$O(|\mathcal{D}|^2 \log|\mathcal{D}|)$; усреднение по группе – $O(|\mathcal{D}|^2)$. Сложность дивизимного алгоритма зависит от выбранного дополнительного алгоритма плоской кластеризации, если выбран алгоритм k-средних, то – $O(|\mathcal{D}|)$.

§ 2.2. Алгоритм k-средних

Первые применения алгоритма k-средних были описаны в работе Джеймса МакКуина в 1967 году. При заранее известном числе кластеров k алгоритм k-средних начинает с некоторого начального разбиения документов и уточняет его, оптимизируя целевую функцию – среднеквадратичную ошибку кластеризации как среднеквадратичное расстояние между документами и центрами их кластеров:

$$e(\mathcal{D}, \mathcal{C}) = \sum_{j=1}^{k} \sum_{i: d_i \in C_j} \left\| \vec{d}_i - \vec{\mu}_j \right\|^2, \quad (42)$$

где μ_j – центр, или центроид, кластера $C_j, j = \overline{1, |\mathcal{C}|}$, $|\mathcal{C}| = k$, вычисляющийся по формуле:

$$\vec{\mu}_j = \frac{1}{|C_j|} \sum_{i:\, d_i \in C_j} \overrightarrow{d_i}, \quad (43)$$

где $|C_j|$ – количество документов в C_j.

Идеальным кластером алгоритм k-средних считает сферу с центроидом в центре сферы.

Действие алгоритма начинается с выбора k начальных центров кластеров. Обычно исходные центры кластеров выбираются случайным образом. Затем каждый документ присваивается тому кластеру, чей центр является наиболее близким документу, и выполняется повторное вычисление центра каждого кластера как центроида, или среднего своих членов. Такое перемещение документов и повторное вычисление центроидов кластеров продолжается до тех пор, пока не будет достигнуто условие остановки. Условием остановки может служить следующее: (а) достигнуто пороговое число итераций, (б) центроиды кластеров больше не изменяются и (в) достигнуто пороговое значение ошибки кластеризации. На практике используют комбинацию критериев остановки, чтобы одновременно ограничить время работы алгоритма и получить приемлемое качество.

В общем случае алгоритм k-средних достигает локального минимума целевой функции, что приводит к субоптимальному разбиению документов. Поэтому важен способ выбора начальных значений центроидов. Для этого известны различные эвристические правила, например, получить начальные центры с помощью другого алгоритма – детерминированного, например, иерархического агломеративного.

Алгоритм в общем виде.

Вход: множество проиндексированных документов $\boldsymbol{\mathcal{D}}$, количество кластеров k.

Шаг 1. Инициализация центров кластеров $\{\vec{\mu}_j\}, j = \overline{1, k}$, например, случайными числами.

Шаг 2. $C_j \coloneqq \{\}, j = \overline{1, k}$.

Шаг 3. Для каждого $d_i \in \boldsymbol{\mathcal{D}}$:

Шаг 4. $j^* \coloneqq arg\ \min_j \left\| \vec{\mu}_j - \vec{d}_i \right\|, j = \overline{1, k}$;

Шаг 5. $C_{j^*} \coloneqq C_{j^*} + \{d_i\}$;

Шаг 6. Для каждого $C_j \in \boldsymbol{\mathcal{C}}$:

Шаг 7. $\vec{\mu}_j := \frac{1}{|C_j|}\sum_{i:\, d_i \in C_j} \overrightarrow{d_i}$,

Шаг 8. Если не достигнуто условие остановки, то повторить с *шага 2*.

Выход: множество центров кластеров $\{\vec{\mu}_j\}$ и множество самих кластеров $\boldsymbol{\mathcal{C}}$.

Пример. Пусть k=2.

Итерация 1. Начальные $\{\vec{\mu}_j\}, j = \overline{1,k}$ инициализированы случайным образом:

$$\mu_1 = [0.96\ \ 0.80\ \ 0.42\ \ 0.79\ \ 0.66\ \ 0.85];$$
$$\mu_2 = [0.49\ \ 0.14\ \ 0.91\ \ 0.96\ \ 0.04\ \ 0.93]$$

dist	**d1**	**d2**	**d3**	**d4**	**d5**	**d6**
$\boldsymbol{\mu_1}$	1.55	1.81	1.66	1.51	1.38	0.85
$\boldsymbol{\mu_2}$	1.82	1.38	1.37	1.74	1.59	0,93

⇨ C1 := {d1, d4, d5, d6}; C2 := {d2, d3}.

