ЛЕКЦИЯ 3а. Интеллектуальный анализ данных: задача кластеризации

Методы по способу обработки данных:

1. Иерархические методы:

При иерархической кластеризации выполняется последовательное объединение меньших кластеров в большие или разделение больших кластеров на меньшие.

1.1. Агломеративные методы AGNES (Agglomerative Nesting): алгоритмы CURE (Clustering Using REpresentatives); ROCK; CHAMELEON и т.д.

Эта группа методов характеризуется последовательным объединением исходных элементов и соответствующим уменьшением числа кластеров. В начале работы алгоритма все объекты являются отдельными кластерами. На первом шаге наиболее похожие объекты объединяются в кластер. На последующих шагах объединение продолжается до тех пор, пока все объекты не будут составлять один кластер.

1.2. Дивизимные методы DIANA (Divisive Analysis): Алгоритм BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies); Алгоритм MST (Algorithm based on Minimum Spanning Trees) и т.д.

Эта группа методов характеризуется последовательным разделением исходного кластера, состоящего из всех объектов, и соответствующим увеличением числа кластеров. В начале работы алгоритма все объекты принадлежат одному кластеру, который на последующих шагах делится на меньшие кластеры, в результате образуется последовательность расщепляющих групп.

2. Неиерархические методы.

Неиерархическая кластеризация включает в себя формирование новых кластеров путем слияния или разделения кластеров. Он не следует древовидной структуре, подобной иерархической кластеризации. Этот метод группирует данные, чтобы максимизировать или минимизировать некоторые критерии оценки. Эта кластеризация является эффективным способом неиерархической кластеризации. В этом методе разделы создаются таким образом, что неперекрывающиеся группы не имеют иерархических отношений между собой.

Итеративные методы: k-средних (k-means); PAM (k-means + k-medoids); DBSCAN, CLOPE; LargeItem и т.д.

Процесс неиерархической кластеризации всегда является итеративным. Итеративные методы кластеризации различаются выбором следующих параметров: начальной точки; правилом формирования новых кластеров; правилом остановки.

Такая неиерархическая кластеризация состоит в разделении набора данных на определенное количество отдельных кластеров. Существует два подхода. Первый заключается в определении границ кластеров как наиболее плотных участков в многомерном пространстве исходных данных, т.е. определение кластера там, где имеется большое "сгущение точек". Второй подход заключается в минимизации меры различия объектов.

Также можно выделить кластеризацию:

Методы по способу анализа данных: четкие; нечеткие.

Методы по количеству применений алгоритмов кластеризации: с одноэтапной кластеризацией; с многоэтапной кластеризацией.

Методы по возможности расширения объема обрабатываемых данных: масштабируемые; немасштабируемые.

Методы по времени выполнения кластеризации: потоковые; не потоковые.

Пока будем рассматривать только неиерархические итеративные методы

Алгоритм PAM

Использование средних подразумевает, что кластеризация k-средних очень чувствительна к выбросам. Это может серьезно повлиять на назначение наблюдений кластерам. Алгоритм PAM (Partitioning Around Medoids) обеспечивает более надежный алгоритм.

Алгоритм PAM очень похож на алгоритм K-means, в основном потому, что оба являются алгоритмами кластеризации, другими словами, оба разделяют множество объектов на группы (кластеры) и работа обоих основана на попытках минимизировать ошибку, но PAM работает с медоидами – объектами, являющимися частью исходного множества и представляющими группу, в которую они включены, а K-means работает с центроидами - искусственно созданными объектами, представляющими кластер.

Введем следующие обозначения для формального описания алгоритма PAM. Пусть $O = \{o_1, o_2, \ldots, o_n\}$ — это множество кластеризуемых объектов, где каждый объект — это кортеж, состоящий из p вещественных чисел. Пусть k количество кластеров, $k \ll n, C = \{c_1, c_2, \ldots, c_k\}$ множество медоидов, $C \subset O$, и $\rho: O \times C \rightarrow R$ — это метрика расстояния.

