ЛЕКЦИЯ 3. Интеллектуальный анализ данных: задача кластеризации

Кластеризация – одна из задач Data Mining, а кластер – группа похожих объектов. Существует много определений кластеризации, поэтому приведем несколько из них.

Кластеризация – группировка объектов на основе близости их свойств; каждый кластер состоит из схожих объектов. а объекты разных кластеров существенно отличаются;

Кластеризация – процедура. которая любому объекту $x \in X$ *ставит в соответствие метку кластера* $y \in Y$.

Кластеризацию используют, когда отсутствуют априорные сведения относительно классов, к которым можно отнести объекты исследуемого набора данных, либо кода число объектов велико, что затрудняет их ручной анализ.

Постановка задачи кластеризации сложна и неоднозначна, так как:

- оптимальное количество кластеров в общем случае неизвестно;
- выбор меры «похожести» или близости свойств объектов между собой, как и критерия качества кластеризации, часто носит субъективный характер.

Объекты внутри кластера должны быть похожими друг на друга и отличаться от других, которые вошли в другие кластеры. В задачах кластеризации не требуется указание выходной переменной, т.е. имени кластера, а число кластеров, в которые необходимо сгруппировать все множество данных, может быть неизвестным. Выходом кластеризации является не готовый ответ (например, «плохо» / «удовлетворительно» / «хорошо»), а группы похожих объектов — кластеры. Кластеризация указывает только на схожесть объектов, и не более того. Для объяснения образовавшихся кластеров необходима их дополнительная интерпретация. Кластеризация может использоваться, например, для сегментации и построения профилей клиентов банка, телекоммуникационной или страховой кампаний. Так, в задаче определения групп клиентов при достаточно большом их числе становится трудно подходить к каждому индивидуально, поэтому их удобно объединять в группы — сегменты с однородными признаками. Выделять сегменты можно по нескольким группам признаков, например, по сфере деятельности, географическому расположению, статусу и т.п. После кластеризации можно узнать, какие сегменты наиболее активны, какие приносят наибольшую прибыль, выделить характерные для них признаки. Эффективность работы с клиентами повышается благодаря учету их персональных предпочтений. Задача кластеризации известна давно, и специалисты в различных областях знаний оперируют рядом других терминов

— таксономия, сегментация, группировка, автоматическая классификация и др. В *Data Mining* используется термин «кластеризация». Например, в аналитике кластеризация применяется для решения следующих задач.

Изучение данных. Разбиение множества объектов на схожие группы помогает выявить структуру данных, увеличить наглядность их представления, выдвинуть новые гипотезы, понять, насколько информативны свойства объектов.

Облегчение анализа. При помощи кластеризации можно упростить дальнейшую обработку данных и построение моделей. Каждый кластер обрабатывается индивидуально, и модель создается для каждого кластера индивидуально. В этом смысле кластеризация является подготовительным этапом перед решением других задач *Data Mining*.

Сжатие данных. В случае, когда данные имеют большой объем (сотни тысяч и миллионы строк), кластеризация позволяет сократить объем хранимых данных, оставив по одному наиболее типичному представителю от каждого кластера.

Прогнозирование. Кластеры используются не только для краткого описания имеющихся объектов, но и для распознавания новых. Каждый новый объект относится к тому кластеру, присоединение к которому наилучшим образом удовлетворяет критерию качества кластеризации. Далее можно прогнозировать поведение объекта, предположив, что оно будет схожим с поведением других объектов кластера.

Обнаружение аномалий. Кластеризация применяется для выделения нетипичных объектов, которые не присоединяются ни к одному из кластеров.

Алгоритм k-means

Сегодня предложено несколько десятков алгоритмов кластеризации и еще больше их разновидностей. Несмотря на это, в *Data Mining* применяются в первую очередь понятные и простые в использовании алгоритмы. К таким относится алгоритм *k-means* — в русскоязычном варианте *k*-средних (от англ. *mean* — «среднее значение»). Его основная идея состоит в том, что для выборки данных, содержащей n записей (объектов), задается число кластеров — k, которое должно быть сформировано. Затем алгоритм разбивает все объекты выборки на k разделов ($k < n$), которые и представляют собой кластеры.

