Занятия 1 и 2. Кластеризация

Анализ данных и машинное обучение

Гирдюк Дмитрий Викторович

31 марта 2021 г.

СПбГУ, ПМ–ПУ

Содержание

Введение в направление

Чем будем заниматься? [1]

Обучение без учителя (unsupervised learning) – это тип машинного обучения, который ищет ранее необнаруженные закономерности в наборе данных без ранее существовавших меток и с минимальным или полностью отсутствующим контролем человека.

Задача обучения без учителя покрывает не только *кластеризацию*, но и

- *поиск ассоциативных правил*
- *заполнение пропущенных значений*
- *поиск аномалий*
- *сокращение размерности и визуализация данных*

Постановка заадчи кластеризации

Постановка задачи

Кластерный анализ или *кластеризация* – это задача группировки набора объектов таким образом, чтобы объекты в одной группе (называемой кластером) были более похожи (в некотором смысле) друг на друга, чем на объекты в других группах (кластерах).

Проще говоря, имеем

- Пространство объектов X и выборку из него X^l
- Мера расстояния между объектами $\rho : X \times X \to R^+$

Хотим получить

- Множество групп/кластеров Y
- Алгоритм кластеризации $\alpha : X \to Y$

С какой целью используется

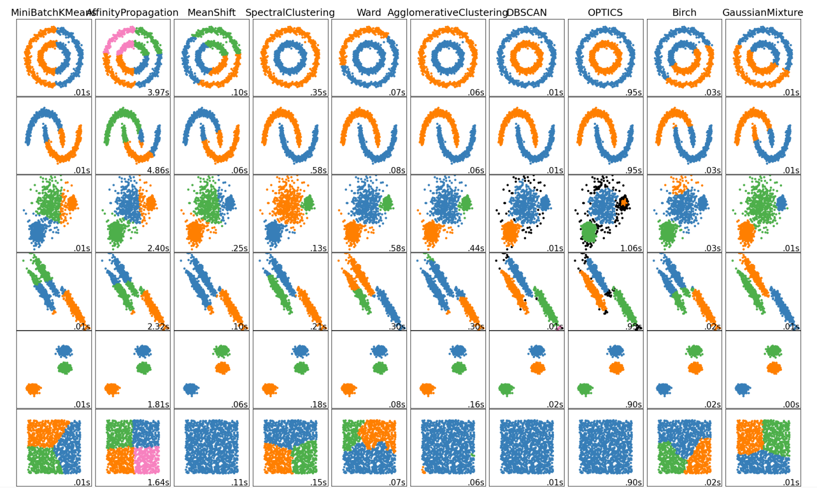
- Разделение на группы с целью упрощения работы (отдельные модели для каждой группы)
- Сокращение объемов наблюдений и сжатие данных (например, квантизация нейронных сетей)
- Выделение новизны/аномалий
- Построение иерархии/таксономии объектов

Некорректность постановки задачи

Сама по себе постановка задачи кластеризации некорректна, а именно

- не существует единого критерия качества (их, скорее, наоборот слишком много)
- число кластеров может быть заранее неизвестно
- сильная зависимость от метрики ρ

Иллюстрация проблем на примерах



И что с этим делать?

- Не полагаться на нее, если не существует альтернативных способов подтверждения адекватности ее результатов
- Ответственно подходить к предварительному изучению данных, отбору обучающей выборки и метрике

Требования к алгоритмам

- Масштабируемость
- Работа с большими размерностями
- Устойчивость к выбросам
- Устойчивость к различным типам кластерных структур
- Интерпретируемость результатов
- Временная сложность

Общая классификация алгоритмов

Классификация алгоритмов [2—4]

В большинстве источников выделяют пять групп алгоритмов

- *Основанные на центроидах* (centroid based): k-means, k-modes, k-medoids, **Meanshift**, FCM, Affiniy propagation
- *Иерархические* (hierarchical): **агломеративные** (Ward, single/average/complete linkage), BIRCH, на основе теории графов (выделение связных компонент и минимальное остовное дерево), **Spectral Clustering**, CURE, ROCK, Chameleon, Echidna, SNN, CACTUS, GRIDCLUST
- *Основанные на плотности* (density based): **DBSCAN**, OPTICS, DBCLASD, GDBSCAN, DENCLU, SUBCLU
- *Сеточные* (grid based): STING, Wave cluster, BANG, CLIQUE, OptiGrid, MAFIA, ENCLUS, PROCLUS, ORCLUS, FC, STIRR
- *Основанные на модели данных* (model based): **Expectation Maximization** (EM), COBWEB, CLASSIT, SOM

