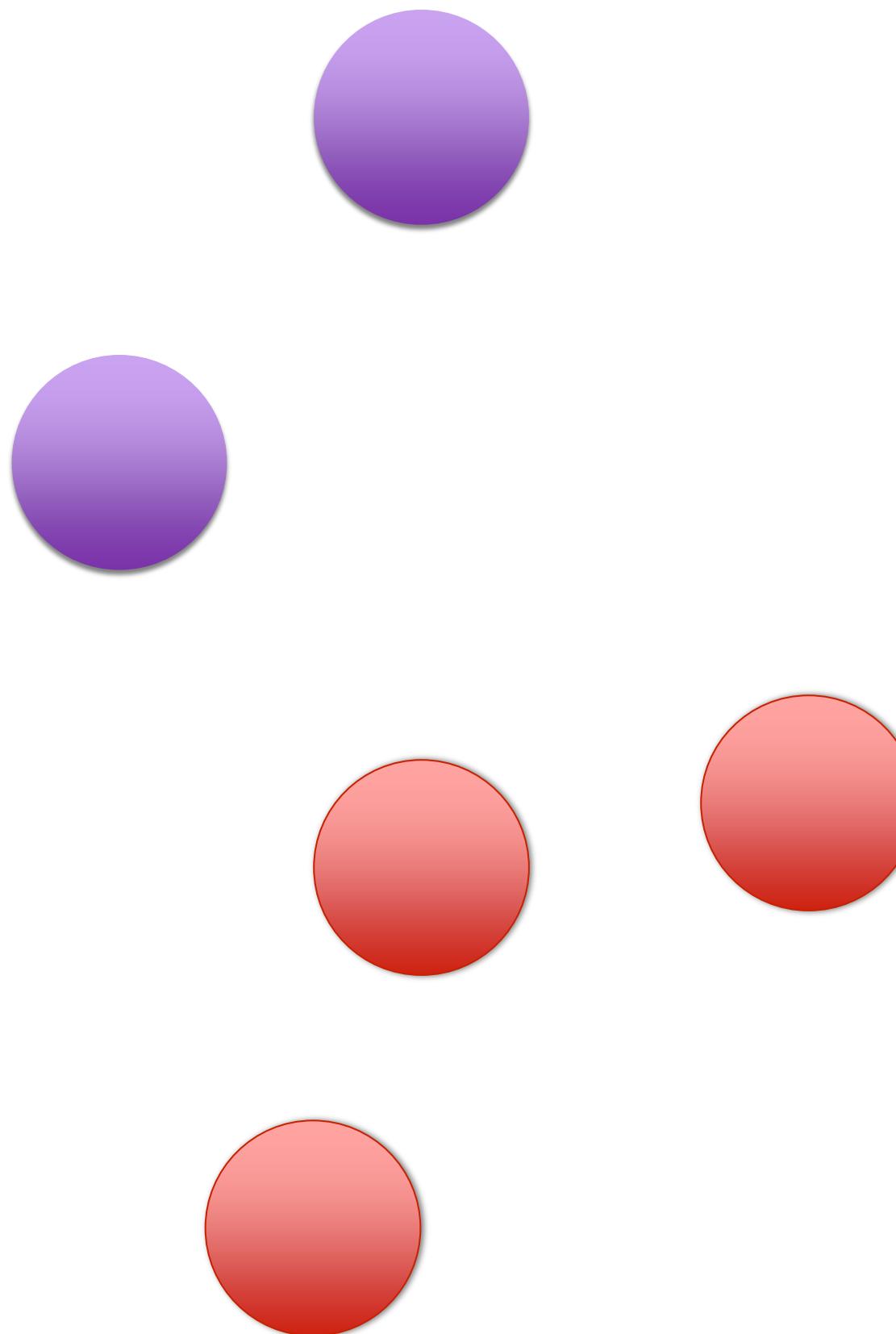


Межклusterное расстояние основывается на расстоянии между объектами. Если с межклusterным расстоянием есть рекомендация брать Уорда, то выбор функции расстояния между объектами более зависит от данных. Слева представлен пример оригинальных данных 3 сигналов, с которыми не справляется косинусное и евклидово расстояние, однако справляется расстояние городских кварталов ( $\| \cdot \|_1$ )

\* sklearn, clustering example

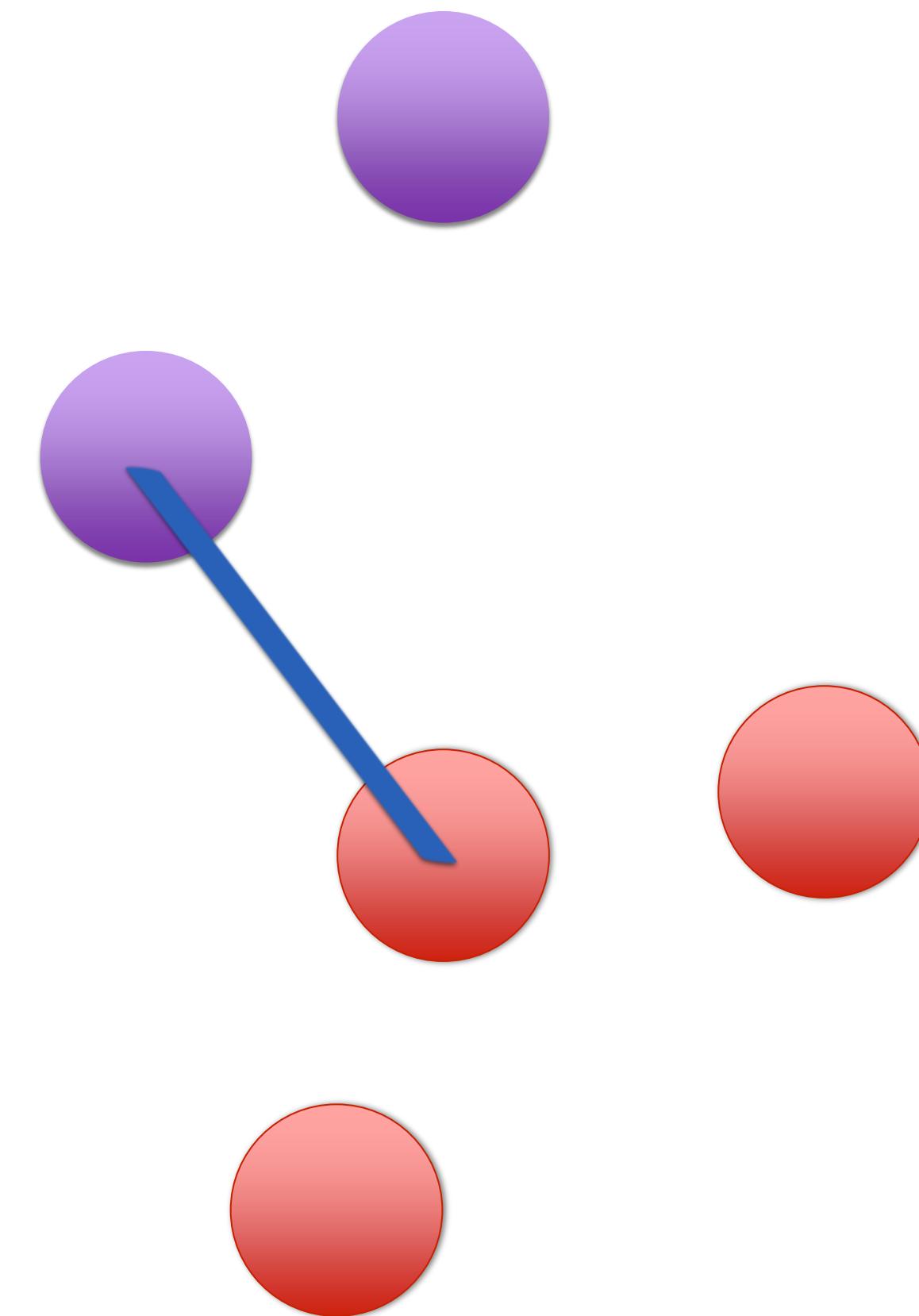
# РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* Ближнего соседа
- \* Дальнего соседа
- \* Групповое среднее
- \* Расстояние между центрами
- \* Расстояние Уорда



# РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* **Ближнего соседа**
- \* Дальнего соседа
- \* Групповое среднее
- \* Расстояние между центрами
- \* Расстояние Уорда