Алгоритм выполняется в четыре шага:

1) задается число кластеров — k, которое должно быть сформировано из объектов исходной выборки;
2) случайным образом выбирается k записей исходной выборки, которые

будут служить начальными центрами кластеров. Начальные точки, из которых потом вырастает кластер, часто называют «семенами» (от англ. *seeds* — «семена», «посевы»). Каждая такая запись представляет собой своего рода «эмбрион» кластера, состоящий только из одного элемента;

3) для каждой записи исходной выборки определяется ближайший к ней центр кластера. Чтобы определить, в сферу влияния какого центра кластера входит та или иная запись, вычисляется расстояние от каждой записи до каждого центра в многомерном пространстве признаков и выбирается то «семя», для которого данное расстояние минимальное;

4) в анализе данных распространенной оценкой близости между объектами является *метрика*, или способ задания расстояния. Выбор конкретной метрики зависит от аналитика и конкретной задачи. Наиболее популярные метрики — евклидово расстояние и расстояние Манхэттена.

Евклидово расстояние, или метрика L_2, применяется для вычисления расстояний следующее правило по формуле:

$$d_E(X,Y) = \sqrt{\sum_i (x_i - y_i)^2},$$

где $X = (x_1, x_2, \ldots, x_m)$, $Y = (y_1, y_2, \ldots, y_m)$ — векторы значений признаков двух записей.

Поскольку множество точек, равноудаленных от некоторого центра, при использовании евклидовой метрики будут образовывать *сферу* (или круг в двумерном случае), то кластеры, полученные с использованием евклидова расстояния, также будут иметь форму, близкую к сферической.

Расстояние Манхеттена, или метрика L_1, вычисляется по формуле:

$$d_M(X,Y) = \sum_i |x_i - y_i|.$$

Фактически *расстояние Манхеттена* — кратчайшее расстояние между двумя точками, пройденное по линиям, параллельным осям координатой системы. Преимущество метрики L_1 заключается в том, что она позволяет снизить влияние аномальных значений на работу алгоритмов. Кластеры, построенные на основе расстояния Манхеттена, стремятся к *кубической* форме.

Используя метрики L_1 или L_2, для каждой записи исходной выборки определяется ближайший к ней центр (*центроид*) кластера.

Например, если в кластер вошли три записи с наборами признаков (x_1, y_1), (x_2, y_2), (x_3, y_3), то координаты его центроида по метрике L_1 будут рассчитываться следующим образом:

$$(x, y) = \left(\frac{(x_1 + x_2 + x_3)}{3}, \frac{(y_1 + y_2 + y_3)}{3} \right);$$

5) старый центр кластера смещается в его центроид.

Таким образом, центроиды становятся новыми центрами кластеров для следующего итерации алгоритма. Шаги 3 и 4 повторяются до тех пор, пока выполнение алгоритма не будет прервано или пока не будет выполнено условие в соответствии с некоторым *критерием сходимости*.

Остановка алгоритма производится, когда границы кластеров и расположение центроидов перестают изменяться от итерации к итерации, т.е. на каждой итерации в каждом кластере остается один и тот же набор записей. Алгоритм *k-means* обычно позволяет находить набор стабильных кластеров за несколько десятков итераций.

Что касается *критерия сходимости*, то чаще всего используется сумма квадратов ошибок между центроидом кластера и всеми вошедшими в него записями:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \left(p - m_i \right)^2,$$

где $p \in C_i$ — произвольная точка данных, принадлежащих кластеру C_i; m_i — центроид данного кластера.

Иными словами, алгоритм остановится тогда, когда ошибка E достигнет достаточно малого.

Один из основных недостатков, присущих алгоритму *k-means*, — отсутствие четких критериев выбора числа кластеров, целевой функции их инициализации и модификации. Кроме того, он очень чувствителен к шумам в данных и аномальным значениям, поскольку они способны существенно повлиять на среднее значение, используемое при вычислении положений центроидов. Чтобы снизить влияние таких факторов, как шумы и аномальные значения, иногда на каждой итерации используют не среднее значение признаков, а их медиану. Данная модификация алгоритма называется *k-mediods* (*k*-медиан).

Пример работы алгоритма k-means

Пусть имеется набор из восьми точек данных в двумерном пространстве, из которого требуется получить два кластера. Значения точек приведены в табл. 5.1 и на рис. 5.4.

Таблица 5.1

Объекты для кластеризации

A	*B*	*C*	*D*	*E*	*F*	*G*	*H*
(1;3)	(3;3)	(4;3)	(5;3)	(1;2)	(4; 2)	(1;1)	(2;1)

