Кластеризация является одной из самых востребованных задач в машинном обучении. Нехватка информации о производительности работы того или иного алгоритма машинного обучения ведёт к увеличению роли специалистов в данной области. В реальной жизни задача выбора и настройки алгоритма машинного обучения является экспертной. Однако такая работа слишком затратная по времени, так как выполняется человеком практически вручную, а, следовательно, неэффективна. Это обуславливает актуальность разработки методологического, алгоритмического и программного инструментария для автоматизации поиска или синтеза решений для задач машинного обучения. Было предложено большое число подходов к решению задачи кластеризации. Один из этих подходов заключается в том, что производится процесс оптимизации некоторых функционалов, которые обычно не являются конечными мерами качества разбиений. Данный подход реализуют алгоритмы кластеризации. Исходя из всего изложенного выше, была поставлена задача по выбору наиболее быстрого и точного алгоритма для классификации кольцевых множеств.

Для начала рассмотрим алгоритм k-means. Это самый простой, но в то же время достаточно неточный метод кластеризации в классической реализации. Он разбивает множество элементов пространства на известное число кластеров k, которое было заранее задано. Алгоритм заключается в том, чтобы стремится минимизировать среднеквадратичное отклонение на точках каждого кластера. Основная идея заключается в том, что на каждой итерации перевычисляется центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике. Алгоритм завершается, когда на какой-то итерации не происходит изменения кластеров.

Демонстрация работы алгоритма кластеризации k-means

|  |  |
| --- | --- |
| https://upload.wikimedia.org/wikipedia/commons/thumb/5/5e/K_Means_Example_Step_1.svg/1024px-K_Means_Example_Step_1.svg.png  Случайно выбранные начальные точки и исходные точки. | https://upload.wikimedia.org/wikipedia/commons/thumb/a/a5/K_Means_Example_Step_2.svg/1024px-K_Means_Example_Step_2.svg.png  Точки, отнесённые к начальным центрам. Разбиение на плоскости — диаграмма Вороного относительно начальных центров. |
| https://upload.wikimedia.org/wikipedia/commons/thumb/3/3e/K_Means_Example_Step_3.svg/1024px-K_Means_Example_Step_3.svg.png  Вычисление новых центров кластеров (Ищется центр масс). | https://upload.wikimedia.org/wikipedia/commons/thumb/d/d2/K_Means_Example_Step_4.svg/1024px-K_Means_Example_Step_4.svg.png  Предыдущие шаги, за исключением первого, повторяются, пока алгоритм не сойдётся. |

Преимущества и недостатки алгоритма k-means:

Преимущества:

* Простота реализации
* Работа с огромным набором данных
* Быстрое обучение на новых примерах
* Поддержка сложных форм и размеров

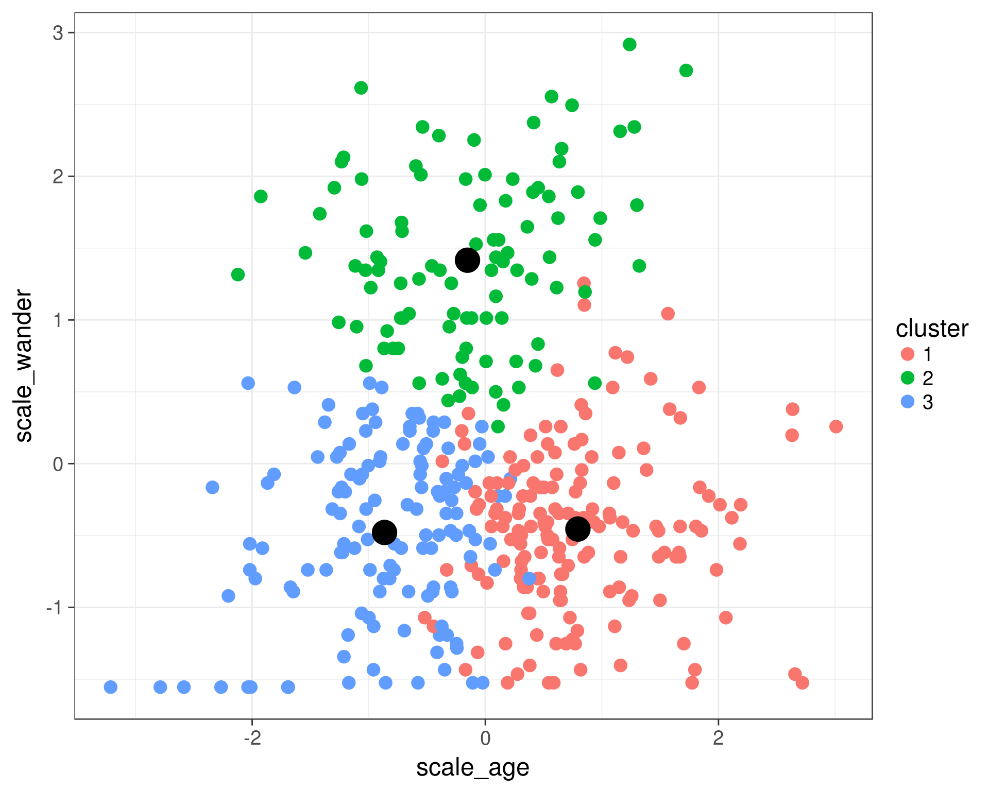
Недостатки:

