



ТЕХНОСФЕРА

Лекция 3 Алгоритмы кластеризации

Николай Анохин

13 марта 2015 г.

Краткое содержание предыдущей лекции

Дано. N обучающих D -мерных объектов $\mathbf{x}_i \in \mathcal{X}$, образующих тренировочный набор данных (training data set) X .

Найти. Модель $h^*(\mathbf{x})$ из семейства параметрических функций $H = \{h(\mathbf{x}, \theta) : \mathcal{X} \times \Theta \rightarrow \mathbb{N}\}$, ставящую в соответствие произвольному $\mathbf{x} \in \mathcal{X}$ один из K кластеров так, чтобы объекты внутри одного кластера были похожи, а объекты из разных кластеров различались.

Краткое содержание предыдущей лекции

- Смоделировали данные как смесь нормальных распределений

$$p(\mathbf{x}|\mu_k, \Sigma_k) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

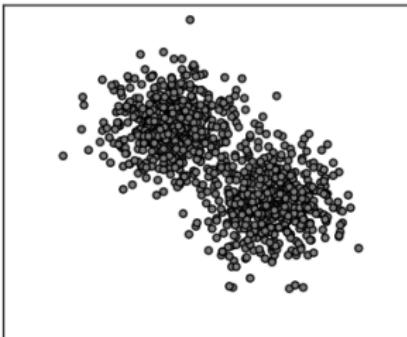
- С применением Maximum Likelihood получили выражения для шагов E и M
- Предельным переходом получили алгоритм k-means, минимизирующий инерцию

$$\tilde{J}(\mu) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} d(\mathbf{x}_n, \mu_k), \quad r_{nk} = \begin{cases} 1, & \text{для } k = \arg \min_j d(\mathbf{x}_n, \mu_j) \\ 0, & \text{иначе} \end{cases}$$

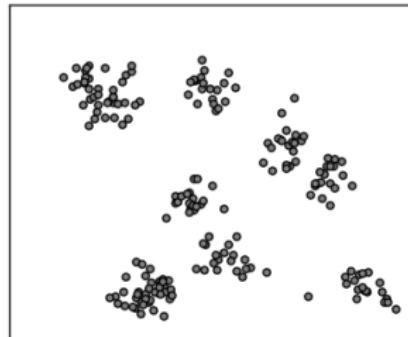
- Формулируя альтернативные $d(\mathbf{x}_1, \mathbf{x}_2)$ и $J(\mu)$, получили семейство алгоритмов, обучаемых с помощью EM