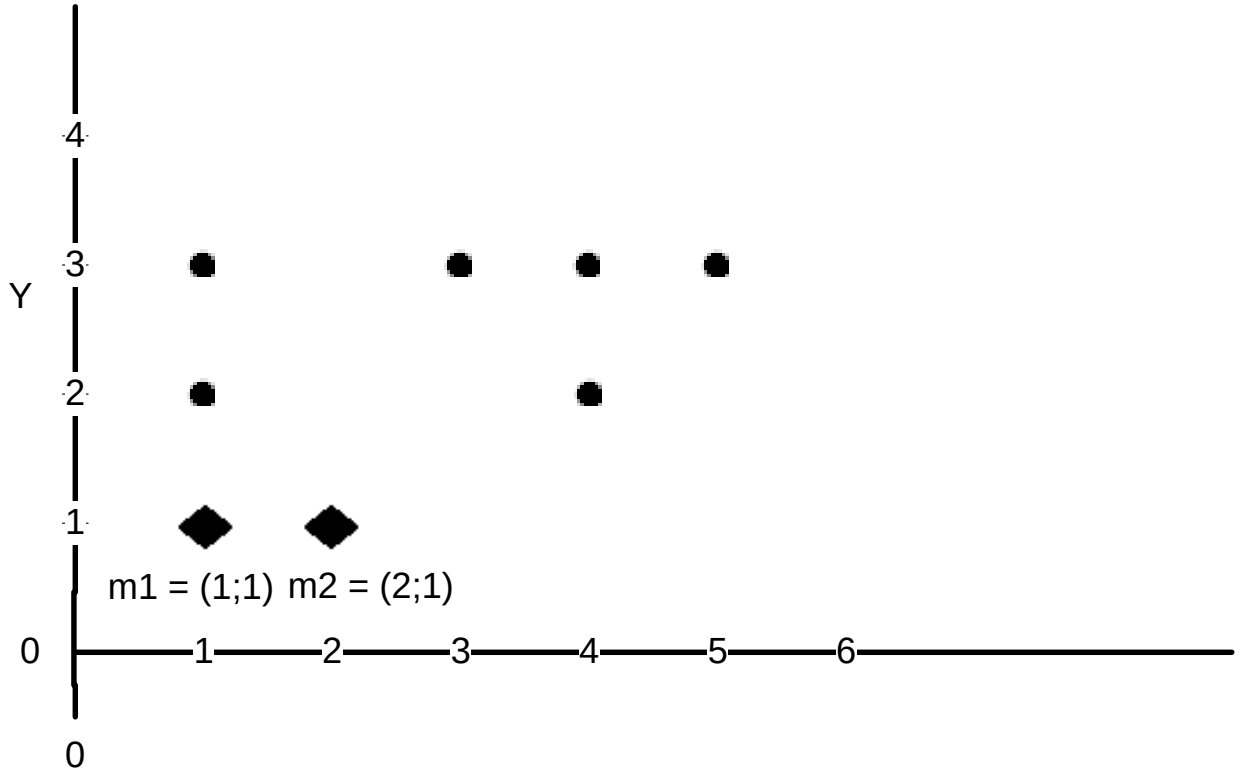

Рис. 5.1. Процедура начальной инициализации: значения буллитов приведены в табл. 5.1

Шаг 1. Определим число кластеров, на которое требуется разбить исходное множество: $k = 2$.

Шаг 2. Случайным образом выберем две точки, которые будут начальными центрами кластеров. Пусть это будут точки $m_1 = (1;1)$ и $m_2 = (2:1)$. На рис. 5.1 они представлены ромбами.

Шаг 3, проход 1. Для каждой точки определим ближайший к ней центр кластера с помощью евклидова расстояния. В табл. 5.2 представлены вычисленные с помощью формулы $d_E(X,Y) = \sqrt{\sum_i (x_i - y_i)^2}$ расстояния между центрами кластеров $m_1 = (1;1)$ и $m_2 = (2:1)$ и каждой точкой исходного множества и указано, к какому кластеру принадлежит та или иная точка (табл. 5.2).

Таблица 5.2

Нахождение ближайшего центра для каждой точки (первый проход)

Точка	Расстояние от m_1	Расстояние от m_2	Принадлежит кластеру
A	2,00	2,24	1
B	2,83	2,24	2
C	3,61	2,83	2
D	4,47	3,61	2
E	1,00	1,41	1
F	3,16	2,24	2
G	0,00	1,00	1
H	1,00	0,00	2

Таким образом, кластер 1 содержит точки *A*, *E*, *G*, а кластер 2 — точки *B*, *C*, *D*, *F*, *H*. Как только определятся члены кластеров, может быть рассчитана сумма квадратов ошибок:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \left(p - m_i \right)^2 = 2^2 + 2,24^2 + 2,83^2 + 3,61^2 + 1^2 + 2,24^2 + 0^2 + 0^2 = 36 .$$

Шаг 4, проход 1. Для каждого кластера вычисляется центроид, и в него перемещается центр кластера.