На каждой итерации выбирается пара медоид c_i и не-медоид o_j такая, что замена медоида на не-медоид дает лучшую кластеризацию из возможных. Оценка кластеризации выполняется с помощью целевой функции, вычисляемой как сумма расстояний от каждого объекта до ближайшего медоида

Таким образом, после нахождения набора из k медоидов кластеры строятся путем сопоставления каждого наблюдения с ближайшим медоидом. Затем каждый выбранный медоид n и каждая немедоидная точка данных меняются местами, и вычисляется целевая функция. Целевая функция соответствует сумме отличий всех объектов от их ближайшего медоида.

$$E = \sum_{j=1}^{n} \min_{1 \le i \le k} \rho(c_i, o_j).$$

Псевдокод можно алгоритм PAM можно представить следующим образом:

Вход : Множество объектов O, количество кластеров k
Выход: Множество кластеров C
1 Инициализировать C ; // фаза BUILD
2 **repeat**// фаза SWAP
3 | Вычислить T_{min} ;
4 | Поменять местами c_{min} и o_{min};
5 **until** $T_{min} < 0$;

Рис. 1. Псевдокод алгоритма PAM

PAM состоит из двух фаз: BUILD и SWAP. В фазе BUILD (строить) выполняется первичная кластеризация, в которой последовательно выбирается k объектов в качестве медоидов. Первый объект c_1 – это объект, сумма расстояний от которого до всех остальных объектов является наименьшей:

$$c_1 = \arg\min_{1 \le h \le n} \sum_{j=1}^{n} \rho(o_h, o_j).$$

Затем выбирается следующий объект, минимизирующий целевую функцию. Для этого производится вычисление целевой функции относительно ранее выбранных объектов c и каждого из невыбранных объектов o:

$$c_2 = \arg\min_{1 \le h \le n} \sum_{j=1}^{n} \min(\rho(c_1, o_j), \rho(o_h, o_j)),$$

$$c_3 = \arg\min_{1 \le h \le n} \sum_{j=1}^{n} \min(\min_{1 \le l \le 2}(\rho(c_l, o_j)), \rho(o_h, o_j)),$$

...

$$c_k = \arg\min_{1 \le h \le n} \sum_{j=1}^{n} \min(\min_{1 \le l \le k-1}(\rho(c_l, o_j)), \rho(o_h, o_j)).$$

Эта процедура повторяется, пока не будет выбрано k объектов.

В фазе SWAP алгоритм PAM пытается улучшить множество медоидов C. Алгоритм выполняет поиск пары объектов (c_{min}, o_{min}), минимизирующих целевую функцию. Для этого перебираются все пары объектов (c_i, o_h), где c_i — это медоид, а o_h не-медоид. Вычисляется изменение целевой функции при исключении c_i из множества медоидов и включении o_h вместо него. Обозначим это изменение как T_{ih}, а минимальное значение T_{min} достигается на паре (c_{min}, o_{min}). Если $T_{min} > 0$, тогда множество C не может быть улучшено, и алгоритм завершается. Для описания вычисления T_{ih} введем следующие обозначения. Пусть $D = \{d_1, d_2, \ldots, d_n\}$ – это множество расстояний от каждого объекта до ближайшего медоида. Пусть $S = \{s_1, s_2, \ldots, s_n\}$ — это множество расстояний от каждого объекта до второго ближайшего медоида. Пусть C_{jih} — это вклад не-медоида o_j в T_{ih} при замене c_i на o_h. В этом случае T_{ih} определяется как сумма C_{jih}:

$$T_{ih} = \sum_{j=1}^{n} C_{jih}$$

Псевдокод вычисления C_{jih} представлен на рис. 2.

Вход : o_j, c_i, o_h, d_j, s_j
Выход: C_{jih}

```
if ρ(o_j, c_i) > d_j and ρ(o_j, o_h) > d_j then
    C_jih ← 0
else if ρ(o_j, c_i) = d_j then
    if ρ(o_j, o_h) < s_j then
        C_jih ← ρ(o_j, o_h) − d_j
    else
        C_jih ← s_j − d_j
    end
else if ρ(o_j, o_h) < d_j then
    C_jih ← ρ(o_j, o_h) − d_j
end
```

Рис. 2. Вычисление C_{jih}

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 как решение проблемы разбиения (изначально пространственных) данных на кластеры произвольной формы. Большинство алгоритмов создают

кластеры, по форме близкие к сферическим, так как минимизируют расстояние элементов до центра кластера. Авторы DBSCAN экспериментально показали, что их алгоритм способен распознать кластеры различной формы. 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. Таким образом, кластеризация, основанная на плотности, использует не расстояние между точками, а локальную плотность точек для определения кластеров.