Основные алгоритмы

Meanshift [5]

- Meanshift – это алгоритм кластеризации, использующий *ядерную оценку плотности* (Kernel Density Estimation, KDE), который итеративно назначает наблюдения кластерам, сдвигая точки в сторону моды
- В отличие от (дешевого и сердитого) K-Means, Meanshift не требует заранее указывать количество кластеров, но требует задать параметр окна для KDE.

Meanshift: описание алгоритма

- Алгоритм итеративно назначает каждую точку ближайшему центроиду кластера.
- Направление к ближайшему центроиду скопления определяется тем, где находится большинство ближайших точек.
- Таким образом, на каждой итерации каждая точка данных будет перемещаться ближе к тому месту, где находится большинство точек, которое является или приведет к центру кластера.
- Когда алгоритм останавливается, каждой точке в соответствии ставится номер кластера

Meanshift: подробнее об алгоритме

- Ядровая оценка плотности

$$f_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

- Итеративное обновление точек

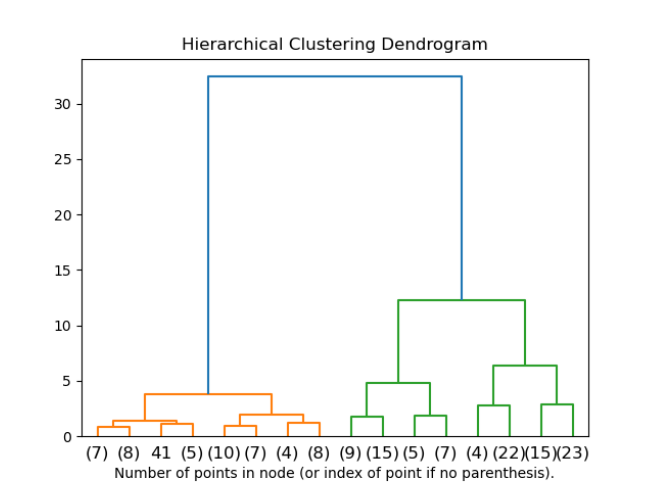
$$x_i^{t+1} = m(x_i^t)$$
$$m(x_i) = \frac{\sum_{x_j \in N(x_i)} K_h(x_j - x_i) x_j}{\sum_{x_j \in N(x_i)} K_h(x_j - x_i)}$$

- Обязательна стандартизация входных данных (привет, bandwidth)
- Есть реализация в scikit-learn'е. Распараллеливание; поддерживает бинаризацию исходных данных; можно даже не задавать h, но тогда страдает производительность; кроме брутфорса для поиска соседей есть KD- и Ball-деревья.

Агломеративная кластеризация

- Иерархическая кластеризация – это метод кластерного анализа, который направлен на построение иерархии кластеров
- Выделяют два подхода
    - Дивизимный (“снизу вверх”)
    - Агломеративный (“сверху вниз”)
- В большинстве реализаций слияния и разбиения происходят жадно
- Результаты иерархической кластеризации обычно представляются в виде дендрограммы.

Агломеративная кластеризация: пример дендрограммы



Агломеративная кластеризация: описание подхода

- Изначально каждая точка – отдельный кластер
- Объединение кластеров происходит путем поиска такой их пары, которая имеет наименьшее значение симметричной метрики близости между кластерами
- Процесс завершается, когда все сливается в один единственный кластер (или до фиксированного числа кластеров)
- Все, что остается, произвести “разрез” в получившейся иерархии кластеров

Агломеративная кластеризация: итеративный алгоритм Ланса-Уильямса

Algorithm 1: Алгоритм Ланса–Уильямса

input : Каждая точка – кластер $C_1 = \{\{x_1\}, \dots, \{x_n\}\}$,
подсчитаны расстояния между кластерами-точками
$R_{\{x_i\}\{x_i\}} = \rho(x_i, x_j)$

for $t \leftarrow 2$ **to** l **do**

  Найти в C_{t-1} пару кластеров U и V с минимальным R_{UV};