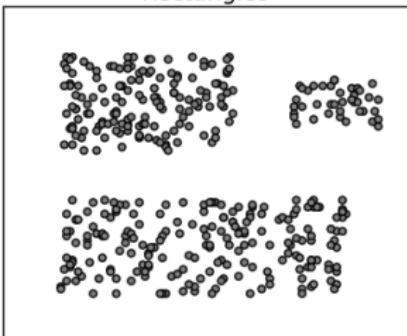
2 clusters



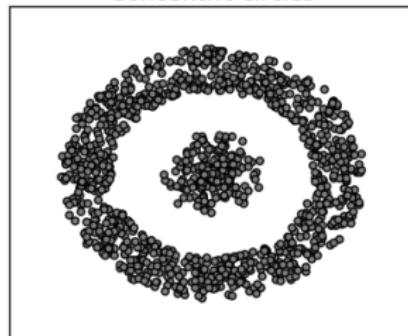
10 clusters



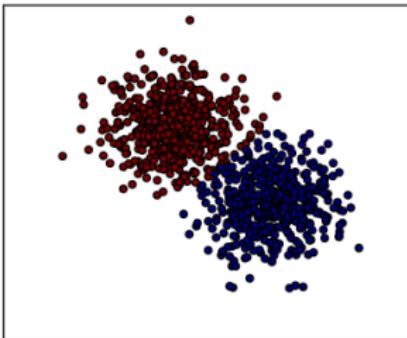
Rectangles



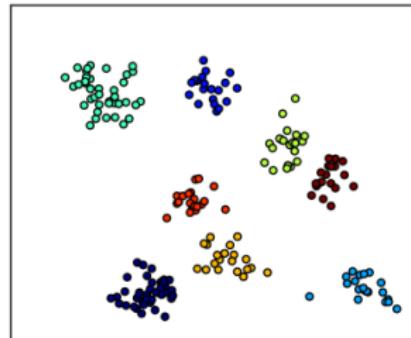
Concentric circles



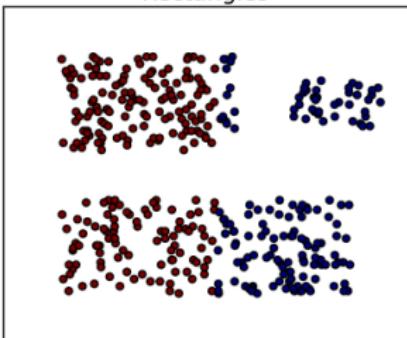
2 clusters



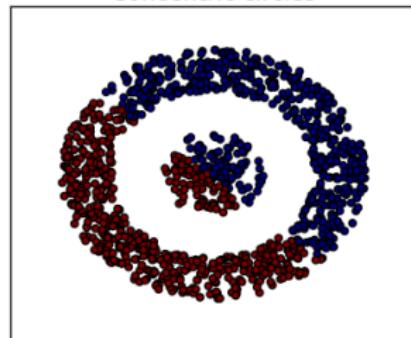
10 clusters



Rectangles



Concentric circles



План занятия

Иерархическая кластеризация

DBSCAN

Выбор количества кластеров

Иерархическая кластеризация: идея метода

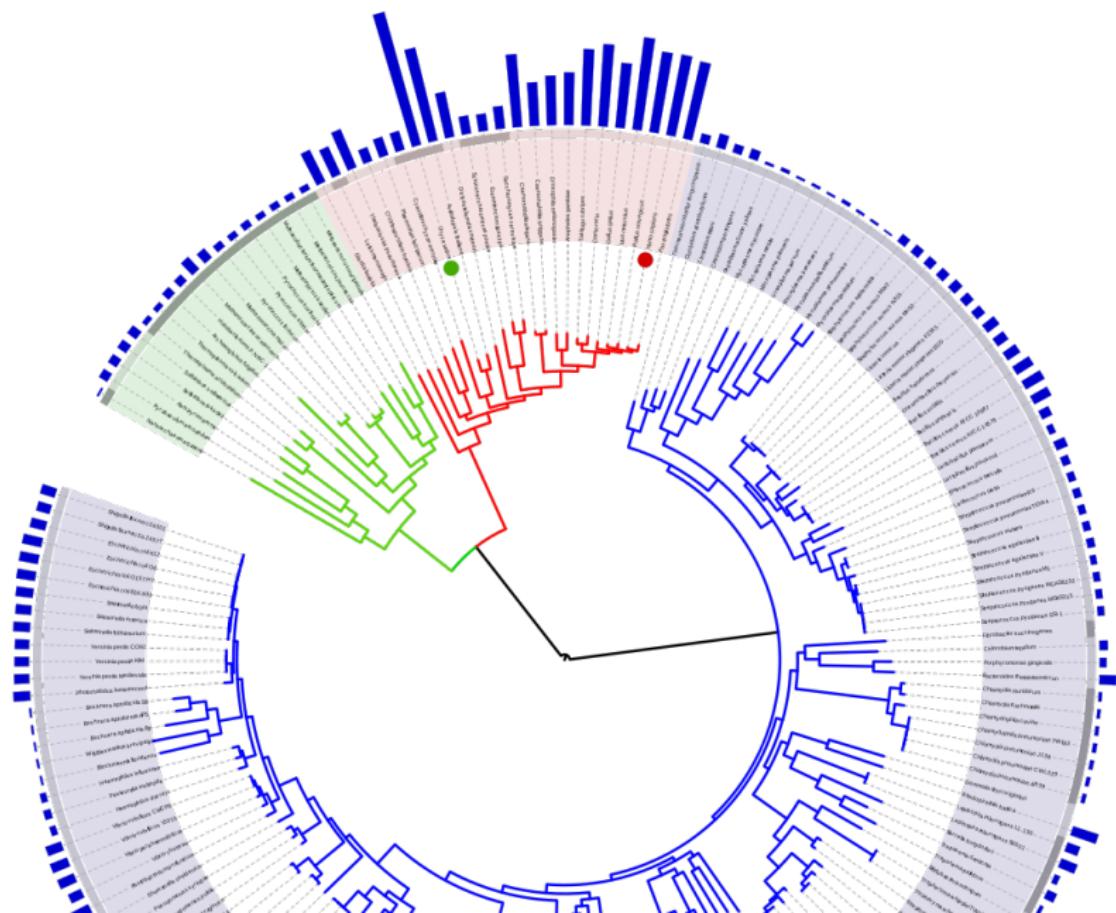
Agglomerative

1. начинаем с ситуации, когда каждый объект – отдельный кластер
2. на каждом шаге совмещаем два наиболее близких кластера
3. останавливаемся, когда получаем требуемое количество или единственный кластер