Дадим формальные определения терминам, используемым при описании алгоритма DBSCAN.

Определение. ε–окрестностью точки p, обозначаемой $N_{Eps}(p)$ назовём множество $N_{Eps}(p) = \{q \in D | dist(p,q) < \varepsilon\}$.

Иногда называют ε-соседями точки $p \in \mathbb{R}^d$ сферу радиуса ε вокруг этой точки. Где $dist(p,q)$ представляет собой расстояние между точками p и q. Здесь как правило, речь идёт о евклидовом расстоянии, то есть, $dist(p,q) = \sqrt{(p-q)^2}$, но могут применяться и другие метрики.

В любом множестве может существовать два типа точек: внутренние (core points) – находящиеся внутри кластера и граничные (border points) – точки, находящиеся на границе кластера (рис.3).

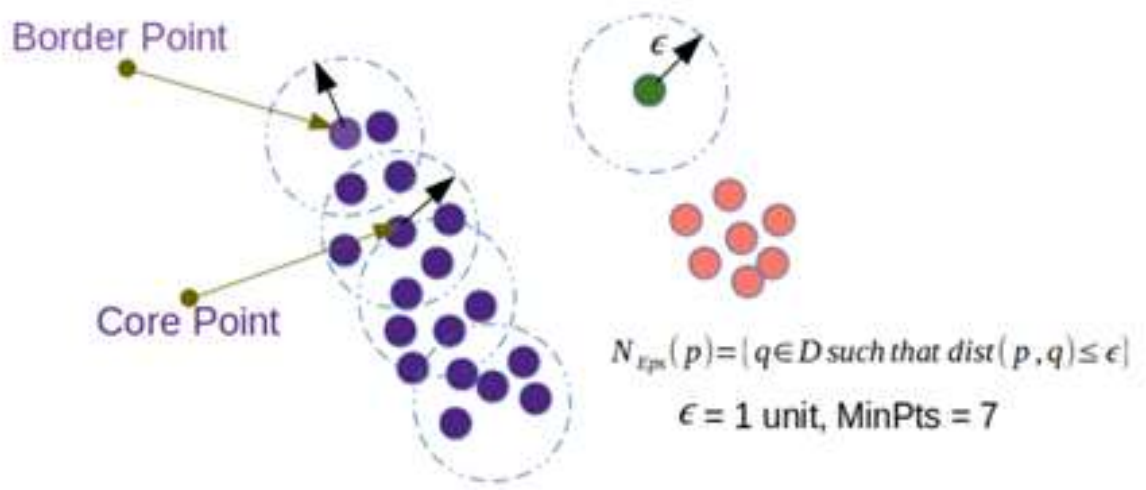

Рис. 3.

Определение. Точка p является непосредственно-достижимой из q, если:

1. $p \in N_{Eps}(p) - p$ лежит в окрестности q.
2. $|NEps(q)| \geq MinPts$.

Определение. Точка является достижимой, если ∃ такая цепь точек $p_1, \ldots, p_n, p_1 = q, p_n n = p$, что $p_i + 1$ непосредственно достижима из p_i.

Определение. Точка p плотностно-связна с точкой q, если ∃ точка O, такая, что p и q достижимы из O.

Определение. Пусть D – множество точек. Кластером C назовём непустое множество в D, удовлетворяющее следующим условиям:

1. $\forall p, q$: если $p \in C$ и q плотностнодостижимы из p, тогда $q \in C$;

2. $\forall p, q \in C$: p плотностно связно с q.

Таким образом, будем считать, что точка p напрямую достижима по плотности из точки q, если $p \in N_\varepsilon(q)$ и q является базовой точкой. Точка p достижима по плотности из точки q, если существует последовательность точек $p_0, p_1, \dots, p_l,$ такая, что $p = p_0, q = xl$ и p_i напрямую достижима из p_{i-1} для всех $i = 1, \dots, l$. Другими словами, существует множество базовых точек, ведущих от q к p. Заметим, что отношение достижимости по плотности асимметричное или направленное. Определим любые две точки p и q как связанные по плотности, если существует базовая точка o такая, что и p, и q достижимы из o. Кластером, основанным на плотности, называется максимальное множество точек, связанных по плотности.