* Не гарантируется достижение глобального минимума суммарного квадратичного отклонения J, а только одного из локальных минимумов.
* Результат зависит от начального выбора центров кластеров {X(0)k} , их оптимальный выбор неизвестен.
* Число кластеров k надо знать заранее.

Следующий алгоритм, который мы рассмотрим будет метод нечёткой кластеризации C-means. Он позволяет разбить имеющееся множество элементов мощностью {\displaystyle N}на заданное число нечётких множеств {\displaystyle k}k. Метод нечеткой кластеризации C-средних можно рассматривать как усовершенствованный метод k-средних, при котором для каждого элемента из рассматриваемого множества рассчитывается степень его принадлежности каждому из кластеров. Целью этого алгоритма является: распределить точки входного множества на кластеры так, чтобы средние точки разных кластеров различались как можно сильнее. Алгоритм k-means давал однозначный ответ, принадлежит ли какая-то точка тому или иному кластеру, но данный алгоритм позволяет одной точке лежать одновременно в двух или более кластерах. Степень принадлежности точки i кластеру j характеризуется величиной µij ∈ [0, 1]. Эта величина обладает следующим свойством для любой точки i: ∑ kj=0 µij = 1, где k - количество кластеров. Чтобы хранить эти значения для всех N точек, используется матрица µ с N строчками и k столбцами. Эта матрица называется матрицей распределения. Так же, как и в предыдущем алгоритме, для запуска требуется гипотеза о количестве кластеров k. Также нужно выбрать параметры ϵ > 0 и m > 1. Первый из них нужен, чтобы останавливаться в тот момент, когда изменение матрицы распределения между двумя итерациями становится незначительным. Второй параметр называется коэффициентом нечеткости и определяет то, насколько нечетким будет разбиение. Чем больше значение m, тем меньше будут величины µij . При m, близком к 1, результат разбиения становится похож на результат работы алгоритма k-means. Сам алгоритм c-means описан ниже:

1. Распределяем точки по кластерам случайным образом.
2. Для каждого кластера находим его центр, для этого считаем покомпонентное среднее всех относящихся к нему точек, причем каждая точка берется с весом, равным степени ее принадлежности к кластеру:
3. Пересчитываем матрицу распределения по следующей формуле:
4. Если на предыдущем шаге матрица распределения изменилась меньше, чем на ϵ, или если мы совершили максимально допустимое число итераций, прекращаем работу. Иначе переходим к пункту 2.
5. Результат работы алгоритма - текущее разбиение точек на кластеры.

Начальная инициализация производилась путем выбора случайных точек в качестве центров кластеров. Результат работы алгоритма представлен на картинке.



Алгоритм кластеризации DBSCAN является одним из наиболее часто используемых алгоритмов кластеризации, и наиболее часто упоминается в научной литературе.

Основной идеей алгоритма DBSCAN является представление объектов кластера в виде группы точек в метрическом пространстве, являющихся вершинами одного связного графа. Причём две точки в таком графе соединяются ребром только в том случае, если расстояние между ними в заданной метрике не превышает определённого расстояния. Если рядом с некоторым объектом нет достаточно близких соседей, то он признаётся выбросом.