Центроид для кластера 1: [(1 + 1 + 1) / 3, (3 + 2 + 1) / 3] = (1; 2).

Центроид для кластера 2: [(3 + 4 + 5 + 4 + 2) / 5, (3 + 3 + 3 + 2 + 1) / 5] = (3,6; 2,4).

Расположение кластеров и центроидов после первого прохода алгоритма представлено на рис. 5.2.

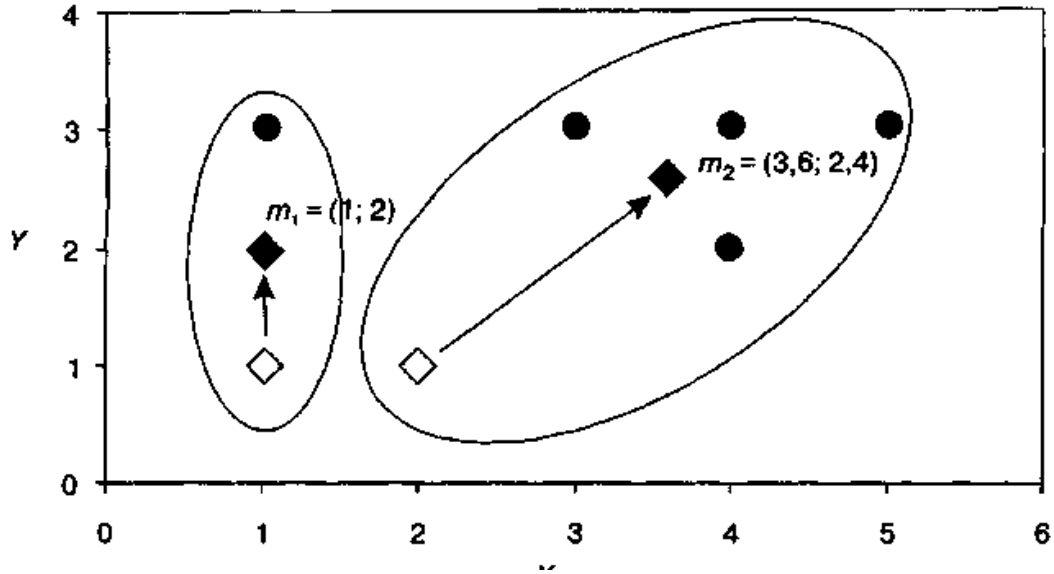

Рис. 5.2. Расположение кластеров и центроидов после первого прохода алгоритма

Здесь начальные центры кластеров представлены светлыми ромбами, а центроиды, вычисленные при первом проходе алгоритма, — темными ромбами. Они и станут новыми центрами кластеров, принадлежность точек данных к которым будет определяться на втором проходе.

Шаг 3, проход 2. После того, как найдены новые центры кластеров, для каждой точки снова определяется ближайший к ней центр и ее отношение к соответствующему кластеру. Для этого еще раз вычисляются евклидовы расстояния между точками и центрами кластеров. Результаты вычислений приведены в табл. 5.3.

Таблица 5.3

Нахождение ближайшего центра для каждой точки (второй проход)

Точка	Расстояние от m_1	Расстояние от m_2	Принадлежит кластеру
A	1,00	2,67	1
B	2,24	0,85	2
C	3,16	0,72	2
D	4,12	1,52	2
E	0,00	2,63	1
F	3,00	0,57	2
G	1,00	2,95	1
H	1,41	2,13	1

Относительно большое изменение m_2 привело к тому, что запись *H* оказалась ближе к центру m_1, что автоматически сделало ее членом кластера 1. Все остальные записи остались в тех же кластерах, что и на предыдущем проходе алгоритма. Таким образом, кластер 1 будет содержать точки *A*, *E*, *G*, *H*, а кластер 2 — *B*, *C*, *D*, *F*. Новая сумма квадратов ошибок составит:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \left(p - m_i \right)^2 = 1^2 + 0,85^2 + 0,72^2 + 1,52^2 + 0^2 + 0,57^2 + 1^2 + 1,41^2 = 7,88$$

Вычисление показывает уменьшение ошибки в сравнении с начальным состоянием центров кластеров (на первом проходе она составляла 36). Это говорит об улучшении качества кластеризации, т.е. о более высокой «кучности» объектов относительно центра кластера.

Шаг 4, проход 2. Для каждого кластера вновь вычисляется центроид, и в него перемещается центр кластера.

Новый центроид для кластера 1: [(1 + 1 + 1 + 2) / 4, (3 + 2 + 1 + 1) / 4] = (1,25; 1,75).