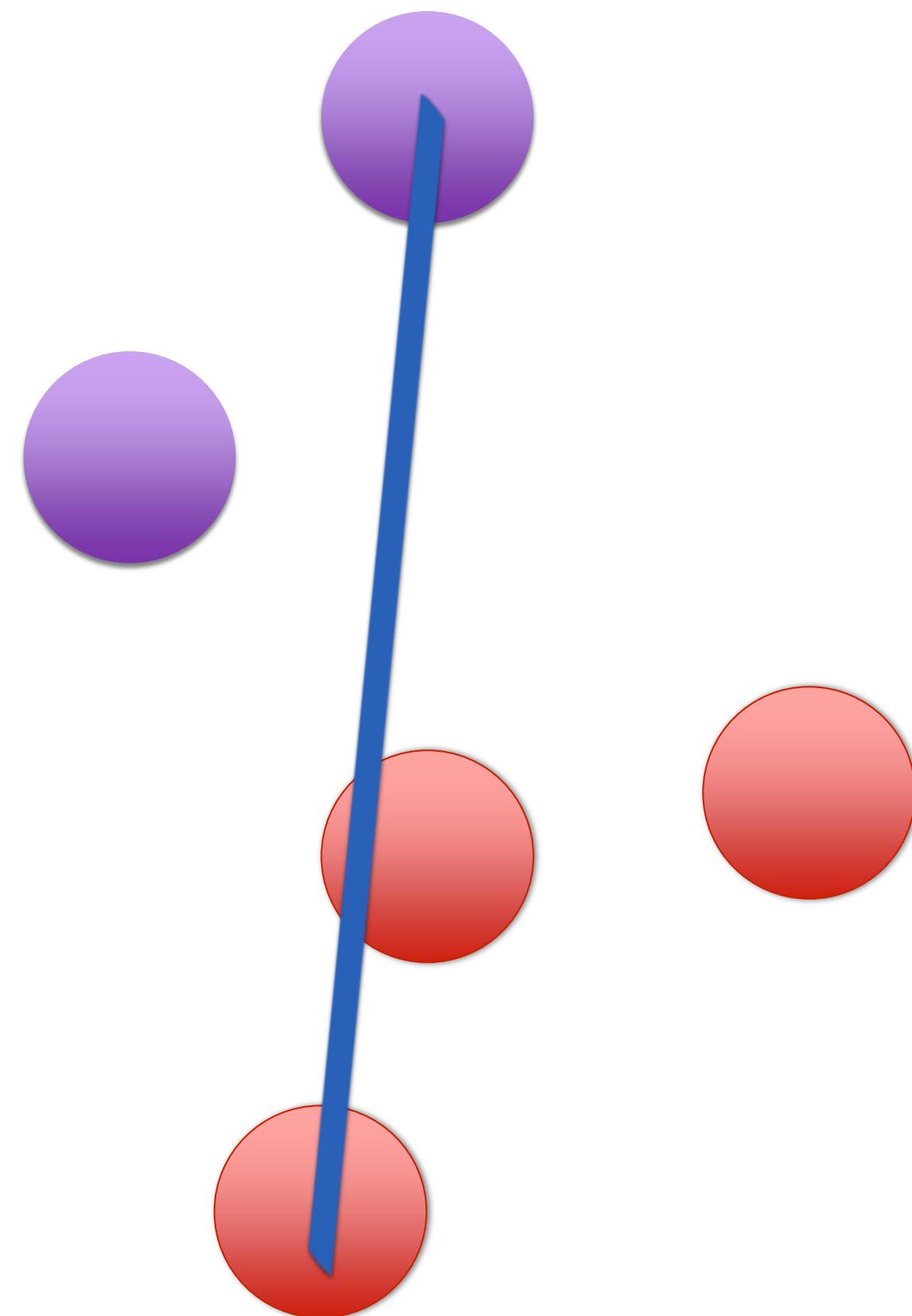


$$R^b(W, S) = \min_{w,s} \rho(w, s)$$

## РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* Ближнего соседа
- \* **Дальнего соседа**
- \* Групповое среднее
- \* Расстояние между центрами
- \* Расстояние Уорда

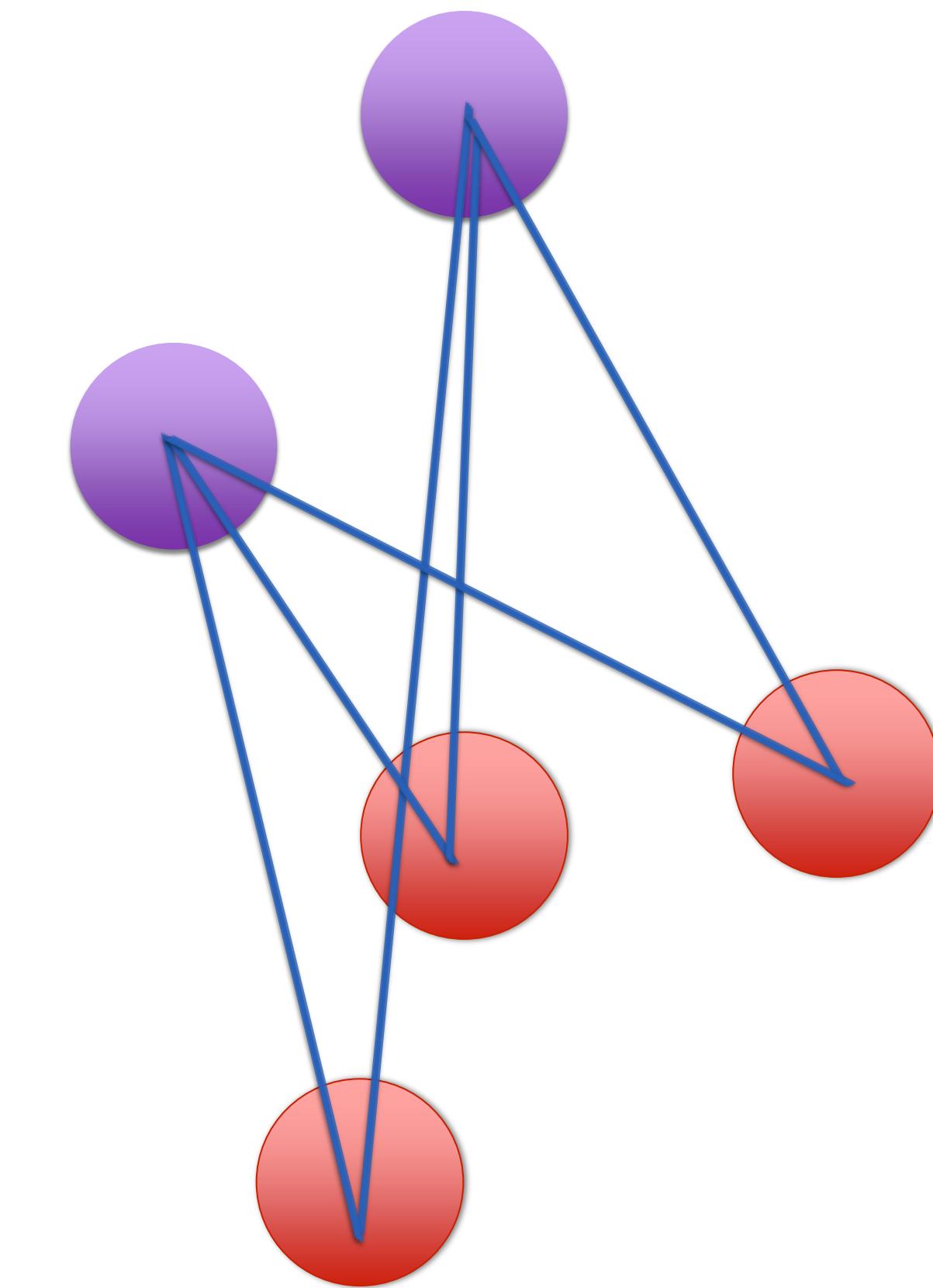
$$R^d(W, S) = \max_{w,s} \rho(w, s)$$



# РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* Ближнего соседа
- \* Дальнего соседа
- \* **Групповое среднее**
- \* Расстояние между центрами
- \* Расстояние Уорда

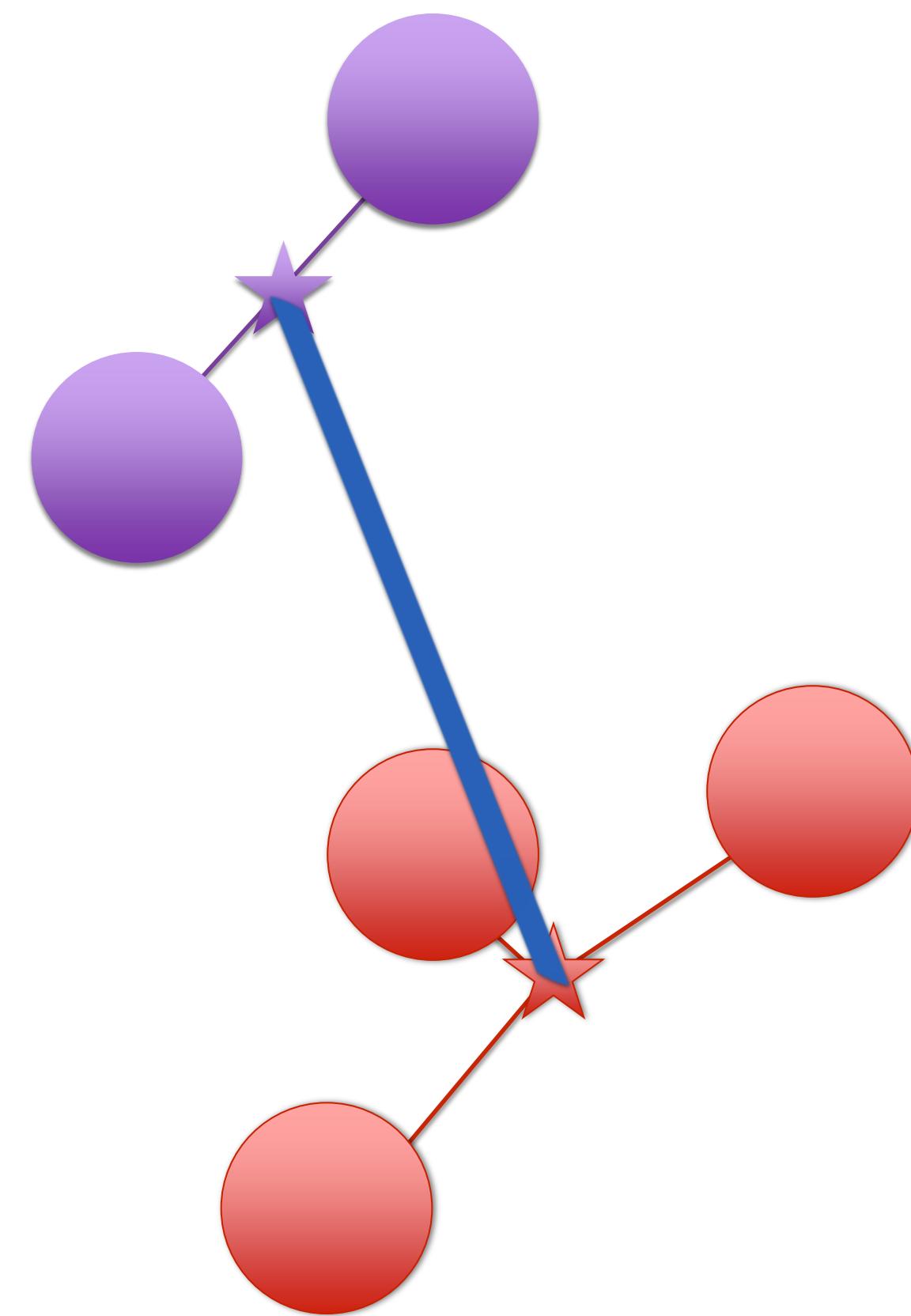
$$R^{\Gamma}(W, S) = \frac{1}{|W| * |S|} \sum_w \sum_s \rho(w, s)$$



# РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* Ближнего соседа
- \* Дальнего соседа
- \* Групповое среднее
- \* **Расстояние между центрами**
- \* Расстояние Уорда

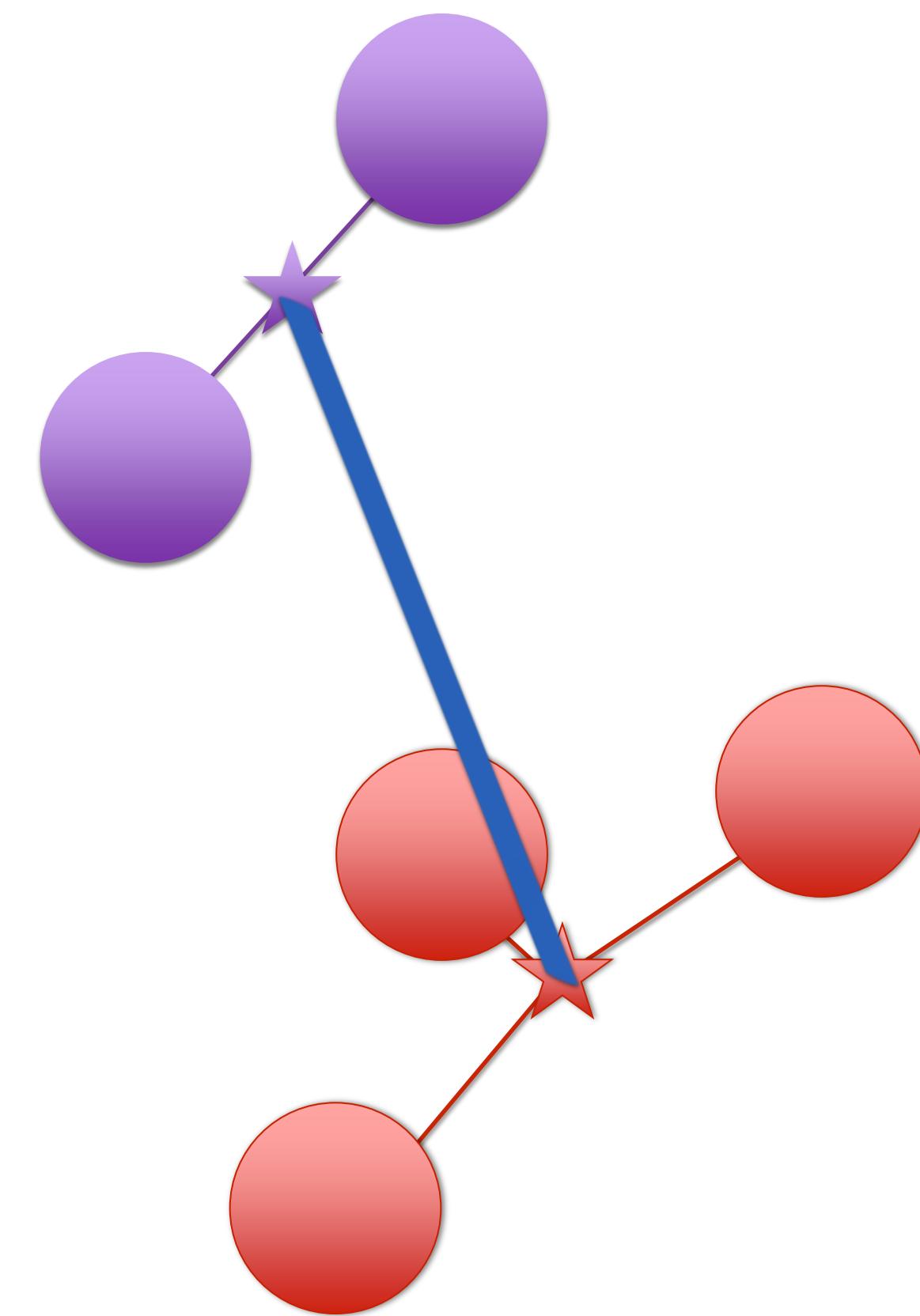
$$R^{\Pi}(W, S) = \rho^2 \left( \sum_w \frac{w}{|W|}, \sum_s \frac{s}{|S|} \right)$$



# РАССТОЯНИЕ МЕЖДУ КЛАСТЕРАМИ

- \* Ближнего соседа
- \* Дальнего соседа
- \* Групповое среднее
- \* Расстояние между центрами
- \* **Расстояние Уорда**

$$R^y(W, S) = \frac{|W| * |S|}{|W| + |S|} \rho^2 \left( \sum_w \frac{w}{|W|}, \sum_s \frac{s}{|S|} \right)$$



## СВОЙСТА РАССТОЯНИЙ

Расстояние **монотонно, если** при каждом слиянии расстояние между кластерами растёт:  $R_2 \leq R_3 \leq R_4 \dots$

*Расстояние между центрами - не монотонно. Остальные - да*

Расстояние **растягивающее**, если при каждом слиянии увеличение расстояний между кластерами растёт:

$$R_3 - R_2 \leq R_4 - R_3 \leq R_5 - R_4 \dots$$

*Расстояние дальнего соседа и Уорда - растягивающие*

## РЕКОМЕНДУЕМОЕ РАССТОЯНИЕ

Расстояние Уорда (Ward)

Оно:

- \* монотонное
- \* растягивающее
- \* работает с центрами кластеров

# РЕАЛИЗАЦИЯ В SKLEARN

## AgglomerativeClustering

- \* n\_clusters=2
- \* affinity='euclidean'
- \* memory=Memory(cachedir=None)
- \* connectivity=None
- \* compute\_full\_tree='auto'
- \* linkage='ward'
- \* pooling\_func=<function mean>

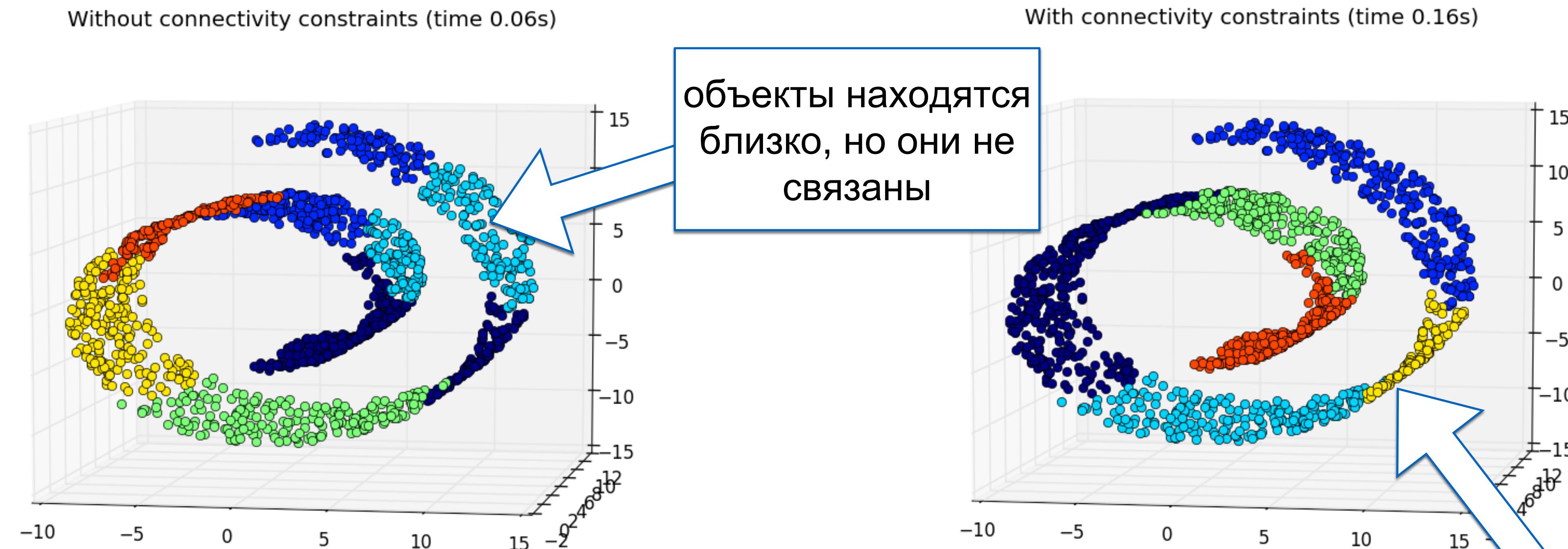
### **Основные параметры**

- \* n\_clusters - количество кластеров для разбиения
- \* linkage: «ward», «complete», «average»
- \* affinity: “euclidean”, “l1”, “l2”, “manhattan”, “cosine”, «precomputed» (для linkage = «ward» доступно только евклидово)
- \* connectivity - априорные знания о структуре данных, подробнее на следующем слайде