Свойства алгоритма:

* Алгоритм находит заранее неизвестное число кластеров произвольной формы.
* Алгоритм работает в условиях зашумлённых данных, выделяя выбросы в отдельную категорию объектов.
* Алгоритм обладает сбалансированностью вычислительного процесса относительно всех типов операций при разбиении входных данных на части примерно одинакового размера.
* Алгоритм не является детерминированным, так как в некоторых случаях граничные точки могут попасть в несколько различных кластеров, что зависит от порядка их формирования. Однако это не оказывает значительного влияния на результаты работы алгоритма. Существует вариация алгоритма DBSCAN, которая относит все граничные точки к шуму, тем самым достигая полной детерминированности.
* Качество работы алгоритма сильно зависит от используемой метрики, а также от правильно выбранных параметров MinPts и ε для заданной предметной области. В частности, для объектов в пространстве большой размерности при использовании метрики Евклида имеет место проклятие размерности.
* Алгоритм плохо работает для разнородных данных, состоящих из кластеров разной плотности, так как тогда параметры алгоритмы MinPts и ε не могут быть выбраны оптимальным образом.

Абстрактно алгоритм можно представить, как последовательность следующих этапов:

* Найти точки в ℰ-окрестности каждой точки и определить основные точки с более чем minPts соседями
* Найти связные компоненты основных точек на графе соседей, игнорируя все неосновные точки.
* Назначить каждую неосновную точку ближайшему кластеру, если кластер является ℰ-соседним, в противном случае считаем точку шумом.

|  |  |
| --- | --- |
| https://miro.medium.com/max/1200/0*dcKggtgWiuTSKN7Y.png | На этой диаграмме . Точка A и другие красные точки являются основными точками, поскольку область с радиусом , окружающая эти точки, содержит по меньшей мере 4 точки (включая саму точку). |
| Поскольку все они достижимы друг из друга, точки образуют один кластер. Точки B и C основными не являются, но достижимы из A (через другие основные точки), и также принадлежат кластеру. Точка N является точкой шума, она не является ни основной точкой, ни доступной прямо. | |

Преимущества и недостатки алгоритма:

Преимущества:

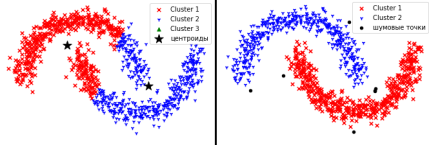
* DBSCAN не требует спецификации числа кластеров в данных априори в отличие от k-means.
* DBSCAN может найти кластеры произвольной формы. Он может найти даже кластеры полностью окруженные (но не связанные с) другими кластерами.
* DBSCAN имеет понятие шума и устойчив к выбросам.
* DBSCAN требует лишь двух параметров (minPts и ℰ) и большей частью нечувствителен к порядку точек в базе данных

Недостатки:

* DBSCAN не полностью однозначен – краевые точки, которые могут быть достигнуты из более чем одного кластера, могут принадлежать любому из этих кластеров, что зависит от порядка просмотра точек.
* Качество DBSCAN зависит от функции измерения расстояния. Наиболее часто используемой метрикой расстояний является евклидова метрика. В случае кластеризации данных высокой размерности эта метрика может оказаться почти бесполезной, что делает трудным делом нахождение подходящего значения ℰ
* DBSCAN не может хорошо разделить на кластеры наборы данных с большой разницей в плотности, поскольку не удается выбрать приемлемую для всех кластеров комбинацию minPts и ℰ

Для выбора наилучшего алгоритма рассмотрим примеры задач и различие работы алгоритмов K-means и DBSCAN в этой задаче:

Рассмотрим пример. Это более сложный пример, так как является линейно разделимым.



Как мы можем увидеть метод k-means не смог разделить данные верно. Это связано с тем, что идея алгоритма заключается в том, что необходимо находить точки вокруг центров. DBSCAN кластеризовал данные точно, определив несколько точек, как шумовые. В данном примере точки располагаются довольно близко, тем самым облегчая кластеризацию для этого метода

Исходя из всего вышесказанного можно сделать вывод, что для достижения наилучшего результата необходимо использовать алгоритм DBSCAN, так как он работает наиболее точнее, имеет понятие шума и устойчив к выбросам и требует лишь двух параметров (minPts и ℰ) и большей частью нечувствителен к порядку точек в базе данных

Список литературы

[1]. Воронцов К.В. Алгоритмы кластеризации и многомерного шкалирования. Курс лекций. МГУ, 2007.

[2]. Jain A., Murty M., Flynn P. Data Clustering: A Review. // ACM Computing Surveys. 1999. Vol. 31, no. 3.

[3]. Котов А., Красильников Н. Кластеризация данных. 2006.

[4]. Мандель И. Д. Кластерный анализ. — М.: Финансы и Статистика, 1988.