  $W = U \cup V$;

  $C_t = C_{t-1} \cup W \setminus \{U, V\}$;

  for $S \in C_t$ **do**

    $R_{WS} = \alpha_U R_{US} + \alpha_V R_{VS} + \beta R_{UV} + \gamma |R_{US} - R_{VS}|$;

Агломеративная кластеризация: частные случаи метрики схожести

- Расстояние ближайшего соседа/Single/Minimum:

$$R_{WS} = \min_{w \in W, s \in S} \rho(w, s),$$

$$\alpha_U = \alpha_V = \frac{1}{2}, \beta = 0, \gamma = -\frac{1}{2}$$

- Расстояние дальнего соседа/Complete/Maximum:

$$R_{WS} = \max_{w \in W, s \in S} \rho(w, s),$$

$$\alpha_U = \alpha_V = \frac{1}{2}, \beta = 0, \gamma = \frac{1}{2}$$

- Групповое среднее расстояние/Average:

$$R_{WS} = \frac{1}{|W||S|} \sum_{w \in W, s \in S} \rho(w, s),$$

$$\alpha_U = \frac{|U|}{|W|}, \alpha_V = \frac{|V|}{|W|}, \beta = \gamma = 0$$

Агломеративная кластеризация: частные случаи метрики схожести (ii)

- Расстояние между центроидами/Centroid:

$$R_{WS} = \rho^2 \left(\sum_{w \in W} \frac{w}{|W|}, \sum_{s \in S} \frac{s}{|S|} \right),$$

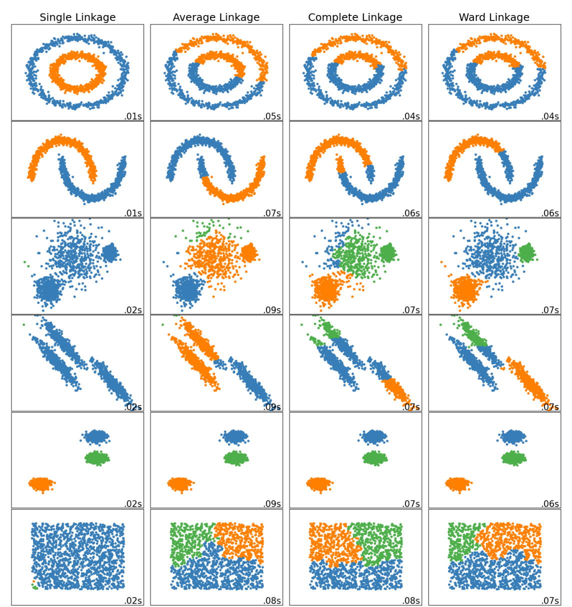
$$\alpha_U = \frac{|U|}{|W|}, \alpha_V = \frac{|V|}{|W|}, \beta = -\alpha_U \alpha_V, \gamma = 0$$

- Расстояние Уорда/Ward:

$$R_{WS} = \frac{|S||W|}{|S| + |W|} \rho^2 \left(\sum_{w \in W} \frac{w}{|W|}, \sum_{s \in S} \frac{s}{|S|} \right),$$

$$\alpha_U = \frac{|S| + |U|}{|S| + |W|}, \alpha_V = \frac{|S| + |V|}{|S| + |W|}, \beta = -\frac{|S|}{|S| + |W|}, \gamma = 0$$