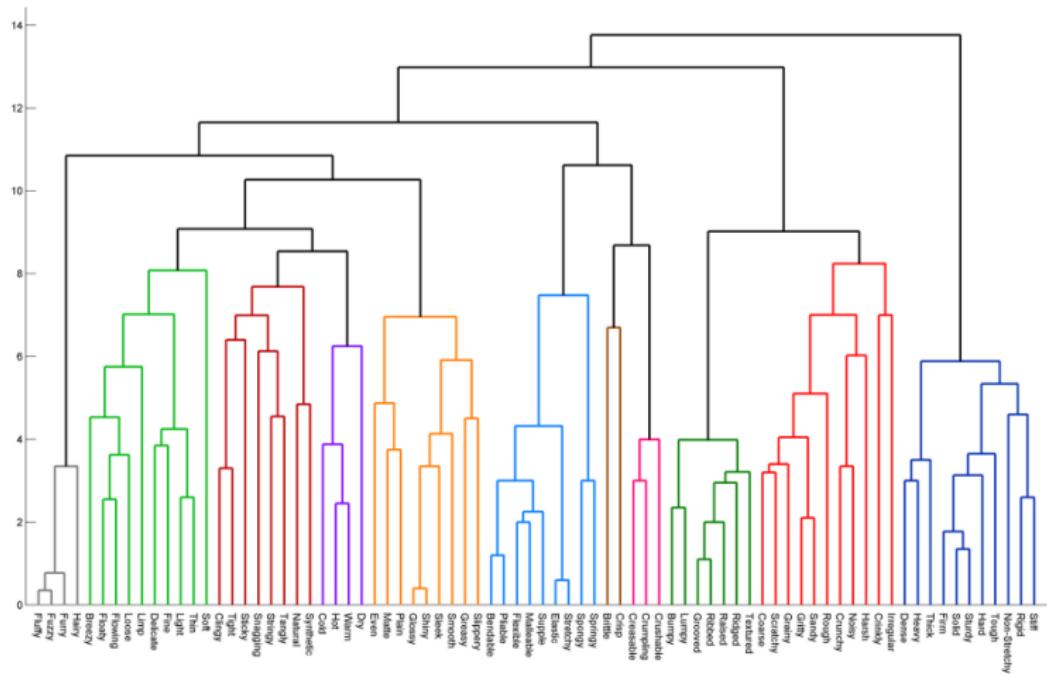
Divisive

1. начинаем с ситуации, когда все объекты составляют один кластер
2. на каждом шаге разделяем два один из кластеров пополам
3. останавливаемся, когда получаем требуемое количество или N кластеров

Радиальная дендрограмма (источник)



Дендрограмма



Агломеративный алгоритм

```
1 function agglomerative(X, K):
2     initialize N # number of objects
3     initialize C = N # number of clusters
4     initialize C_i = x_i # initial clusters
5     while C > K:
6         C_a = C_b = None # closest clusters
7         min_dist = +inf # distance between closest
8         for i in 1 .. C:
9             for j in i + 1 .. C:
10                dist = d(C_i, C_j) # dist. betw. clusters
11                if dist < min_dist:
12                    min_dist = dist
13                    C_a = C_i
14                    C_b = C_j
15                merge(C_a, C_b)
16                C = C - 1
17    return C_1, ..., C_K
```

память $O(N^2)$, сложность $O(N^2 \log N)$

Расстояние между кластерами

- ▶ single-linkage

$$d_{min}(C_i, C_j) = \min_{\mathbf{x} \in C_i, \mathbf{x}' \in C_j} \|\mathbf{x} - \mathbf{x}'\|$$

- ▶ complete-linkage

$$d_{max}(C_i, C_j) = \max_{\mathbf{x} \in C_i, \mathbf{x}' \in C_j} \|\mathbf{x} - \mathbf{x}'\|$$