Новый центроид для кластера 2: [(3 + 4 + 5 + 4) /4, (3 + 3 + 3 + 2) / 4] = (4; 2,75).

Расположение кластеров и центроидов после второго прохода алгоритма представлено на рис. 5.3.

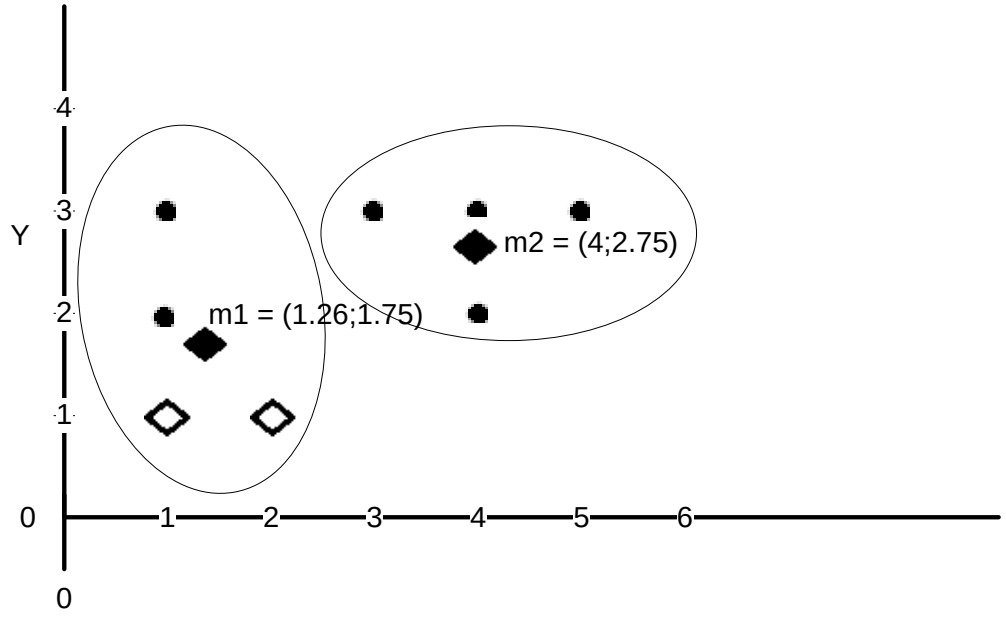

Рис. 5.3. Расположение кластеров и центроидов после второго прохода алгоритма

По сравнению с предыдущим проходом центры кластеров изменились незначительно.

Шаг 3, проход 3. Для каждой записи вновь ищется ближайший к ней центр кластера. Полученные на данном проходе расстояния представлены в табл. 5.4.

Таблица 5.4

Нахождение ближайшего центра для каждой точки (третий проход)

Точка	Расстояние от m_1	Расстояние от m_2	Принадлежит кластеру
A	1,27	3,01	1
B	2,15	1,03	2
C	3,02	0,25	2
D	3,95	1,03	2
E	0,35	3,09	1
F	2,76	0,75	2
G	0,79	3,47	1
H	1,06	2,66	1

Следует отметить, что записей, сменивших кластер на третьем проходе

алгоритма, не было. Новая сумма квадратов ошибок составит:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \left(p - m_i \right)^2 = 1,27^2 + 1,03^2 + 0,25^2 + 1,03^2 + 0,35^2 + 0,75^2 + 0,79^2 + 1,06^2 = 6,$$

Таким образом, сумма квадратов ошибок изменилась незначительно по сравнению с предыдущим проходом.

Шаг 4, проход 3. Для каждого кластера вновь вычисляется центроид, и центр кластера в него перемещается. Но поскольку на данном проходе ни одна запись не изменила своего членства в кластерах и положение центроидов не поменялось, алгоритм завершает работу.

Алгоритм *k-means* приобрел популярность благодаря следующим свойствам. Один из основных недостатков, присущих алгоритму *k-means,* — отсутствие четких критериев выбора числа кластеров, целевой функции их инициализации и модификации. Кроме того, он очень чувствителен к «шумам» в данных и аномальным значениям, поскольку они способны существенно повлиять на среднее значение, используемое при вычислении положений центройдов. Чтобы снизить влияние таких факторов, как шумы и аномальные значения, иногда на каждой итерации используют не среднее значение признаков, а их медиану. Данная модификация алгоритма называется *k-mediods* (*k*-медиан).