Агломеративная кластеризация: пример дендрограммы

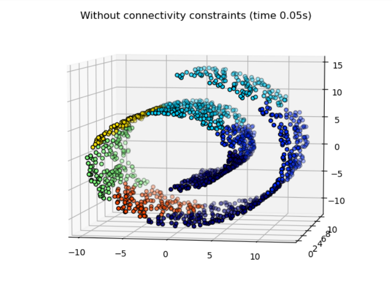


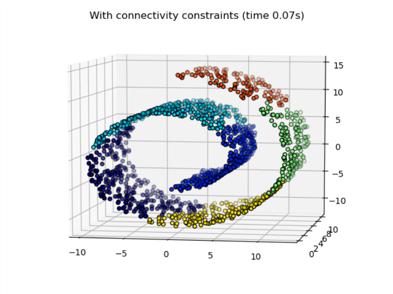
Агломеративная кластеризация: подробнее

- Подбор функции расстояния между точками, очевидно, имеет существенное влияние. Чаще всего выбор между l_2 и l_1
- Стандартизация тоже влияет. Но производить ли ее в случае иерахической кластеризации – вопрос спорный
- Метрика близости: обычно Ward. Ну и тут ограничение на использование только l_2. В таких случаях в первую очередь смотрите на Average (устойчивее к выбросам)
- Обычно количество кластеров определяют по дендограмме: "отсекают" там, где дельта метрики на двух итерациях имеет наибольшее значение

Агломеративная кластеризация: подробнее (ii)

- Чаще всего применяют в случае, когда данных мало, либо когда хочется построить иерахию/таксономию (ваш кэп)
- Имплементировано в scikit-learn'e. Из интересного: есть возможность докидывать ограничения на локальную структуру в данных



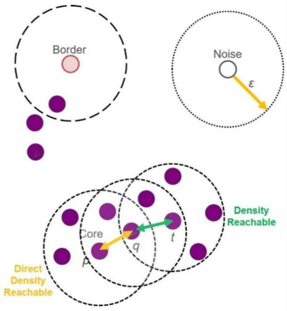


DBSCAN [6]

- Density-Based Spatial Clustering of Appliations with Noise (DBSCAN) – алгоритм кластеризации, основанный на плотности точек, изначально разработанный с целью кластеризации в базах данных, содержащих геометрические представления наблюдений
- Основными преимуществами алгоритма авторы выделили минимальную необходимость понимания предметной области данных при подборе гиперпараметров метода, а также способность обнаруживать кластеры произвольной формы
- Алгоритм достаточно прост, наряду с k-means один из самых популярных

DBSCAN: описание алгоритма

- Алгоритм имеет 2 гиперпараметра: величина окрестности точки ε и минимальное количество наблюдений в окрестности $MinPts$
- При кластеризации точка может быть причислено к 3 типам:
    - корневая: в его ε-окрестности не менее $MinPts$ точек
    - граничная: в его ε-окрестности меньше $MinPts$ точек, но среди них есть как минимум одна корневая
    - шумовая: не корневая и не граничная



DBSCAN: алгоритм

Algorithm 2: DBASCAN

input : Выборка $X = \{x_1, \ldots, x_n\}$, параметры ε и $MinPts$

$U = X, N = \emptyset, a = 0$;

while $U \neq \emptyset$ **do**

    Взять $x \in U$;

    if $|U_\varepsilon(x)| < MinPts$ **then**

        Пометить x как потененциально шумоваю точку;

    else

        $K = U_\varepsilon(x), a = a + 1$;

    for $x' \in K$ **do**

        if $|U_\varepsilon(x)| \geq MinPts$ **then**

            $K = K \cup U_\varepsilon(x')$;

        else

            пометить x' как граничную точку кластера K;

    foreach $x_i \in K$ **do** $a_i = a$;

    $U = U \setminus K$;

DBSCAN: комментарии i

- Подбор гиперпараметров.
  - Общая идея состоит в построении графика, по ординате у которого расстояние до $MinPts$-го соседа, а по абсциссе – точки, отсортированные в порядке увеличения этого расстояния.
  - Существенный скачок в значении идентифицирует выбросы, посему задавая некоторый процент на их число можно определить ε.
  - Обычно строят несколько таких графиков для различных значений $MinPts$.
  - В некоторых источниках значение $MinPts$ предлагают выбирать равным $\dim X + 1$, Где-то встречается $2 * \dim X$

DBSCAN: комментарии ii

- Есть реализация в sklearn'е. Кроме того, там же представлена модификация алгоритма под названием OPTICS [7], фактически отличающаяся от него тем, что задает интервал для значений ε, что позволяет выделять кластеры с различными плотностями