### **Основные методы**

- \* fit, fit\_predict

# РЕАЛИЗАЦИЯ В SKLEARN. CONNECTIVITY



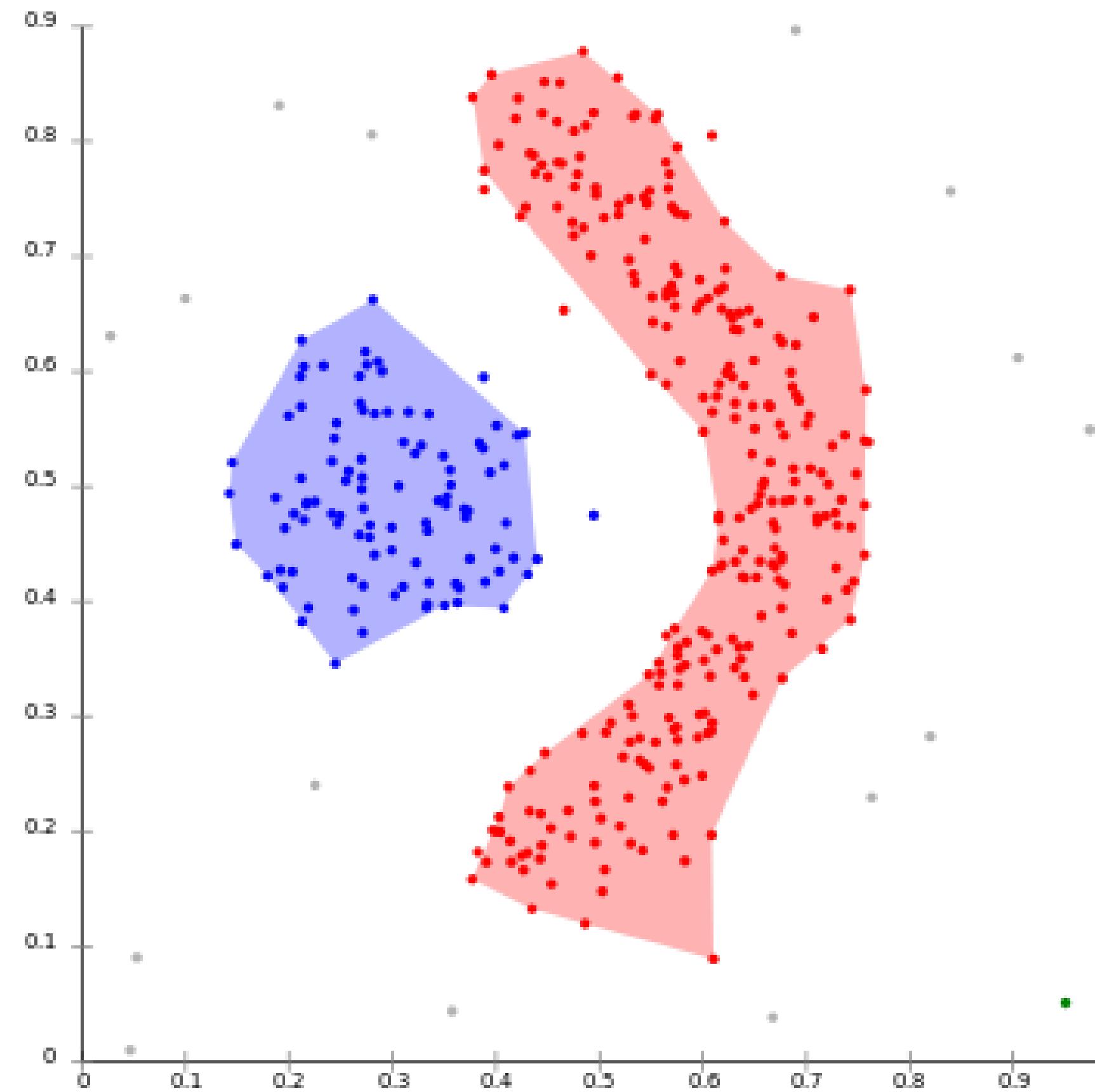
[sklearn full example info](#)

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. DBSCAN

---

# DBSCAN

# ИДЕЯ *Density-Based Spatial Clustering of Applications with Noise*

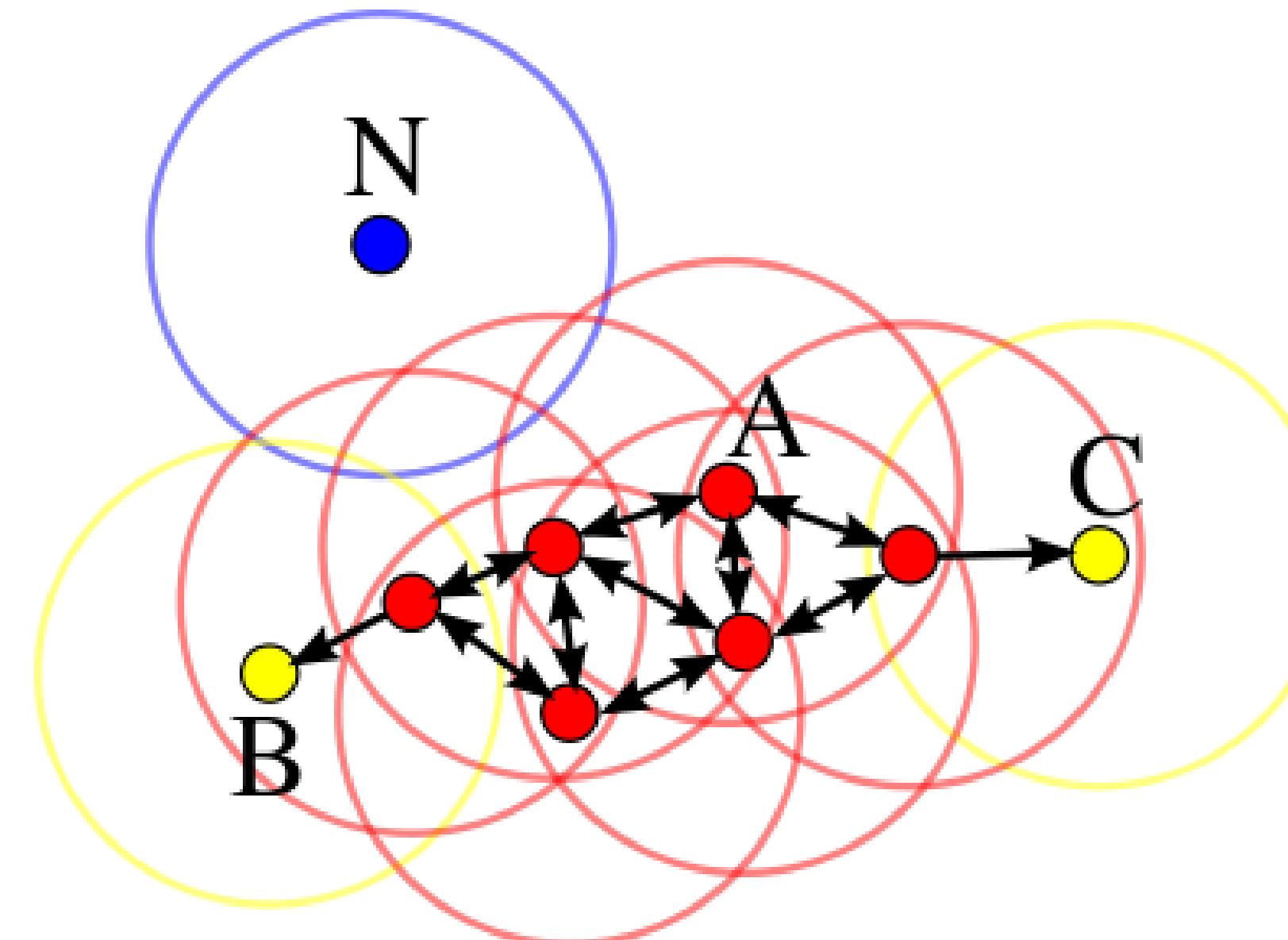


Рассматриваем объекты как ядра,  
вокруг которых собираются другие  
объекты

Если не собираются - это выброс

Если ядра связаны - то они и  
достижимые из них объекты образуют  
кластер

# ИДЕЯ *Density-Based Spatial Clustering of Applications with Noise*



Все точки делятся на 3 типа:

- \* ядра  
(в  $\text{eps}$ -окрестности  $\geq N$  точек)
- \* достижимые из ядра  
(в  $\text{eps}$ -окрестности  $< N$  точек,  $> 0$  ядер)
- \* выбросы  
(остальные)

Ядра и достижимые из них точки образуют кластеры  
Выбросы не принадлежат ни одному кластеру

# РЕАЛИЗАЦИЯ В SKLEARN

## DBSCAN

- \* eps=0.5
- \* min\_samples=5
- \* metric='euclidean'
- \* algorithm='auto'
- \* leaf\_size=30
- \* p=None
- \* n\_jobs=1

### **Основные параметры**

- \* eps - размер окрестности
- \* min\_samples - кол-во точек в окрестности ядра
- \* n\_jobs - кол-во процессоров для расчёта (-1 - max)

### **Основные методы**

- \* fit, fit\_predict

# ДОСТОИНСТВА И НЕДОСТАТКИ

## **Достоинства:**

- \* не нужно указывать кол-во кластеров
- \* произвольная форма данных
- \* обнаруживает выбросы

## **Недостатки:**

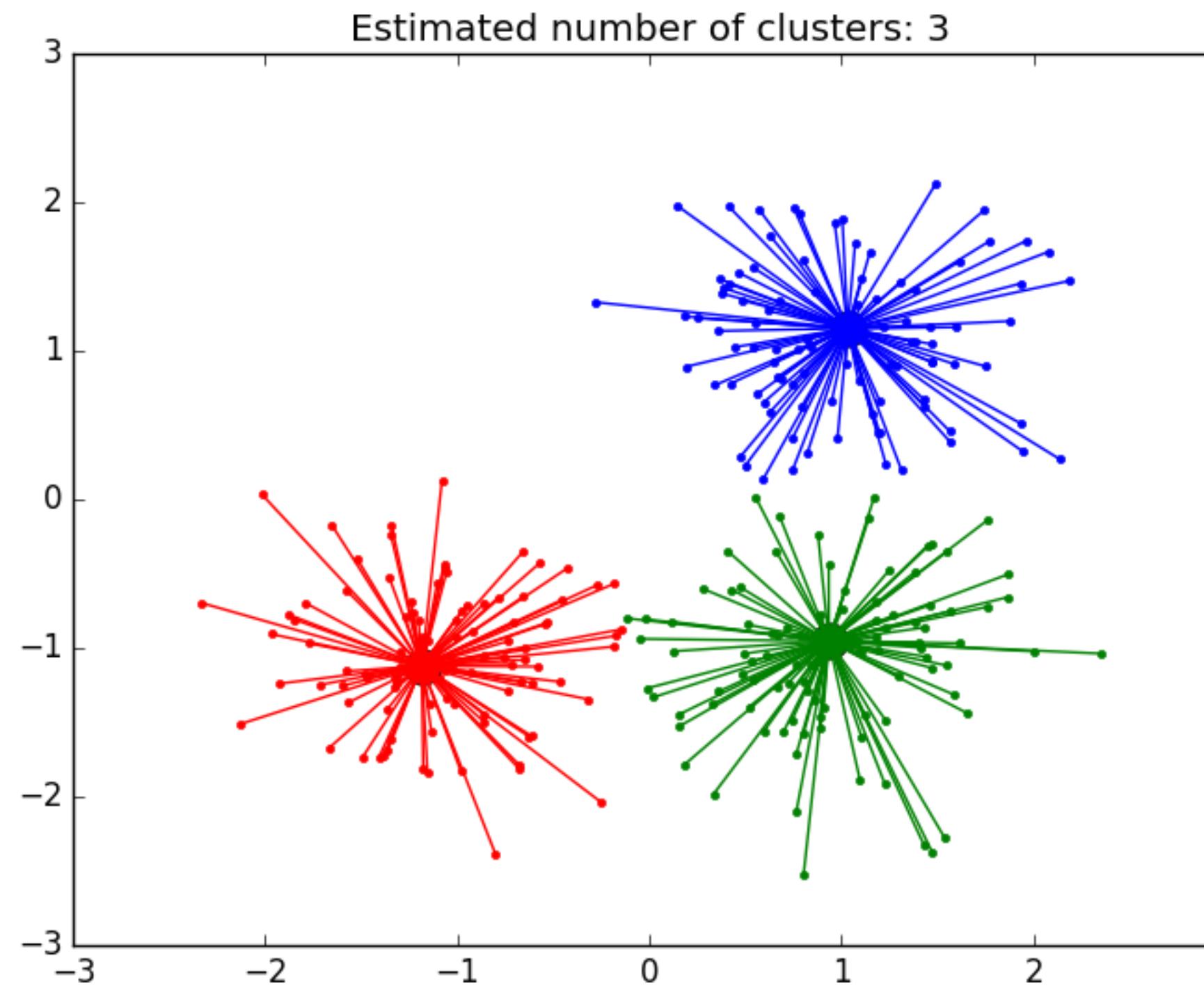
- \* сложность выбора комбинации  $\text{eps}+\text{N}$
- \* плохо работает с кластерами разной плотности

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. AFFINITY PROPAGATION

---

# AFFINITY PROPAGATION

# ИДЕЯ



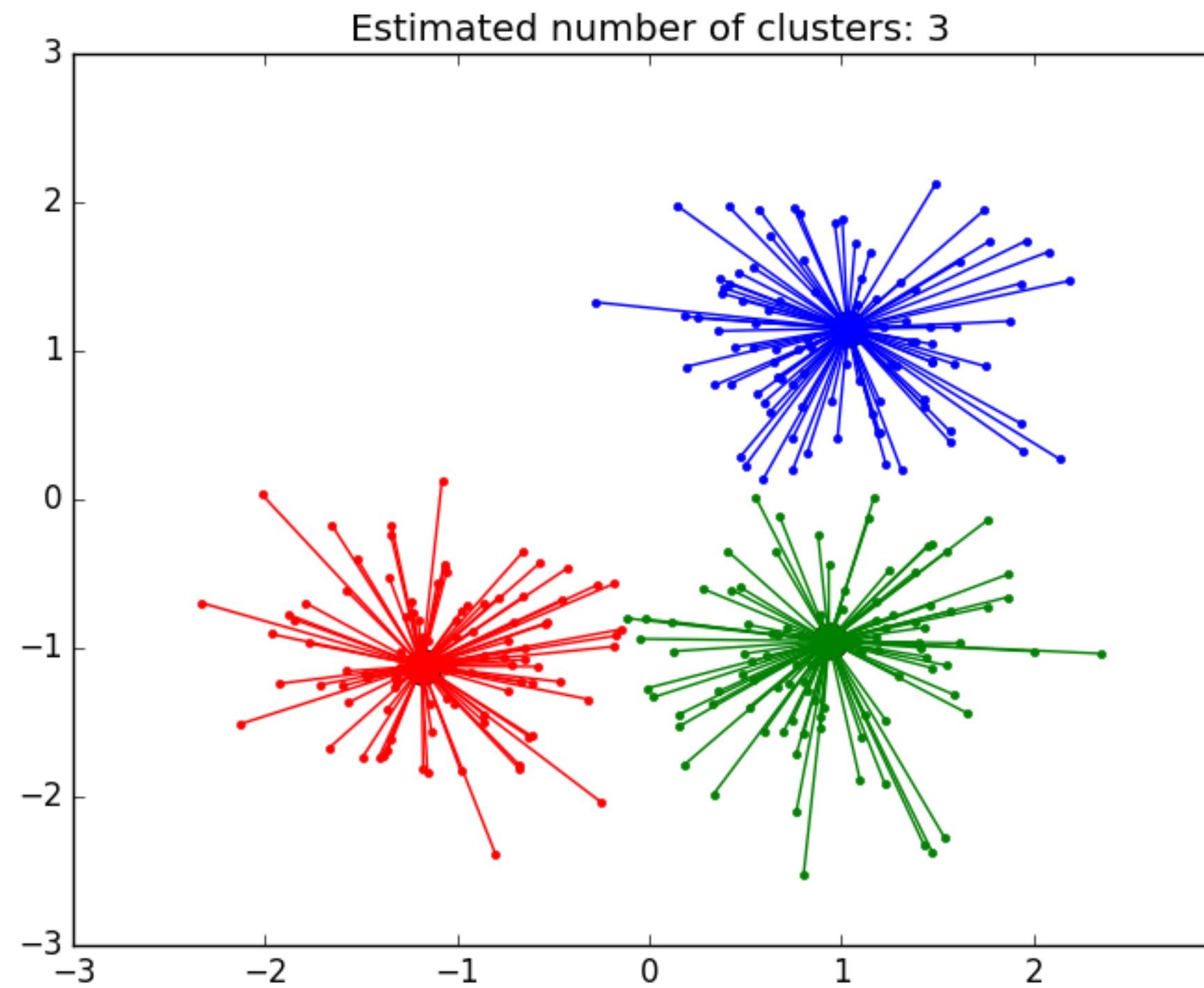
Объекты обмениваются двумя видами сообщений:

- \* насколько объект 1 готов быть экземпляром объекта 2
- \* насколько объект 2 готов предоставить право быть объекту 1 своим экземпляром