Алгоритм G-means

Одним из недостатков алгоритма *k-means* является отсутствие, как мы уже сказали, ясного Критерия для выбора оптимального числа кластеров. Действительно, пусть множество данных содержит 5 групп, внутри которых объекты похожи, а в различных группах существенно отличаются. Тогда логично задать k = 5, чтобы каждая группа оказалась ассоциирована с отдельным кластером. Но, как правило, такая априорная информация отсутствует и аналитику приходится действовать методом проб и ошибок. Если будет выбрано k = 3, то какие-то две из 5 групп окажутся «распылены» по «чужим» кластерам и не будут обнаружены. Это может привести к потере потенциально ценных знаний. Кроме того, в общем то, мало похожие объекты могут оказаться в одном кластере, что затруднит интерпретацию результатов анализа. Если выбрать большее число кластеров, например, k = 7, то будут сформированы «лишние» кластеры. При этом получится, что достаточно похожие объекты окажутся в различных кластерах. Это проиллюстрировано на рис. 5.4, где знаком «+» отмечены центры кластеров, сформированные обычным алгоритмом k-means.

На рис. 5.4.а, где для 5 групп сформировано всего 3 кластера, можно увидеть, что группы 1 и 5, а также 2 и 3 оказались ассоциированы с одним

центром, то есть попали в общий кластер. В результате аналитик может сделать ошибочный вывод о сходстве объектов из групп, объединенных в один кластер, На рис. 5.4.б, где число кластеров было задано слишком большим, наблюдается обратная ситуация, группы 3 и 5 оказались ассоциированы с двумя центрами, и, следовательно, каждая из них разбросана по двум кластерам.

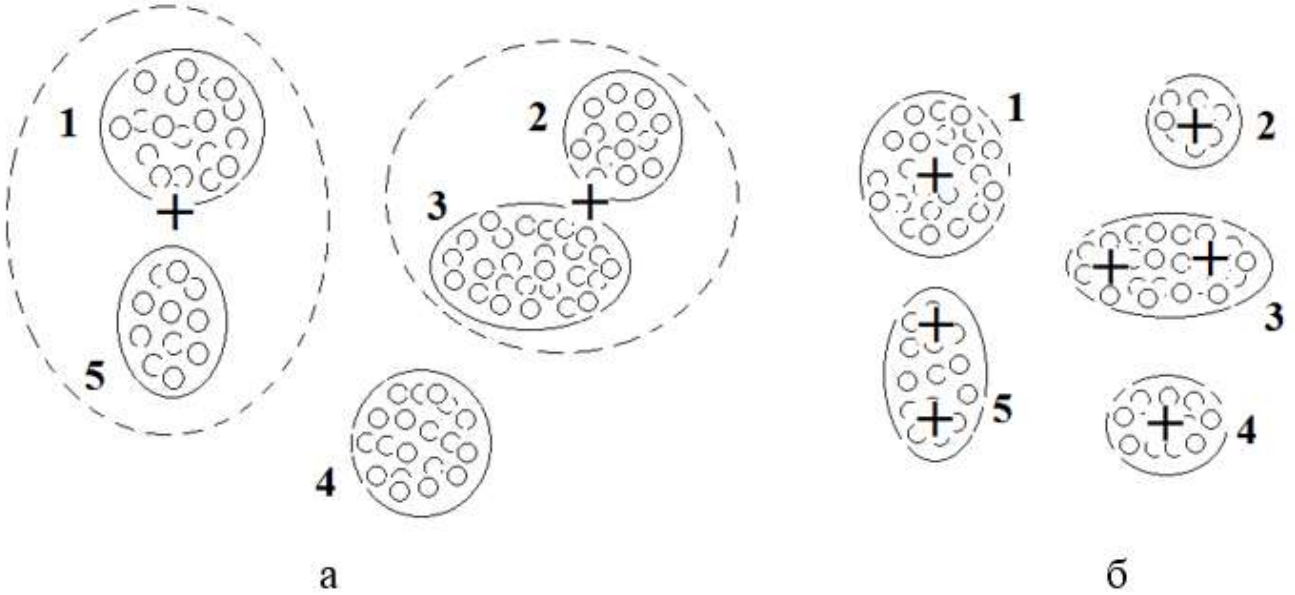

Рис. 5.4. Недостатки алгоритма k-means

Чтобы решить данную проблему, было разработано большое количество алгоритмов, позволяющих производить автоматический выбор числа кластеров, оптимального с точки зрения того или иного критерия. Обычно в них строится несколько моделей для различных значений k, а затем выбирается наиболее подходящая. Примерами могут служить алгоритм X-means, в основе которого лежат байесовские оценки логарифмического правдоподобия; алгоритм, основанный на принципе минимальной длины описания (MDL) и др.