Итерация 2.

$$\mu_1 = [0.24\ \ 0.45\ \ 0\ \ 0\ \ 0.43\ \ 0.35];$$
$$\mu_2 = [0.159\ \ 0\ \ 0.49\ \ 0.49\ \ 0\ \ 0]$$

dist	**d1**	**d2**	**d3**	**d4**	**d5**	**d6**
$\boldsymbol{\mu_1}$	0.74	1.21	1.22	0.68	0.61	0.67
$\boldsymbol{\mu_2}$	1.18	0.69	0.69	1.21	1.18	1.24

⇨ C1 := {d1, d4, d5, d6}; C2 := {d2, d3}.

Разбиение на кластеры не изменилось – условие остановки выполнено, мы получили итоговые два кластера.

Вычислительная сложность. Алгоритм k-средних линейно зависит от всех своих факторов: от количества документов, количества кластеров, количества терминов и количества итераций. Для сохранения линейной сложности ($O(|\mathcal{D}|)$) при комбинировании с иерархическим с целью эффективного задания начальных центроидов кластеров, предлагается квадратичный иерархический алгоритм применить к выборке документов размером $\sqrt{|\mathcal{D}|}$. Такой подход получил название алгоритм картечи.

§ 2.3. Плотностный алгоритм DBSCAN

Алгоритм DBSCAN (Density Based Spatial Clustering of Applications with Noise), плотностный алгоритм для кластеризации пространственных данных с присутствием шума), был предложен Мартином Эстер, Гансом-Питером Кригель и коллегами в 1996 году как решение проблемы разбиения (изначально пространственных) данных на кластеры произвольной формы [4]. Большинство алгоритмов, производящих плоское разбиение, создают кластеры по форме близкие к сферическим, так как минимизируют расстояние документов до центра кластера.

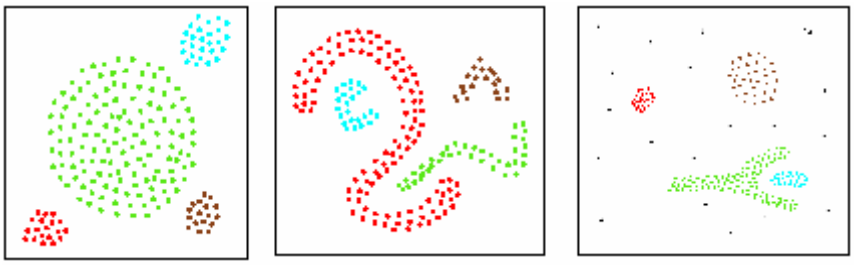

Рис. 8. Примеры кластеров произвольной формы

Авторы DBSCAN экспериментально показали, что их алгоритм способен распознать кластеры различной формы, например, как на рис. 8.

Идея, положенная в основу алгоритма, заключается в том, что внутри каждого кластера наблюдается типичная плотность точек (объектов), которая заметно выше, чем плотность снаружи кластера, а также плотность в областях с шумом ниже плотности любого из кластеров. Ещё точнее, что для каждой точки кластера её соседство заданного радиуса должно содержать не менее некоторого числа точек, это число точек задаётся пороговым значением. Перед изложением алгоритма дадим необходимые определения.

Определение 1. Eps-соседство точки p, обозначаемое как $N_{Eps}(p)$, определяется как множество документов, находящихся от точки p на расстояния не более *Eps*: $N_{Eps}(p) = \{q \in \mathcal{D} | \, dist(p,q) \leq Eps\}$. Поиска точек, чьё $N_{Eps}(p)$содержит хотя бы минимальное число точек (*MinPt*) не достаточно, так как точки бывают двух видов: ядровые и граничные.

Определение 2. Точка p *непосредственно плотно-достижима* из точки q (при заданных *Eps* и *MinPt*), если $p \in N_{Eps}(p)$ и $|N_{Eps}(p)| \geq MinPt$ (рис. 9).