- ▶ average

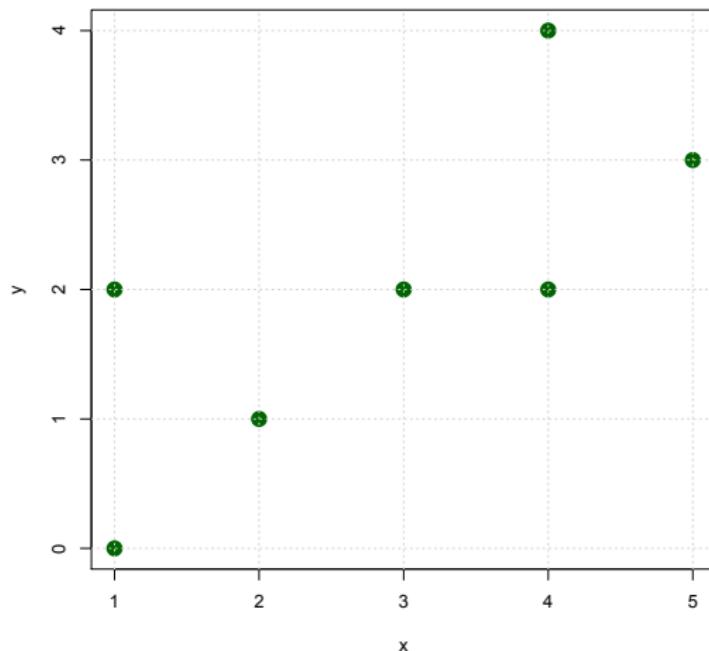
$$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x} \in C_i} \sum_{\mathbf{x}' \in C_j} \|\mathbf{x} - \mathbf{x}'\|$$

- ▶ mean

$$d_{mean}(C_i, C_j) = \|\mathbf{m}_i - \mathbf{m}_j\|$$

Задача

Кластеризовать данные иерархическим методом с использованием расстояний между кластерами d_{min} и d_{max}



Stepwise-optimal HC

Какой критерий мы оптимизируем?

```
1 function swo(X, K):
2     initialize N # number of objects
3     initialize C = N # number of clusters
4     initialize C_i = x_i # initial clusters
5     while C > K:
6         # choose the pair that optimizes
7         # the given criterion J when merged
8         C_a, C_b = find_best_merge(J, C_1, ..., C_C)
9         merge(C_a, C_b)
10        C = C - 1
11    return C_1, ..., C_K
```

d_{max} обеспечивает наименьшее увеличение диаметра кластера

d_e обеспечивает наименьшее увеличение квадратичного критерия

$$d_e(C_i, C_j) = \sqrt{\frac{n_i n_j}{n_i + n_j}} \|\mathbf{m}_i - \mathbf{m}_j\|$$

Неевклидовы пространства

Проблема. Как измерить расстояние между кластерами, если невозможно определить центроид?

Идея. В каждом из кластеров выбрать “типичный” пример – clustroid.

Минимизируем

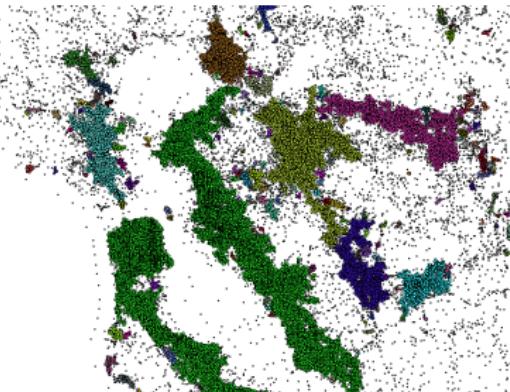
- ▶ сумму расстояний до других объектов в кластере
- ▶ сумму квадратов расстояний до других объектов в кластере
- ▶ максимальное расстояние до других объектов в кластере

Иерархическая кластеризация: итог

- + Несферические кластеры
- + Разнообразие критериев
- + Любые K из коробки
- Требует много ресурсов

DBSCAN: идея метода

- ▶ Кластеризация, основанная на плотности объектов
- ▶ Кластеры – участки высокой плотности, разделенные участками низкой плотности



(источник)