Одним из самых популярных алгоритмов кластеризации с автоматическим выбором числа кластеров является G- means. В его основе лежит предположение о том, что кластеризуемые данные подчиняются некоторому унимодальному закону распределения, например, гауссовскому (откуда и название алгоритма), тогда центр кластера, определяемый как среднее значений признаков, попавших в него объектов, может рассматриваться как мода соответствующего распределения. Если исходные данные описываются унимодальным гауссовским распределением с заданным средним, то можно предположить, что все они относятся к одному кластеру. Если распределение данных не гауссовское, то можно попробовать выполнить разделение на два кластера. Если в этих кластерах распределения окажутся близки к гауссовскому, то можно в первом приближении считать, что k = 2 будет оптимальным. В противном случае будут

построены новые модели с большим числом кластеров, и так до тех пор, пока распределение в каждом из них не окажется достаточно близким к гауссовскому. Такая модель и, соответственно, число кластеров в ней будут считаться оптимальными.

Алгоритм G-means является итеративным, где на каждом шаге с помощью обычного k-means строится модель с определенным числом кластеров. Обычно G-means начинает работу с небольшого значения k, и на каждой итерации оно увеличивается. Как правило, начальное значение k выбирается равным 1. На каждой итерации увеличение k производится за счет разбиения кластеров, в которых данные не соответствуют гауссовскому распределению.

Алгоритм принимает решение о дальнейшем разбиении на основе статистического теста данных, связанных с каждым центроидом. Если при этом будет обнаружено, что они распределены по гауссовскому закону, то дальнейшего смысла в их разбиении нет. Фактически в процессе работы G-means алгоритм k-means будет повторен k раз.

Проблемы алгоритмов кластеризации.

Ранее уже отмечалось, что одно и то же множество объектов можно разбить на несколько кластеров по разному. Это привело к изобилию алгоритмов кластеризации.

Пожалуй, ни одна другая задача Data Mining не имеет в своем арсенале столько алгоритмов и методов решения. Причинами сложившейся ситуации является несколько факторов, имеющих общее объяснение: не существует одного универсального алгоритма кластеризации. Перечислим эти факторы.

1. Неопределенность в выборе критерия качества кластеризации.

В Data Mining при решении задач кластеризации популярны алгоритмы, которые ищут оптимальное разбиение множества данных на группы. Критерий оптимальности определяется видом целевой функции, от которой зависит результат кластеризации.

Например, семейство алгоритмов k-means показывает хорошие результаты, когда данные в пространстве образуют компактные сгустки, четко отличимые друг от друга. Поэтому и критерий качества основан на вычислении расстояний точек до центров кластера.

Главная трудность в выборе критерия качества кластеризации заключается в том, что на практике в условиях, когда объекты описываются десятками и сотнями свойств, становится сложно оценить взаимное расположение объектов и подобрать адекватный алгоритм.

2. Трудность выбора меры близости, обусловленная различной природой данных

Особенность данных такова, что в таблицах, описывающих свойства объектов, могут присутствовать различные типы данных. Для задачи кластеризации это чаще всего числовые и строковые данные. Строковый тип, в свою очередь, делится на упорядоченный и категориальный.

Присутствие тех или иных типов данных в наборе определяет его природу. Назовём набор данных числовым, если он состоит только из целых и вещественных признаков. Для вычисления расстояний между объектами таких наборов чаще всего применяется популярная метрика евклидово расстояние.

Назовем набор данных строковым, если он состоит из упорядоченных и категориальных признаков (сюда же относятся логические признаки). Для упорядоченных можно также использовать евклидово расстояние, закодировав значения признака целыми числами. А вот к категориальным типам эта мера не подходит. Здесь нужно применять специальную меру расстояния, например функцию отличия (difference function), которая задается следующим образом:

$$d(x, y) = \begin{cases} 0, & \text{если } x = y \\ 1, & \text{в остальных случаях} \end{cases}$$

где x и y – категориальные значения.

Наборы данных, содержащие признаки, к которым нельзя применять одну и ту же меру расстояния, называются смешанными.

Проиллюстрируем вышесказанное на примере. Пусть требуется вычислить попарные расстояния между следующими объектами с атрибутами *Возраст, Цвет глаз, Образование*: (1) {23, карий, высшее}; (2) {25, зеленый, среднее}; (3) {26, серый, среднее}.

Первый атрибут является числовым, остальные – строковыми, причем признак *Цвет глаз* имеет категориальный тип, а *Образование* – упорядоченный, Если для вычисления расстояний выберем одну метрику – евклидову, то возникнут проблемы с признаком *Цвет глаз*. Для него подходит только функция отличия.

Главная трудность в выборе меры близости состоит в том, что необходимость использования комбинации метрик ухудшает работу алгоритма, а эффективных алгоритмов кластеризации для смешанных наборов данных мало.

