

# К-means (Евдокимов)

## Описание программы

Задача программы состоит в кластеризации точек на плоскости с помощью алгоритма k-means в Python.

Для начала по заданным данным строится модель k-means с помощью библиотеки sklearn. Затем для определения эффективности модели или корректности вводимых данных, проверяю получившиеся кластеры на наличие точек пересечения. Затем программа сверяет изначально заданные кластеры с полученными при помощи k-means, и демонстрирует совпадения. Если мы имеем 100% совпадений, то наш k-means отлично сработала для этих данных. После этого выводятся кластеры, полученные методом Уорда (иерархическая кластеризация), и тоже сравниваются с изначальноными. Затем два полученных набора кластеров сравниваются между собой.

## Тесты и эксперименты

(Можно самостоятельно запустить тесты в файле [test.py](#)) Проверим программу на разных данных:

### Тест на случайных точках

Генирируются 2 случайных кластера по 300 точек. С большой вероятностью никогда не получится так, что модель когда-нибудь со 100% точностью сопоставит кластеры. Но чисто теоритически, это может когда-то произойти.

### Тестируем на простых квадратах

Оба вида кластеризации хорошо справляются с этой задачей. Явные 3 квадрата были разделены на 3 кластера. Фото в папке.

#### Результат:

Запускаю k-means

Одинаковые кластеры

```
[[2, 2], [2, 3], [2, 4], [3, 2], [3, 3], [3, 4], [4, 2], [4, 3], [4, 4]]
```

```
[[2, 2], [2, 3], [2, 4], [3, 2], [3, 3], [3, 4], [4, 2], [4, 3], [4, 4]]
```

```
|-|
```

Одинаковые кластеры

```
[[6, 6], [6, 7], [6, 8], [7, 6], [7, 7], [7, 8], [8, 6], [8, 7], [8, 8]]
```

```
[[6, 6], [6, 7], [6, 8], [7, 6], [7, 7], [7, 8], [8, 6], [8, 7], [8, 8]]
```

```
|-|
```

Одинаковые кластеры

```
[[8, 2], [8, 3], [8, 4], [9, 2], [9, 3], [9, 4], [10, 2], [10, 3], [10, 4]]
```

```
[[8, 2], [8, 3], [8, 4], [9, 2], [9, 3], [9, 4], [10, 2], [10, 3], [10, 4]]
```

```
|-|
```

Данный вид кластеризации работает отлично

Запускаю иерархическую кластеризацию по методу Уорда

Одинаковые кластеры

```
[[2, 2], [2, 3], [2, 4], [3, 2], [3, 3], [3, 4], [4, 2], [4, 3], [4, 4]]
```

```
[[2, 2], [2, 3], [2, 4], [3, 2], [3, 3], [3, 4], [4, 2], [4, 3], [4, 4]]
```

```
|-|
```

Одинаковые кластеры

```
[[6, 6], [6, 7], [6, 8], [7, 6], [7, 7], [7, 8], [8, 6], [8, 7], [8, 8]]
```

```
[[6, 6], [6, 7], [6, 8], [7, 6], [7, 7], [7, 8], [8, 6], [8, 7], [8, 8]]
```

|-|

Одинаковые кластеры

[[8, 2], [8, 3], [8, 4], [9, 2], [9, 3], [9, 4], [10, 2], [10, 3], [10, 4]]

[[8, 2], [8, 3], [8, 4], [9, 2], [9, 3], [9, 4], [10, 2], [10, 3], [10, 4]]

|-|

Данный вид кластеризации работает отлично

## **Ромб, треугольник, квадрат**

Три впритык расположенные фигуры. Верно определил k-means только квадрат, остальные точки фигур перемешались в кластерах. Уорд не справился вовсе. Фото в папке.

## **Окружность с точкой внутри**

В этот раз Уорд смог распознать точку внутри окружности как отдельный кластер, в то время как кольцо находится в отдельном кластере. K-means с задачей не справился Фото в папке.

## **Блоки точек и диагональ**

Ни один из видов кластеризации не справился с задачей, кластеры перемешались. Фото в папке.

## **Выводы**

Смотря на результаты тестов, можно утверждать, что для разных данных требуются тщательный анализ и разные подходы к кластеризации. Нельзя ограничиться только k-means и иерархической кластеризацией. Например, алгоритм выделения связных компонент, минимального покрывающего дерева или послойная кластеризация. В дополнении, проведя несколько тестов, можно заметить, что точность k-means растет в зависимости от количества данных точек: чем больше точек - больше точность. Алгоритм Уорда же начинает работать дольше, так как вычислительная сложность составляет  $O(n^2)$ .