Определение. Пусть $C_1, C_2, \dots, C_k$ – кластеры в D. Тогда шум – множество точек не принадлежит $\notin Cj$, т.е. шум $= \{p \in D | p \notin C_j\}$.

Будем называть точку $p \in D$ базовой (ядерной), если хотя бы $minpts$ точек являются её ε-соседями. Другими словами, точка **x** является базовой, если $N_\varepsilon p \geq minPts$, где $minPts$ – задаваемая пользователем локальная плотность или порог частоты. Граничной точкой называется точка, которая не удовлетворяет порогу $minpts$, то есть, для неё $N_\varepsilon p \leq minPts$, но при этом она является ε-соседней для некоторой базовой точки z, то есть, $p \in N_\varepsilon(z)$. Наконец, если точка не является ни базовой, ни граничной, она считается точкой шума или выбросом.

Шаги алгоритма DBSCAN:

1. Алгоритм начинается с произвольной точки, которая не была посещена, и информация о его окрестности извлекается из параметра.

2. Если этот пункт содержит $minPts$ в ε-окрестности начинается формирование кластера. В противном случае точка помечается как шум. Эта точка может быть позже найдена в ε-окрестности другой точки и, таким образом, может стать частью кластера. Здесь важна концепция достижимости плотности и точек, связанных плотностью.

3. Если точка найдена как центральная точка, то точки в окрестности также являются частью кластера. Таким образом, все точки, найденные в ε-

окрестности, добавляются вместе с их собственной ε-окрестностью, если они также являются центральными точками.

4. Вышеописанный процесс продолжается до тех пор, пока кластер, связанный плотностью, не будет найден полностью.

5. Процесс возобновляется с новой точкой, которая может быть частью нового кластера или помечена как шум.

Согласно алгоритму, можно составить следующий программный код (рис. 4). Сначала DBSCAN вычисляет ϵ-соседей $N_\epsilon(x_i)$ для всех точек x_i датасета D, а затем проверяет, являются ли они базовыми (строки 2-5). Кроме того, алгоритм присваивает всем точкам значение идентификатора кластера $id\ x_i = \emptyset$, отмечая, что они не принадлежат ни одному кластеру. Далее, начиная с каждой базовой точки, не присвоенной ни одному кластеру, метод рекурсивно ищет все точки, связанные по плотности с исходной, и присваивает их одному кластеру (строка 10). Некоторые граничные точки могут быть достижимы из базовых из более чем одного кластера, они могут быть присвоены любому кластеру или всем (если допускается пересечение кластеров). Те точки, которые не принадлежат ни одному кластеру, помечаются как выбросы или шум.

```
   DBSCAN (D, ϵ, minpts):
1  Core ← ∅
2  foreach x_i ∈ D do // Find the core points
3      Compute N_ϵ(x_i)
4      id(x_i) ← ∅ // cluster id for x_i
5      if N_ϵ(x_i) ≥ minpts then Core ← Core ∪ {x_i}
6  k ← 0 // cluster id
7  foreach x_i ∈ Core, such that id(x_i) = ∅ do
8      k ← k + 1
9      id(x_i) ← k // assign x_i to cluster id k
10     DensityConnected (x_i, k)
11 C ← {C_i}_{i=1}^k, where C_i ← {x ∈ D | id(x) = i}
12 Noise ← {x ∈ D | id(x) = ∅}
13 Border ← D \ {Core ∪ Noise}
14 return C, Core, Border, Noise

   DensityConnected (x, k):
15 foreach y ∈ N_ϵ(x) do
16     id(y) ← k // assign y to cluster id k
17     if y ∈ Core then DensityConnected (y, k)
```

Рис. 4.

Достоинства алгоритма DBSCAN:

• не требует заранее определения количества кластеров, в противоположность алгоритму k-means;

• определяет кластеры различной формы. Различает кластеры даже тогда, когда два кластера не соединены друг с другом, но один из кластеров расположен внутри другого;

• способен выделить шум – точки, не принадлежащие ни одному кластеру;

• для работы требует лишь два входных параметра, не чувствителен к порядку записей в базе данных.

Недостатки алгоритма DBSCAN:

• качество реализации сильно зависит от выбора функции расстояния;

• DBSCAN плохо справляется с множествами, в которых имеется большой разброс расстояний между элементами; в таких множествах иногда не удаётся подобрать оптимальные параметры $minPts$ и ε для всех кластеров.