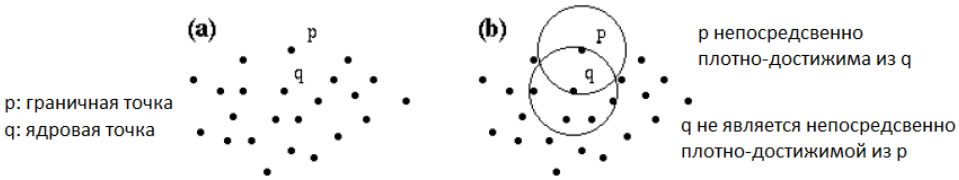

Рис. 9. Пример точек, находящихся в отношении *непосредственно плотной достижимости*

Определение 3. Точка p *плотно-достижима* из точки q (при заданных *Eps* и *MinPt*), если существует последовательность точек $q = p_1, p_2 \dots, p_n = p: p_{i+1}$ непосредственно плотно-достижимы из p_i. Это отношение транзитивно, но не симметрично в общем случае, однако симметрично для двух ядровых точек.

Определение 4. Точка p *плотно-связана* с точкой q (при заданных *Eps* и *MinPt*), если существует точка o: p и q плотно-достижимы из o (при заданных *Eps* и *MinPt*), см. рис. 10.

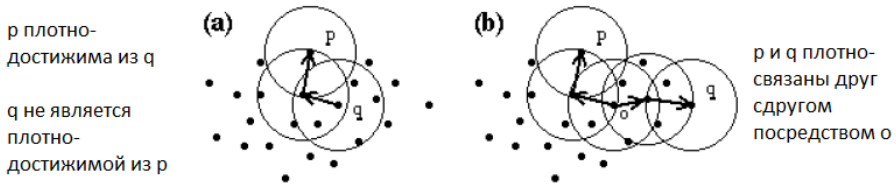

Рис. 10. Пример точек, находящихся в отношении *плотной связанности*

Теперь мы готовы дать определения кластеру и шуму.

Определение 5. Кластер C_j(при заданных *Eps* и *MinPt*) – это не пустое подмножество документов, удовлетворяющее следующих условиям:

1) $\forall p, q$: если $p \in C_j$ и q плотно-достижима из p (при заданных *Eps* и *MinPt*), то $q \in C_j$.
2) $\forall p, q \in C_j$: p плотно-связана с q (при заданных *Eps* и *MinPt*).

Итак, кластер – это множество плотно-связанных точек. В каждом кластере содержится хотя бы *MinPt* документов.

Шум – это подмножество документов, которые не принадлежат ни одному кластеру: $\{p \in \mathcal{D} | \forall j: p \notin C_j, j = \overline{1, |\mathcal{C}|}\}$.

Алгоритм DBSCAN для заданных значений параметров *Eps* и *MinPt* исследует кластер следующим образом: сначала выбирает случайную точку, являющуюся ядровой, в качестве затравки, затем помещает в кластер саму затравку и все точки, плотно-достижимые из неё.

Алгоритм в общем виде.

DBSCAN

Вход: множество индексированных документов $\boldsymbol{\mathcal{D}}$, *Eps* и *MinPt*.

Шаг 1. Установить всем элементам множества $\boldsymbol{\mathcal{D}}$ флаг «не посещён». Присвоить текущему кластеру C_j нулевой номер, j := 0. Множество шумовых документов Noise := $\emptyset$.

Шаг 2. Для каждого $d_i \in \boldsymbol{\mathcal{D}}$ такого, что флаг(d_i) = «не посещён», выполнить:

Шаг 3. флаг(d_i) := «посещён»;

Шаг 4. $N_i := N_{Eps}(d_i) = \{q \in \mathcal{D} | \, dist(d_i, q) \leq Eps\}$

Шаг 5. Если $|N_i| <$ *MinPt*, то

Noise := Noise + $\{d_i\}$

иначе

номер следующего кластера j := j + 1;

EXPANDCLUSTER($d_i, N_i, C_j, Eps, MinPt$);

Выход: множество кластеров $\boldsymbol{\mathcal{C}}$=$\{C_j\}$.

EXPANDCLUSTER

Вход: текущий документ d_i, его eps-соседство N_i, текущий кластер C_j и $Eps, MinPt$.

Шаг 1. $C_j := C_j + \{d_i\}$;

Шаг 2. Для всех документов $d_k \in N_i$:

Шаг 3. Если флаг(d_k) = «не посещён», то

Шаг 4 флаг(d_k) := «посещён»;

Шаг 5. $N_{ik} := N_{Eps}(d_k)$;

Шаг 6. Если $|N_{ik}| >=$ *MinPt*, то $N_i := N_i + N_{ik}$;

Шаг 7. Если $\nexists p$: $d_k \in C_p$, $p = \overline{1, |\mathcal{C}|}$, то $C_j := C_j + \{d_k\}$;

Выход: кластер C_j.