Определения

Плотность

Количество объектов внутри сферы заданного радиуса ε

Core-объект

Объект x является core-объектом, если плотность вокруг него больше min_pts

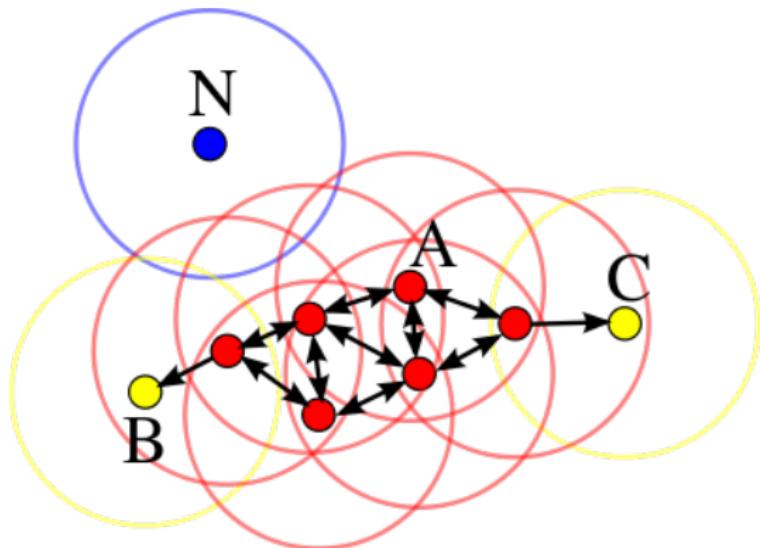
Граничный-объект

Объект x является граничным-объектом, если плотность вокруг него меньше min_pts , но он находится рядом с core-объектом

Шум

Объект x является шумом, если он не является ни core-объектом, ни граничным объектом

Виды объектов



DBSCAN 1

```
1 function dbscan(X, eps, min_pts):
2     initialize NV = X # not visited objects
3     for x in NV:
4         remove(NV, x) # mark as visited
5         nbr = neighbours(x, eps) # set of neighbours
6         if nbr.size < min_pts:
7             mark_as_noise(x)
8         else:
9             C = new_cluster()
10            expand_cluster(x, nbr, C, eps, min_pts, NV)
11            yield C
```

DBSCAN 2

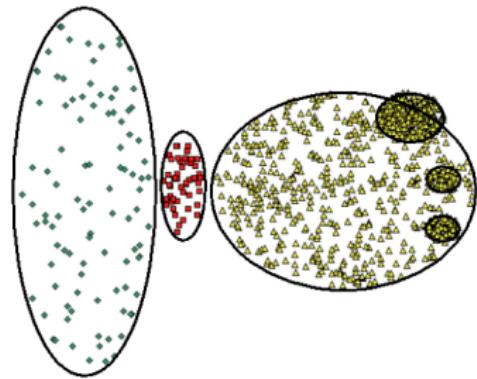
```
1 function expand_cluster(x, nbr, C, eps, min_pts, NV):
2     add(x, C)
3     for x1 in nbr:
4         if x1 in NV: # object not visited
5             remove(NV, x1) # mark as visited
6             nbr1 = neighbours(x1, eps)
7             if nbr1.size >= min_pts:
8                 # join sets of neighbours
9                 merge(nbr, nbr_1)
10            if x1 not in any cluster:
11                add(x1, C)
```

Сложность: $O(n^2)$ или $O(n \log n)$ (R^* Tree)