DBSCAN можно рассматривать как поиск связных компонент в графе, где вершины соответствуют базовым точкам в датасете, и существует (ненаправленное) ребро между двумя вершинами (базовыми точками), если расстояние между ними меньше ϵ, то есть, каждая из них является ϵ-соседом для другой.

Алгоритм CLOPE

Основная идея.

Задачи кластеризации больших массивов категорийных данных весьма актуальна для систем анализа данных. Категорийные данные встречаются в любых областях: производство, коммерция, маркетинг, медицина и т.д. Категорийные данные включают в себя и так называемые транзакционные данные: чеки в супермаркетах, логи посещений веб-ресурсов. Сюда же относится анализ и классификация текстовых документов.

Под категорийными данными понимаются качественные характеристики объектов, измеренные в шкале наименований. Напомним: при использовании шкалы наименований указывается только, одинаковы или нет объекты относительно измеряемого признака.

Применять для кластеризации объектов с категорийными признаками традиционные алгоритмы неэффективно, а часто - невозможно. Основные трудности связаны с высокой размерностью и гигантским объемом, которыми часто характеризуются такие базы данных. Алгоритмы, основанные на парном вычислении расстояний (k-means и аналоги) эффективны в основном на числовых

данных. Их производительность на массивах записей с большим количеством нечисловых факторов неудовлетворительная. И дело даже не столько в сложности задания метрики для вычисления расстояния между категорийными атрибутами, сколько в том, что на каждой итерации алгоритма требуется попарно сравнивать объекты между собой, а итераций может быть очень много. Для таблиц с миллионами записей и тысячами полей это неприменимо.

Поэтому в последнее десятилетие ведутся активные исследования в области разработки масштабируемых алгоритмов кластеризации категорийных и транзакционных данных. К ним предъявляются особые требования, а именно:

- минимально возможное количество «сканирований» таблицы базы данных;
- работа в ограниченном объеме оперативной памяти компьютера;
- работу алгоритма можно прервать с сохранением промежуточных результатов, чтобы продолжить вычисления позже;
- алгоритм должен работать, когда объекты из базы данных могут извлекаться только в режиме однонаправленного курсора (т.е. в режиме навигации по записям).

На сегодняшний день предложено свыше десятка методов для работы с категорийными данными, например, семейство иерархических кластерных алгоритмов. Но не всегда они удовлетворяют перечисленным выше требованиям. Одним из эффективных считается алгоритм LargeItem, который основан на оптимизации некоторого глобального критерия. Этот глобальный критерий использует параметр поддержки (в терминологии здесь много общего с алгоритмами для выявления ассоциативных правил). Вообще, вычисление глобального критерия делает алгоритм кластеризации во много раз быстрее, чем при использовании локального критерия при парном сравнении объектов, поэтому «глобализация» оценочной функции - один из путей получения масштабируемых алгоритмов.

Алгоритм CLOPE очень похож на LargeItem, но быстрее и проще в программной реализации. CLOPE предложен в 2002 году группой китайских ученых. При этом он обеспечивает более высокую производительность и лучшее качество кластеризации в сравнении с алгоритмом LargeItem и многими иерархическими алгоритмами.

Для начала формализуем рассматриваемую задачу кластеризации для категорийных данных. Все изложение будет идти как будто бы у нас в наличии имеется база транзакционных данных. Под термином транзакция здесь понимается некоторый произвольный набор объектов, будь это список ключевых

слов статьи, товары, купленные в супермаркете, множество симптомов пациента, характерные фрагменты изображения и так далее. Задача кластеризации транзакционных данных состоит в получении такого разбиения всего множества транзакций, чтобы похожие транзакции оказались в одном кластере, а отличающиеся друг от друга - в разных кластерах.

В основе алгоритма кластеризации CLOPE лежит идея максимизации глобальной функции стоимости, которая повышает близость транзакций в кластерах при помощи увеличения параметра кластерной гистограммы. Рассмотрим простой пример из 5 транзакций: $\{(a,b),(a,b,c),(a,c,d),(d,e),(d,e,f)\}$. Представим себе, что мы хотим сравнить между собой следующие два разбиения на кластеры:

(1) $\{\{ab,abc,acd\},\{de,def\}\}$

(2) $\{\{ab,abc\},\{acd,de,def\}\}$.