**Итог:**

К объектов - представителей кластеров

# АЛГОРИТМ



**Установить:**

$$r(i,i) = 0, a(i,i) = 0$$

**Повторять, пока** экземпляры меняются:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k} \{a(i,k') + s(i,k')\}$$

$$a(i,k) \leftarrow \min \left( 0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k)) \right) \text{ for } i \neq k$$

$$a(k,k) \leftarrow \sum_{i' \neq k} \max(0, r(i',k)).$$

**Итог:**

экземпляры с  $r(i,i)+a(i,i) > 0$

# РЕАЛИЗАЦИЯ В SKLEARN

## Affinity Propogation

- \* damping=0.5
- \* max\_iter=200
- \* convergence\_iter=15
- \* copy=True
- \* preference=None
- \* affinity='euclidean'
- \* verbose=False

### **Основные параметры**

- \* preference - априорные знания о возможности быть экземпляром
- \* damping - скорость затухания [0.5-1]
- \* convergence\_iter - условие останова, сколько должно пройти итераций без изменений

### **Основные методы**

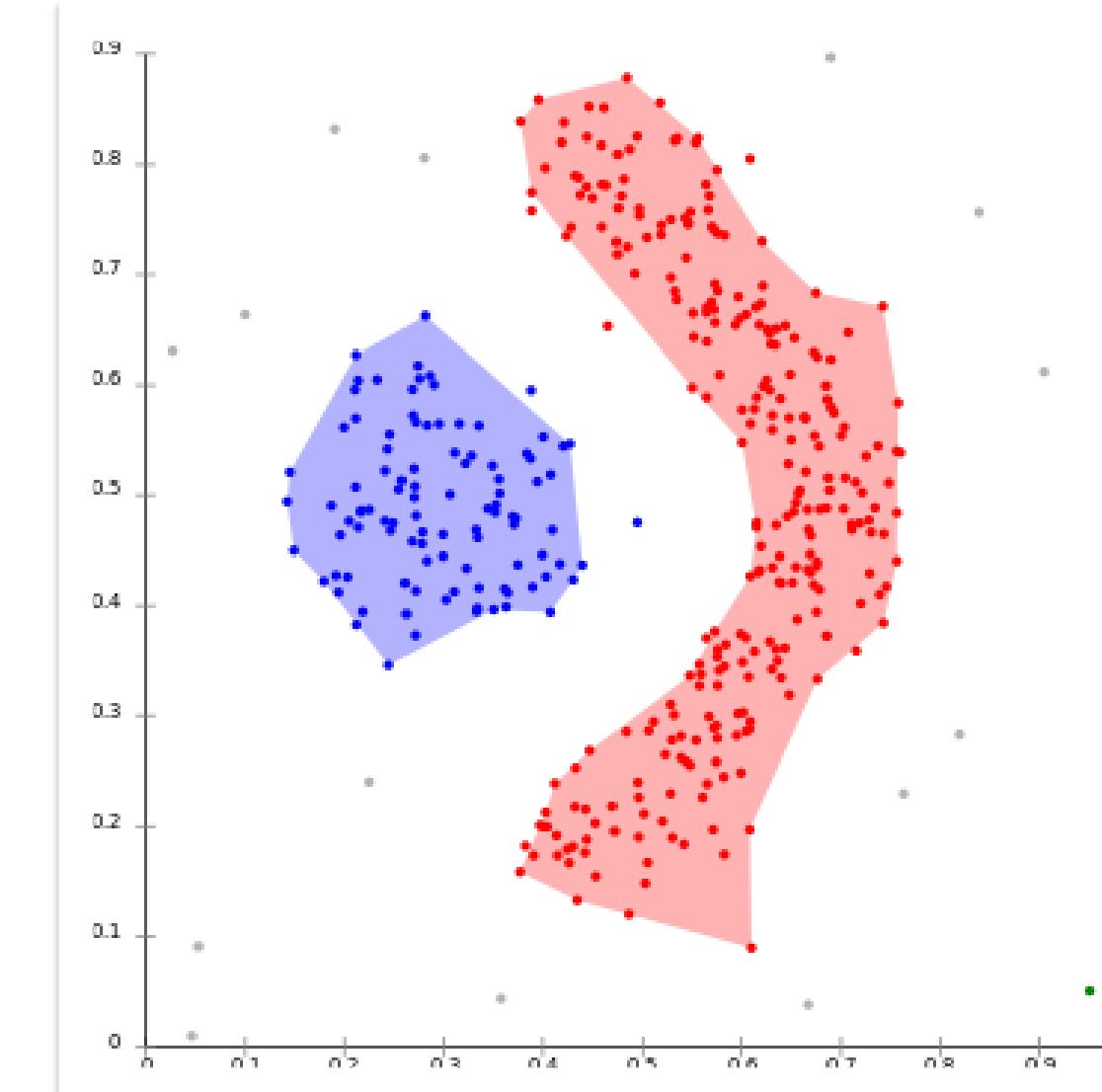
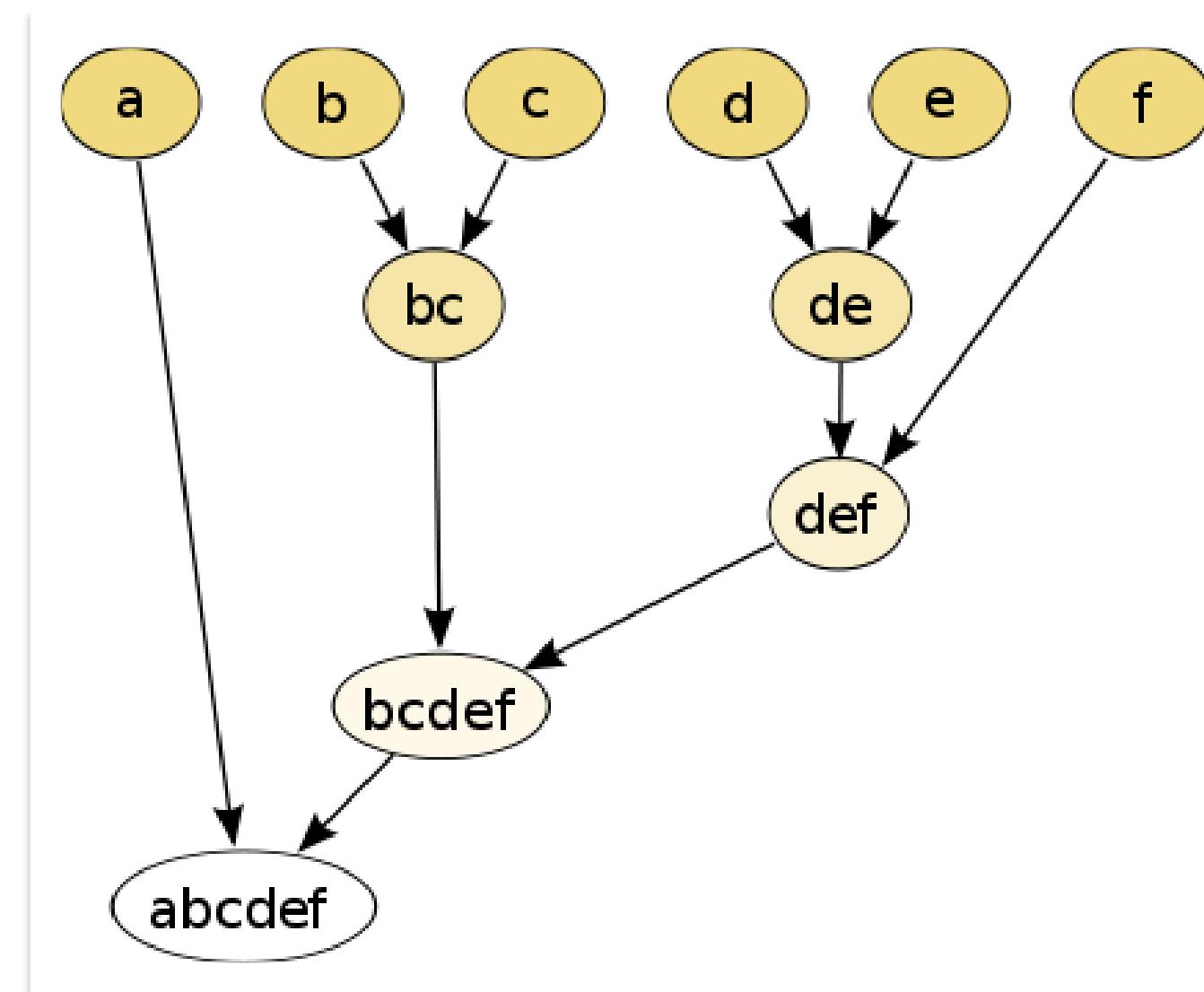
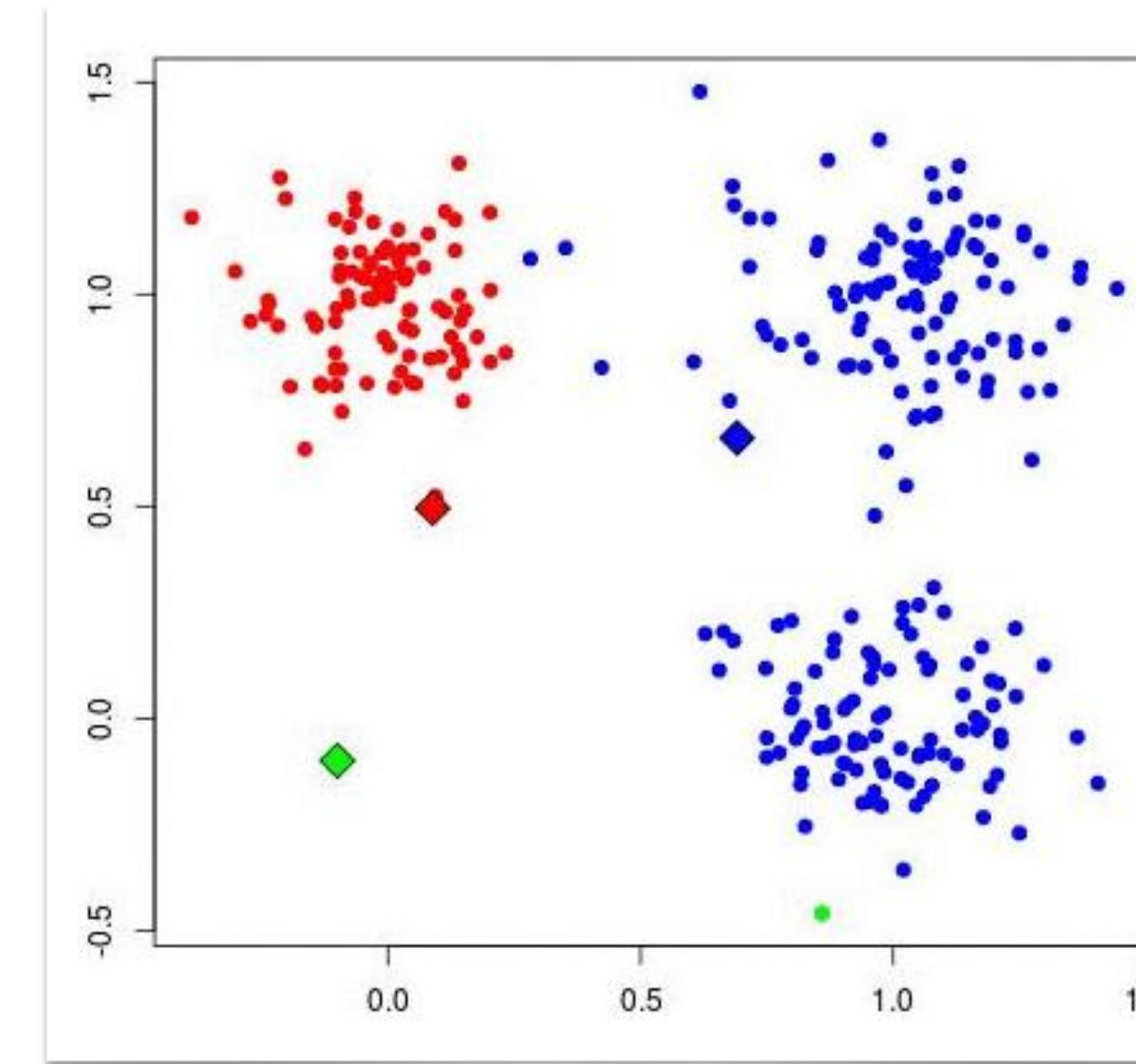
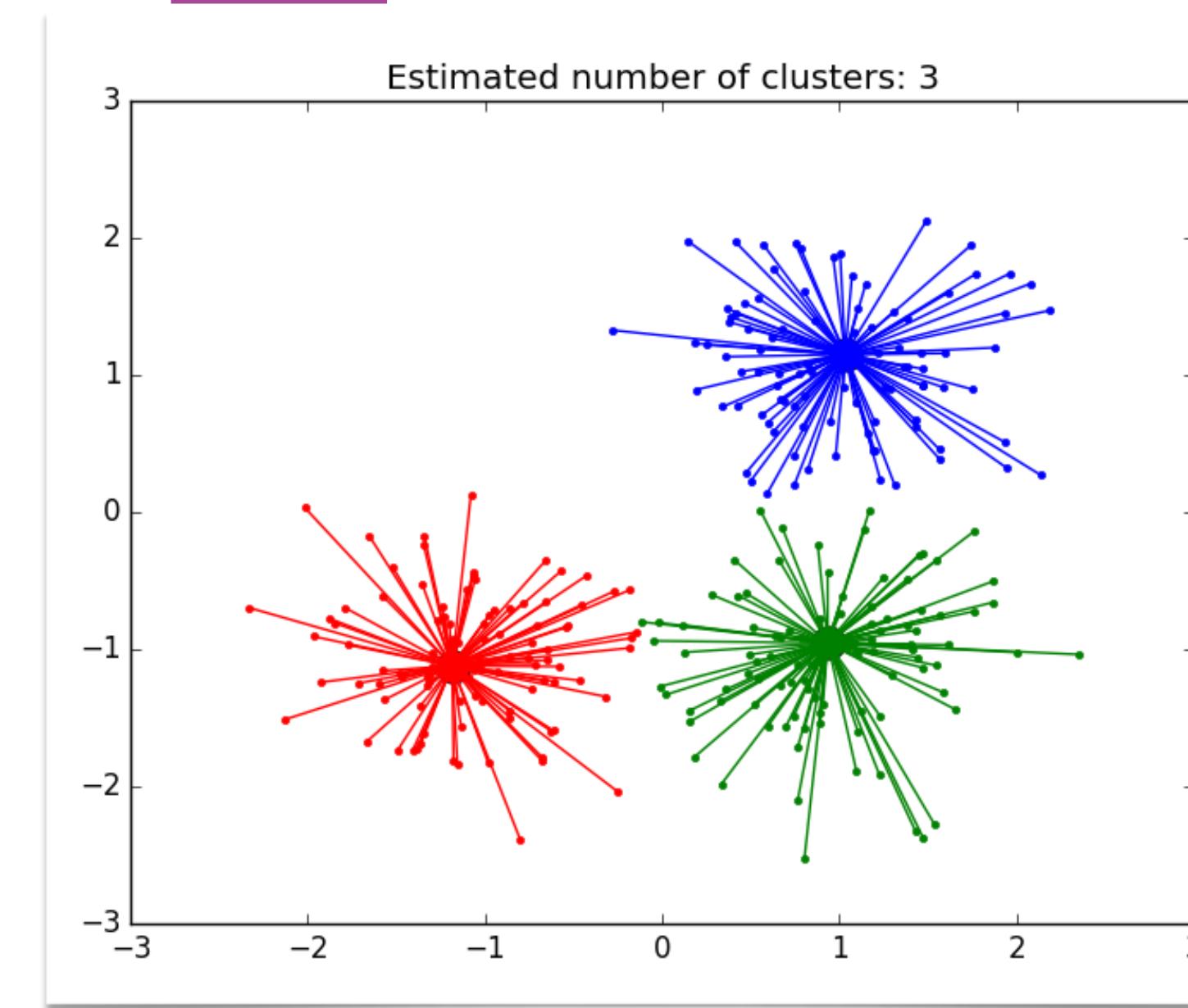
- \* fit, fit\_predict

АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. QUIZ

---

QUIZ

## АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ. QUIZ



K-Means

Agglomerative

DBSCAN

Affinity Propogation

# КАКОЙ АЛГОРИТМ ВЫБРАТЬ?

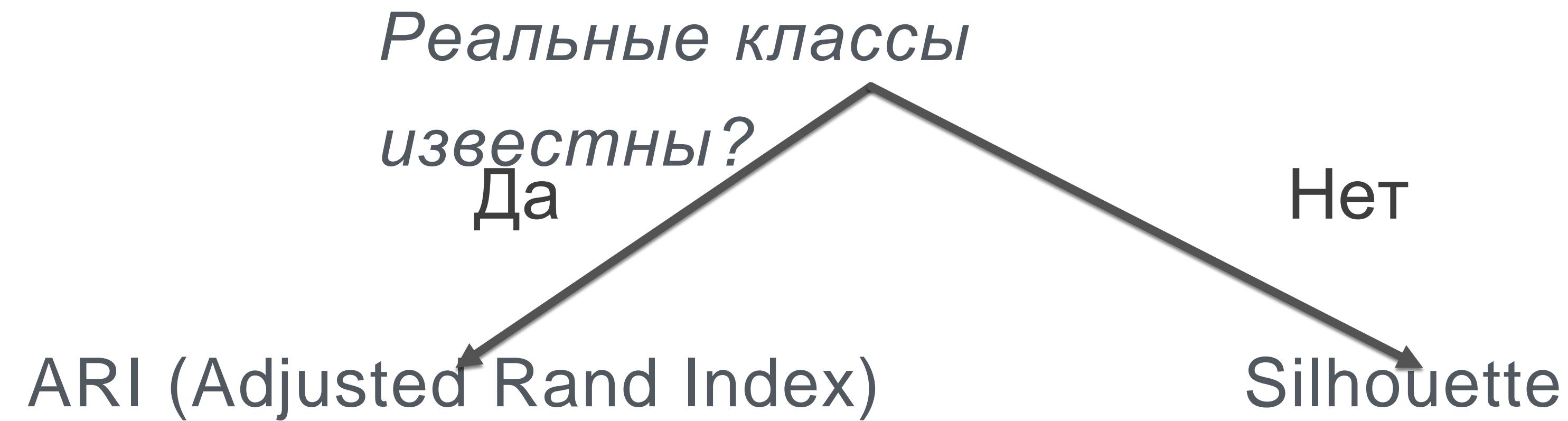
Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large n_samples, medium n_clusters with <a href="#">MiniBatch code</a>	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Agglomerative clustering	number of clusters, linkage type, distance	Large n_samples and n_clusters	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large n_samples, medium n_clusters	Non-flat geometry, uneven cluster sizes	Distances between nearest points