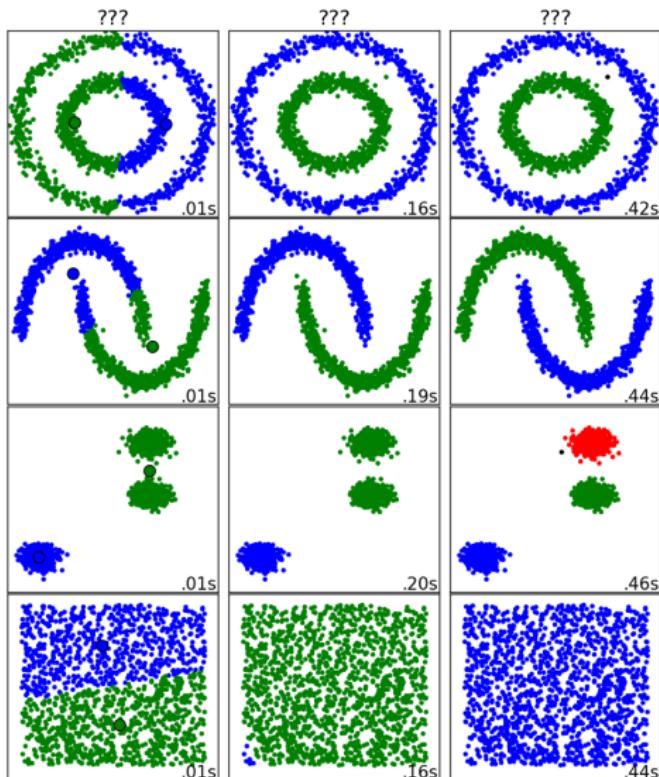
Память: $O(n)$ или $O(n^2)$

DBSCAN: итог

- + не требует K
- + кластеры произвольной формы
- + учитывает выбросы
- Не вполне детерминированный
- Не работает при разных плотностях кластеров



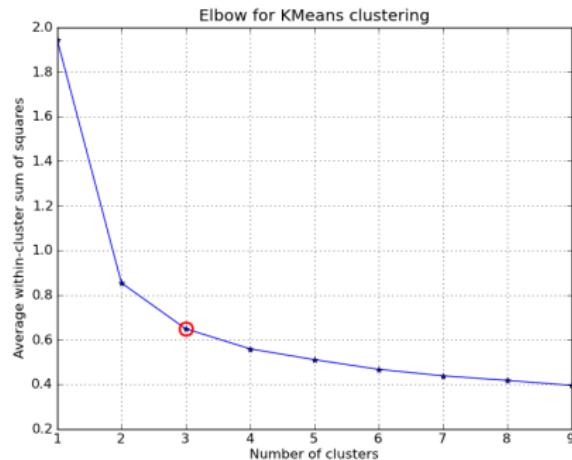
Викторина



Выбор наилучшего K

Идея. Выбрать критерий качества кластеризации и построить его значение для $K = 1, 2, \dots$

- ▶ средняя сумма квадратов расстояния до центроида
- ▶ средний диаметр кластера



Критерий Silhouette

Пусть дана кластеризация в K кластеров, и объект i попал в C_k

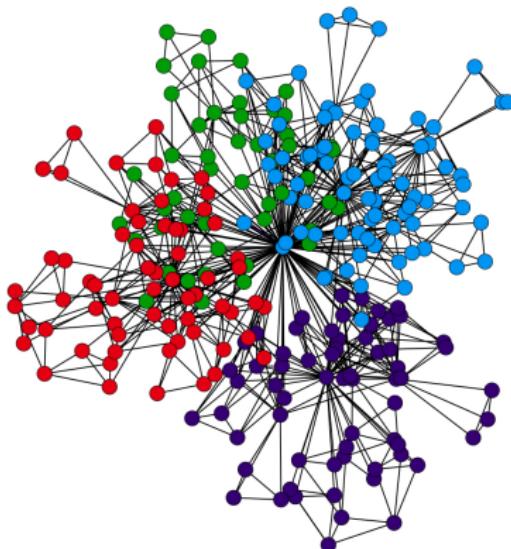
- ▶ $a(i)$ – среднее расстояние от i объекта до объектов из C_k
- ▶ $b(i) = \min_{j \neq k} b_j(i)$, где $b_j(i)$ – среднее расстояние от i объекта до объектов из C_j

$$\text{silhouette}(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

Средний silhouette для всех точек из \mathbf{X} является критерием качества кластеризации.

Кластеризация. Что дальше

- ▶ Графо-теоретические методы (affinity propagation, mcl, etc.)
- ▶ Снижение размерности (SOM, PCA, etc.)
- ▶ Бесконечномерные смеси (Dirichlet process)



Задача

Кластеризовать фотографии с футбольных матчей. Ожидается что фотографии игроков одной команды окажутся в одном кластере.

- ▶ Скачать данные и шаблон кода <http://bit.ly/1uALLl4>
- ▶ Запустить код и интерпретировать дендрограмму

```
python football/football.py football/img/
```

При $K = 3$ сколько ошибок дает алгоритм?

- ▶ Поэкспериментировать с разными видами расстояния между кластерами (-l). Какой из них дает наилучший результат?
- ▶ Реализовать критерии выбора количества кластеров: diameter и silhouette. Какое число кластеров рекомендует каждый из критериев?

Вопросы