Для первого и второго вариантов разбиения в каждом кластере рассчитаем количество вхождений в него каждого элемента транзакции, а затем вычислим высоту (H) и ширину (W) кластера. Например, кластер $\{ab,abc,acd\}$ имеет вхождения $a:3, b:2, c:2$ с $H=2$ (кол-во квадратов 8 на ширину 4) и $W=4$ (кол-во разных элементов). Для облегчения понимания на рис. 5 эти результаты показаны геометрически в виде гистограмм.

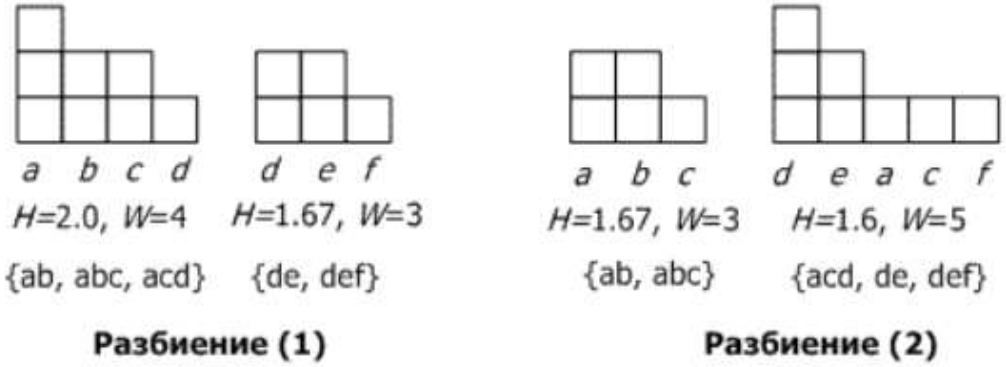

Рис. 5. Гистограммы двух разбиений

Качество двух разбиений оценим, проанализировав их высоту H и ширину W. Кластеры $\{de, def\}$ и $\{ab, abc\}$ имеют одинаковые гистограммы, следовательно, равноценны. Гистограмма для кластера $\{ab,abc,acd\}$ содержит 4 различных элемента и имеет площадь 8 блоков ($H=2.0, H/W=0.5$), а кластер $\{acd,de,def\}$ – 5 различных элементов с такой же площадью ($H=1.6, H/W=0.32$). Очевидно, что разбиение (1) лучше, поскольку обеспечивает большее наложение транзакций друг на друга (соответственно, параметр H там выше).

На основе такой очевидной и простой идеи геометрических гистограмм и работает алгоритм CLOPE (англ.: Clustering with sLOPE).

Алгоритм.

Пусть имеется база транзакций D, состоящая из множества транзакций $\{t_1, t_2, \dots, t_n\}$. Каждая транзакция есть набор объектов $\{i_1, \dots, i_m\}$. Множество кластеров $\{C_1, \dots, C_k\}$ есть разбиение множества $\{t_1, \dots, t_n\}$, такое, что $C_1, \dots, C_k = \{t_1, \dots, t_n\}$ и $C_i \neq \emptyset \wedge C_i \cap C_j = \emptyset$, для $1 \leq i, j \leq k$. Каждый элемент C_i называется кластером, а n, m, k – количество транзакций, количество объектов в базе транзакций и число кластеров соответственно.

Каждый кластер C имеет следующие характеристики:

- $D(C)$ – множество уникальных объектов;
- $Occ(i, C)$ – количество вхождений (частота) объекта i в кластер C;
- $S(C) = \sum_{i \in D(C)} Occ(i, C) = \sum_{t_i \in C} |t_i|$;
- $W(C) = D(C)$;
- $H(C) = S(C)/W(C)$.

Гистограммой кластера C называется графическое изображение его расчетных характеристик: по оси OX откладываются объекты кластера в порядке убывания величины $Occ(i, C)$, а сама величина $Occ(i, C)$ – по оси OY (рис. 6).