3. Различные требуемые машинные ресурсы (память и время)

Алгоритмы кластеризации, как и любые другие, имеют различную вычислительную сложность. Вопрос масштабируемости в кластеризации стоит особенно остро, так как эта задача Data Mining часто выступает первым шагом в

анализе: после выделения схожих групп применяются другие методы, для каждой группы строится отдельная модель. В частности, именно из-за больших вычислительных затрат в Data Mining не получили распространение иерархические алгоритмы, которые строят полное дерево вложенных кластеров.

Для кластеризации больших массивов данных, содержащих миллионы строк, разработаны специальные алгоритмы, позволяющие добиваться приемлемого качества за несколько проходов по набору данных. Такие задачи, к примеру, актуальны при сегментации покупателей супермаркета по их чекам.

Получение масштабируемых алгоритмов основано на идее отказа от локальной функции оптимизации. Парное сравнение объектов между собой в алгоритме k-means есть не что иное, как локальная оптимизация: на каждом шаге необходимо рассчитывать расстояние от центра кластера до каждого объекта. Это ведет к большим вычислительным затратам. При задании глобальной функции оптимизации добавление новой точки в кластер не требует больших вычислений: расстояние рассчитывается на основе старого значения, нового объекта и параметров кластера. К сожалению, ни k-means, ни сеть Кохонена не используют глобальную функцию оптимизации.

Выбор числа кластеров.

Хоть и редко, но встречаются случаи, когда точно известно, сколько кластеров нужно выделить. Но чаще всего перед процедурой кластеризации этот вопрос остается открытым. Если алгоритм не поддерживает автоматическое определение оптимального количества кластеров (как, например, G-means), есть несколько эмпирических правил, которые можно применять при условии, что каждый кластер будет в дальнейшем подвергаться содержательной интерпретации аналитиком.

- Двух или трех кластеров, как правило, недостаточно: кластеризация будет слишком грубой, приводящей к потере информации об индивидуальных свойствах объектов.
- Больше десяти кластеров не укладываются в «число Миллера 7 ∓ 2»: аналитику трудно держать в кратковременной памяти столько кластеров.

Поэтому в подавляющем большинстве случаев число кластеров варьируется от 4 до 9.

При взгляде на изобилие алгоритмов кластеризации возникает вопрос, существует ли объективная, естественная кластеризация, или она всегда носит субъективный характер? Не существует. Любая кластеризация субъективна, потому что выполняется на основе конечного подмножества свойств объектов. А выбор этого подмножества всегда субъективен, как и выбор критерия качества и

меры близости.

Популярные алгоритмы k-means и сеть Кохонена изначально разрабатывались для числовых данных, и, хотя впоследствии появились их модификации, применяемые к смешанным наборам данных, они все равно лучше решают задачи кластеризации на числовых признаках.

Чтобы применять кластеризацию корректно и снизить риск получения результатов моделирования, не имеющих никакого отношения к действительности, необходимо придерживаться следующих правил.

Правило 1. Перед кластеризацией четко обозначьте цели ее проведения: облегчение дальнейшего анализа, сжатие данных и т. п. Кластеризация сама по себе не представляет особой ценности.

Правило 2. Выбирая алгоритм, убедитесь, что он корректно работает с теми данными, которыми вы располагаете для кластеризации. В частности, если присутствуют категориальные признаки, удостоверьтесь, что та реализация алгоритма, которую вы используете, умеет правильно обрабатывать их. Это особенно актуально для алгоритмов k-means и сетей Кохонена (впрочем, и для других, основанных на метриках), в большинстве случаев при меняющих евклидову меру расстояния. Если алгоритм не умеет работать со смешанными наборами данных, постарайтесь сделать набор данных однородным, то есть отказаться от категориальных или числовых признаков.

Правило 3. Обязательно проведите содержательную интерпретацию каждого полученного кластера: постарайтесь понять, почему объекты были сгруппированы в определенный кластер, что их объединяет. Для этого можно использовать визуальный анализ, графики, кластерограммы, статистические характеристики кластеров, карты. Полезно каждому кластеру дать емкое название, состоящее из нескольких слов. Встречаются ситуации, кода алгоритм кластеризации не выделил никаких особых групп. Возможно, набор данных и до кластеризации был однороден, не расслаивался на изолированные подмножества, а кластеризация подтвердила эту гипотезу.

Таким образом, не существует единого универсального алгоритма кластеризации. При использовании любого алгоритма важно пони мать его достоинства, недостатки и ограничения. Только тогда кластеризация будет эффективным инструментом в руках аналитика.