\* [sklearn](#), сравнение кластеризаторов

---

## 3. МЕТРИКИ КАЧЕСТВА КЛАСТЕРИЗАЦИИ

# МЕТРИКИ КАЧЕСТВА



\* *sklearn, metrics guide*

\* *sklearn, metrics descriptions*

# ARI: ADJUSTED RAND INDEX

**Дано:**

y\_pred - вектор меток кластеризации

[0, 0, 0, 1, 1, 1]

y\_true - реальные кластеры

[2, 2, 2, 7, 7, 7]

$ARI \in [-1, 1]$ ;

1 - точное соответствие

$ARI(y_{pred}, y_{true}) = 1$

0 - случайное разбиение кластеров

**Метрике не важны названия кластеров**

1

2

4

3

5

## СИЛУЭТ

**нет** знания правильных кластеров.

Оценим, насколько сильно **один объект** сидит внутри своего кластера и далеко от ближайшего соседнего:

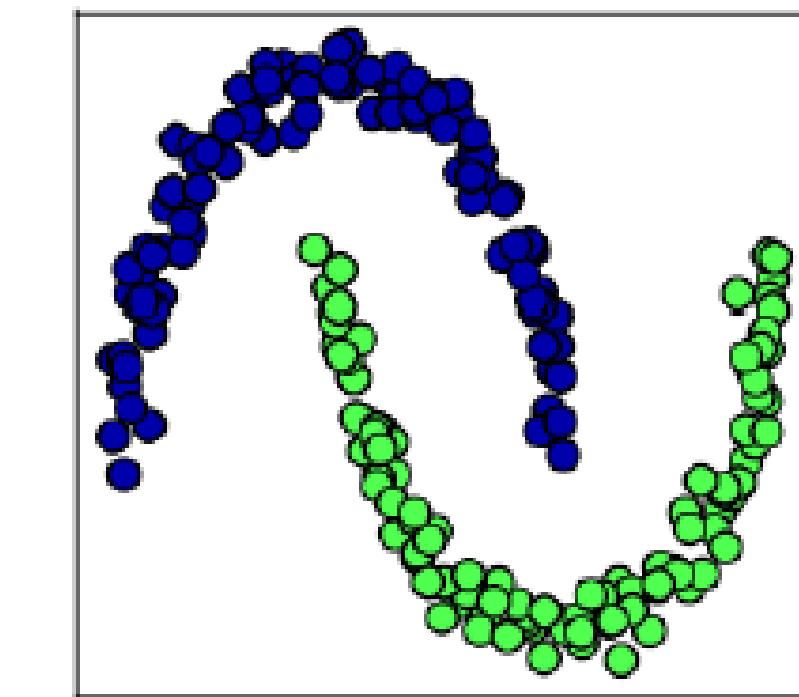
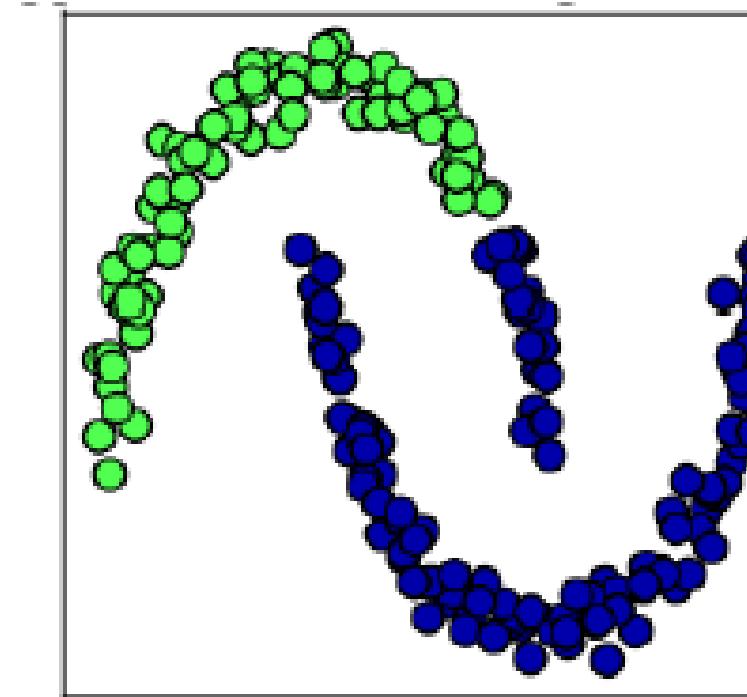
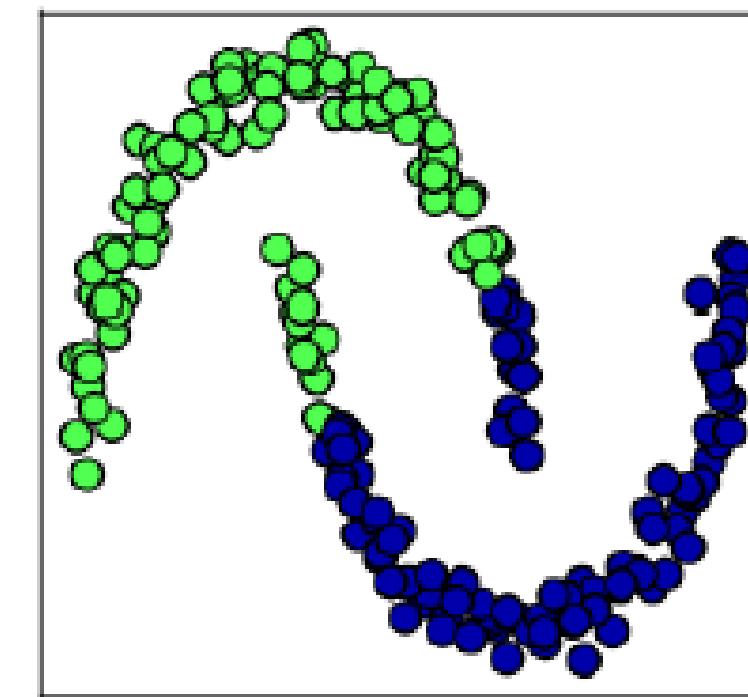
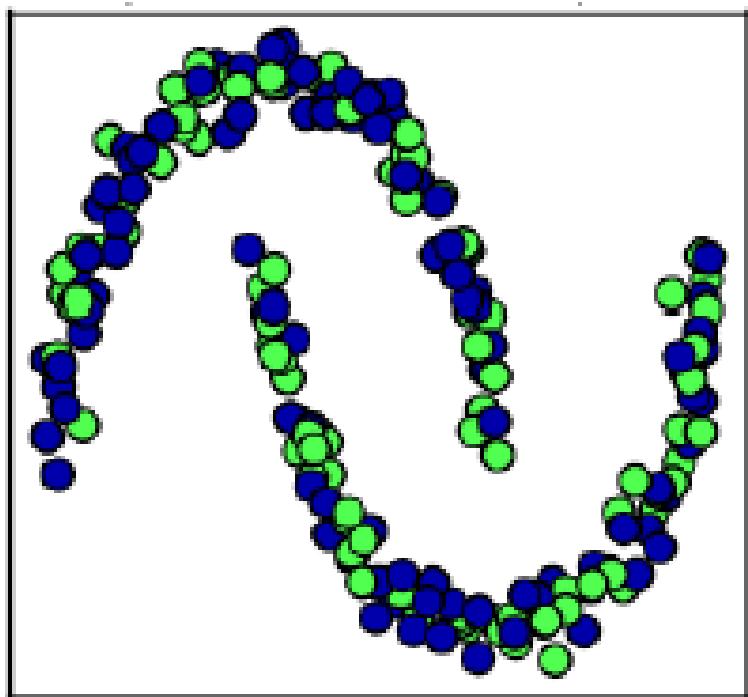
$$s = \frac{b - a}{\max(a, b)}$$

*a* - среднее расстояние до объектов внутри кластера  
*b* - среднее расстояние до объектов ближайшего кластера

$$S = \text{mean}(s)$$

среднее значение по всем объектам - силуэт

## СРАВНЕНИЕ МЕТРИК



<b>ARI</b>	0.00	0.50	0.61	1.00
<b>Silhouette</b>	0.00	0.49	0.46	0.38

---

ЧТО МЫ СЕГОДНЯ УЗНАЛИ

## ЧТО МЫ СЕГОДНЯ УЗНАЛИ

---

1. Кластеризация позволяет **находить структуру** в незамеченных данных, что может послужить **дополнительными признаками** обучения или являться **самодостаточной целью**
2. В задаче кластеризации **нет правильного решения**.  
Метрики качества служат лишь слабым приближением для создания новых алгоритмов или нахождениям критерия останова
3. Разные алгоритмы кластеризации принципиально **работают по-разному**, для конкретного набора данных необходимо выбирать наиболее подходящий

---

# ПОЛЕЗНЫЕ МАТЕРИАЛЫ

## ПОЛЕЗНЫЕ МАТЕРИАЛЫ

---

1. [Документация sklearn по кластеризации](#)
2. [Метрики sklearn для задач кластеризации](#)
3. [Open Data Science, habrahabr: Обучение без учителя: PCA и кластеризация](#)
4. [Книжка: Introduction to ML with Python](#)



НЕТОЛОГИЯ  
групп

Спасибо за внимание!

АЛЕКСЕЙ КУЗЬМИН  aleksej.kyzmin@gmail.com