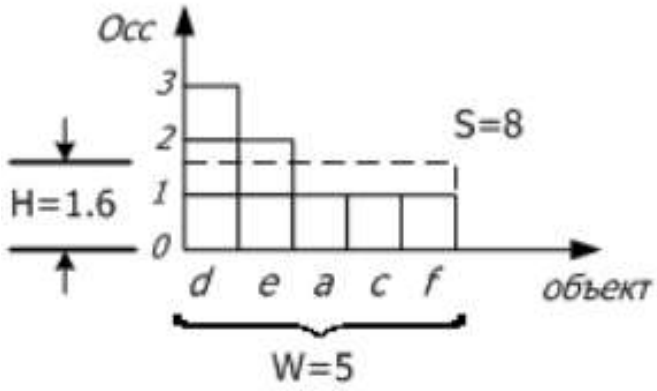

Рис. 6. Иллюстрация гистограммы кластера

На рис.4 $S(C)$, равное 8, соответствует площади прямоугольника, ограниченного осями координат и пунктирной линией. Очевидно, что чем больше значение H, тем более "похожи" две транзакции. Поэтому алгоритм должен выбирать такие разбиения, которые максимизируют H.

Однако учитывать одно только значение высоты H недостаточно. Возьмем базу, состоящую из 2-х транзакций: $\{abc, def\}$. Они не содержат общих объектов, но разбиение $\{\{abc, def\}\}$ и разбиение $\{\{abc\}, \{def\}\}$ характеризуются одинаковой высотой $H = 1$. Получается, оба варианта разбиения равноценны. Но если для оценки вместо H(C) использовать градиент $G(C) = H(C)/W(C) = S(C)/W(C)^2$, то разбиение $\{\{abc\}, \{def\}\}$ будет лучше (градиент каждого кластера равен $1/3$ против $1/6$ у разбиения $\{\{abc, def\}\}$).

Обобщив вышесказанное, запишем формулу для вычисления глобального критерия – функции стоимости $Profit(C)$:

$$Profit(C) = \frac{\sum_{i=1}^{k} G(C_i) \times |C_i|}{\sum_{i=1}^{k} |C_i|} = \frac{\sum_{i=1}^{k} \frac{S(C_i)}{W(C_i)^r} \times |C_i|}{\sum_{i=1}^{k} |C_i|}$$

где $|C_i|$ – количество объектов в i-ом кластере, k – количество кластеров, r – положительное вещественное число большее 1.

С помощью параметра r, названного авторами CLOPE коэффициентом отталкивания, регулируется уровень сходства транзакций внутри кластера, и, как следствие, финальное количество кластеров. Этот коэффициент подбирается пользователем. Чем больше r, тем ниже уровень сходства и тем больше кластеров будет сгенерировано.

Формальная постановка задачи кластеризации алгоритмом CLOPE выглядит следующим образом: для заданных D и r найти разбиение C: $Profit(C, r) \rightarrow max$.

Реализация алгоритма

Предположим, что транзакции хранятся в таблице базы данных. Лучшее решение ищется в течение последовательного итеративного перебора записей базы данных. Поскольку критерий оптимизации имеет глобальный характер, основанный только на расчете H и W, производительность и скорость алгоритма будет значительно выше, чем при попарном сравнении транзакций.

Таким образом, алгоритм состоит из следующих этапов:

1. Инициализация. На этом этапе происходит первый проход по таблице с транзакциями для построения начального разбиения, для каждой транзакции определяется кластер исходя из максимизации стоимости.

Первая часть алгоритма – инициализация: для каждой транзакции вычисляется кластер, к которому она будет отнесена исходя из максимизации функции стоимости. Для этого вычисляется вспомогательная функция DeltaAdd, которая считает стоимость добавления транзакции к одному из существующих кластеров, либо к пустому. В итоге транзакция помещается в кластер, для которого цена добавления максимальна.

2. Итерация. На данном этапе для каждой транзакции проводится попытка перемещения в другой кластер для максимизации функции стоимости.

Вторая часть алгоритма - уточняющие итерации: здесь происходит попытка улучшить вычисленные на первом этапе кластеры. В каждой итерации происходит перебор всех транзакций, и для каждой транзакции вычисляется цена удаления ее из текущего кластера и цена добавления в другой кластер. Суммарная

стоимость перемещения будет являться суммой стоимостей удаления и добавления. Если максимальная стоимость перемещения данной транзакции в какой-либо больше нуля, то она перемещается в этот кластер.