Эвристический подход к заданию начальных параметров. Для некоторого значения k, построить график, на котором каждому документу коллекции поставить в соответствие значение k-dist – расстояние до его k-ого соседа, при этом документы должны быть отсортированы по убыванию значения k-dist. Тогда полученный график может породить догадки о распределении плотности в массиве документов. Ищем пороговую точку p на графике с наибольшим значением k-dist для «самого разреженного» кластера: предполагаем, то это первая точка первой «равнины».

Точки, расположенные левее, предположительно считаются шумовыми, а правее – принадлежащими одному из кластеров. Тогда значения параметров *Eps = k-dist(p)* и *MinPt=k*.

Пример. Воспользуемся эвристикой авторов DBSCAN для настройки начальных параметров, построим 3-dist: (d2;1,43), (d3;1,39), (d1;1,36), (d4;1,36), (d5;1,32), (d6;1,30).

Рис. 11. Поиск значений входных параметров алгоритма DBSCAN

Итак, *Eps* = 1,36 и *MinPt* = *3*.

j := 0; Noise := ∅; Флаг := [0 0 0 0 0 0], где 0 соответствует «не посещён».

1) d_i := **d1**. Флаг = [1 0 0 0 0 0]. $N_i = \{d1, d2, d4, d5, d6\}$. $|N_i| = 5 > MinPt$; j = 1;

2) C_j = {d1}.

3) d_k := **d2**; Флаг = [1 1 0 0 0 0] N_{ik} = {d1, d2, d5}; $|N_{ik}| = 3 >= MinPt$; $N_i = \{d1, d2, d4, d5, d6\}$. C_j = {d1, d2}.

4) d_k := **d4**; Флаг = [1 1 0 1 0 0] N_{ik} = {d1, d4,d5,d6}; $|N_{ik}| = 4 >= MinPt$; $N_i = \{d1, d2, d4, d5, d6\}$. C_j = {d1, d2, d4}.

5) d_k := **d5**; Флаг = [1 1 0 1 1 0] N_{ik} = {d1, d2, d4, d5,d6}; $|N_{ik}| = 5 >= MinPt$; $N_i = \{d1, d2, d4, d5, d6\}$. C_j = {d1, d2, d4, d5}.

6) d_k := **d6**; Флаг = [1 1 0 1 1 1] N_{ik} = {d1, d4, d5,d6}; $|N_{ik}| = 4 >= MinPt$; $N_i = \{d1, d2, d4, d5, d6\}$. C_j = {d1, d2, d4, d5, d6}.

7) d_i := **d3**. Флаг = [1 1 1 1 1 1]. $N_i = \{d3\}$. $|N_i| = 1 < MinPt$; Noise := {d3}.

Получаем один кластер и шумовые документы:

$\mathcal{C}1$ = {d1, d2, d4, d5, d6} и Noise = {d3}. Разницу между ожиданиями, связанными с графиком 3-dist (2 шумовых документа), можно объяснить особенностью построения графика 3-dist для данного частного примера: при вычислении расстояния до k-ого соседа k-соседом становился тот, расстояние до которого отлично от расстояния до (k-1)-соседа.

Вычислительная сложность. В общем случае алгоритм DBSCAN имеет квадратичную вычислительную сложность из-за поиска *Eps*-соседства. Однако авторы алгоритма использовали для этой цели специальную структуру данных – R*-деревья, в результате поиск *Eps*-соседства для одной точки – O(log n). Общая вычислительная сложность DBSCAN – O(n * log n).

§ 2.4. Нечёткий алгоритм c-средних

Нечёткий алгоритм c-средних был предложен Джоном С. Данном в 1973 году (позднее усовершенствован Дж. Бездеком в 1981 году) как решение проблемы мягкой кластеризации, то есть присвоения каждого документа более чем одному кластеру. Как и его чёткий вариант – алгоритм k-средних – данный алгоритм, начиная с некоторого начального разбиения данных, итеративно минимизирует целевую функцию, которой является следующее выражение:

$$e_m(\mathcal{D}, \mathcal{C}) = \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|\mathcal{C}|} u_{ij}^m \left\| \vec{d}_i - \vec{\mu}_j \right\|^2, \quad (44)$$

где m – спепень нечёткости, $1 < m < \infty$;

u_{ij} – степень принадлежности i-ого документа j-ому кластеру,