Вычисления происходят до тех пор, пока за итерацию происходит хотя бы одно перемещение транзакции. Если за итерацию не было произведено ни одного перемещения, то текущее состояние кластеров является устойчивым и оптимальным.

Базовое представление алгоритма кластеризации CLOPE как видно очень простое, состоит из следующих шагов и может быть легко сформулировано следующим образом:

1. Инициализируйте целевой список кластеров $\{C\}$;
2. Извлеките следующий элемент $Ik, k \rightarrow [0; m]$ из набора данных $movies$, и если это не конец набора данных, перейдите к шагу 3;
3. Для текущего элемента Ik извлеките следующий кластер $Ci, i \rightarrow [0; n]$ из целевого списка, если это не конец списка кластеров, и перейдите к шагу 4;
4. Вычислите значение $delta$ для кортежа текущего кластера Ci и элемента Ik, который мы собираемся поместить в кластер Ci;
5. Выполните проверку, является ли значение $delta$ максимально возможным значением во всем списке кластеров. Если это так, поместите текущий элемент Ik в кластер Ci и вернитесь к шагу 2. В противном случае перейдите к шагу 3;

Дельта-функция

Чтобы вычислить значение дельта-функции, нам необходимо выполнить следующие шаги:

1. Присвоите текущую ширину $W(Ci)$ кластера Ci новой ширине $W(Ci)\_New = W(Ci)$;
2. Присвоите текущий размер кластера $S(Ci)$, увеличенный на 1, новому кластеру $S(Ci)\ \_New = S(Ci)\ +\ 1$;
3. Выполните проверку, если текущий элемент фильма Ik еще не был помещен в текущий кластер Ci;
4. Если это так, увеличьте значение новой ширины $W(Ci)\_New = W(Ci)\_New + 1$;
5. Вычислите и верните значение дельты как $S(Ci)\_New * 2/ W(Ci)\_New\hat{}r - S(Ci)/W(Ci)\hat{}r$;

В оригинальном техническом документе алгоритма CLOPE вычисление дельты выполняется для всей транзакции, состоящей из нескольких элементов.

$$\Delta = \frac{S_n(C_i) * (N_t + 1)}{W_n(C_i)^r} - \frac{S(C_i) * N_t}{W(C_i)^r}$$

где $S_n(C_i)$ – сумма вхождений каждого отдельного элемента фильма (например, площадь прямоугольника кластера), $W_n(C_i)$ – новая ширина кластера C_i после размещения элемента Ik, $S(C_i)$ – исходная площадь кластера C_i, $W(C_i)$ - исходная ширина кластера C_i, N_t - количество транзакций.

Как вы можете видеть, значение $delta$ в основном вычисляется как разница между градиентом кластера Ci и градиентом того же кластера сразу после того, как в него был помещен элемент Ik.

Области применения CLOPE

Алгоритм CLOPE предназначен для работы с транзакционными данными, но, как мы увидели, очень много наборов данных с категорийными атрибутами представляют собой транзакционные данные либо сводятся к ним. Ответы респондента в анкете, список ключевых слов документа, множество посещенных веб-ресурсов пользователя, симптомы больного, характеристики объекта – все это не что иное, как транзакция. Поэтому области применения CLOPE распространяются на все массивы категорийных баз данных.

Вообще, кластеризация транзакционных данных имеет много общего с анализом ассоциаций. Обе эти технологии Data Mining выявляют скрытые зависимости в наборах данных. Но есть и отличия. С одной стороны, кластеризация дает общий взгляд на совокупность данных, тогда как ассоциативный анализ находит конкретные зависимости между атрибутами. С другой стороны, ассоциативные правила сразу пригодны для использования, тогда как кластеризация чаще всего используется как первая стадия анализа.

В завершение подчеркнем преимущества алгоритма CLOPE:

1. Высокие масштабируемость и скорость работы, а также качество кластеризации, что достигается использованием глобального критерия оптимизации на основе максимизации градиента высоты гистограммы кластера. Он легко рассчитывается и интерпретируется. Во время работы алгоритм хранит в оперативной памяти небольшое количество информации по каждому кластеру и требует минимальное число сканирований набора данных. Это позволяет применять его для кластеризации огромных объемов категорийных данных;
2. CLOPE автоматически подбирает количество кластеров, причем это регулируется одним единственным параметром - коэффициентом отталкивания.