EM

- Expectation Maximization (EM) – итеративный алгоритм для поиска оценок максимального правдоподобия параметров вероятностных моделей, зависимых от некоторых скрытых переменных
- Алгоритм имеет приложения в дискриминатном анализе, кластеризации (разделение смеси распределений), восстановлении пропусков в данных, обработке сигналов и изображений

EM: теория

- Смесь распределений

$$p(x) = \sum_{j=1}^{k} w_j \varphi(x, \theta_j), \quad \sum_{j=1}^{k} w_j = 1, \quad w_j \geq 0$$

$\varphi(x, \theta_j) = p(x|j)$ – функция правдоподобия j-ой компоненты,

$w_j = P(j)$ – априорная вероятность j-ой компоненты

- Задача поиска максимума правдоподобия

$$L(w, \theta) = \ln \prod_{i=1}^{n} p(x_i) = \sum_{i=1}^{n} \ln \sum_{j=1}^{k} w_j \varphi(x_i, \theta_j) \longrightarrow \max_{w, \theta}$$

$$\text{w.r.t} \ \sum_{j=1}^{k} w_j = 1, \quad w_j \geq 0$$

EM: теория ii

- При разделении смеси расспределений (обычно, гауссовских, отсюда Gaussian Mixtures Model, GMM-EM) EM используется следующим образом [8]:
    - (expectation) вводится вспомогательный вектор скрытых переменных g, такой что его значения могут быть вычислены, зная параметры распределений θ
    - (maximization) вычислив значения скрытых переменных, задача поиска (локального) максимума правдоподобия существенно упрощается

EM: теория iii

- В качестве скрытых переменных выберем

$$g = \{g_{ij}\}, \quad g_{ij} = P(j|x_i)$$

- По формуле Байеса

$$g_{ij} = \frac{w_j \varphi(x_i, \theta_j)}{\sum_{s=1}^{k} w_s \varphi(x_i, \vartheta_s)} \tag{1}$$

- Формулы для θ_j и w_j выводятся из условий Куна–Такера

$$\theta_j = \arg\max_{\theta_s} \sum_{i=1}^{n} g_{ij} \ln \varphi(x_i, \theta_s) \tag{2}$$

$$w_j = \frac{1}{n} \sum_{i=1}^{n} g_{ij} \tag{3}$$

EM: алгоритм

Algorithm 3: EM

input : Выборка $X = \{x_1, \ldots, x_n\}$, начальные приближения для (w, θ), количество кластеров k и ε

while *True* **do**

E: **foreach** $i = 1, \ldots, n, \quad j = 1, \ldots, k$ **do** $g_{ij}^0 = g_{ij}$, вычислить новые значения g_{ij} по формуле (1);

M: **foreach** $j = 1, \ldots, k$ **do** решить задачу (2) для нахождения θ_j, вычислить новые значения w_j по формуле (3);

if $\max_{i,j} |g_{ij} - g_{ij}^0| < \varepsilon$ **then**

Завершить работу

EM: комментарии i

- В случае гауссовских плотностей при условии, что признаки независимы (матрица ковариаций диагональна), задача (2) имеет аналитическое решение
- Кластеризация мягкая, т.е. точкам не ставится в соответствии номер кластера, а лишь вероятность принадлежности. Нужен номер – берите максимум по g_{ij}
- k-means – частный случай EM. На E-этапе вычисляются скрытые переменные – номера кластеров, а на M-этапе происходит обновление центров.
- Начальное приближение: рандом/k-means

EM: комментарии ii

- У EM'а есть масса модификаций: generalized EM (GEM), stochastic EM (SEM), есть вариант с последовательным добавлением компонент
- GaussianMixture в scikit-learn'е. Количество компонент обязательно. Поддерживает различные типы ковариационных матриц

Spectral Clustering [9]

- Spectral Clustering – двухэтапный алгоритм кластеризации, который на первом этапе находит некоторое количество собственных векторов *лаплассиана графа*, образованного из входных наблюдений (фактически процедура снижения размерности), а затем применяет (обычно) k-means к составленной из них [собственных векторов] матрице
- В отличие от алгоритмов, использующих сферические/эллиптические метрики расстояний, умеет выделять существенно невыпуклые кластеры

Spectral Clustering: основная теория

- Предварительный этап состоит в формировании из исходной матрицы с наблюдениями X ненаправленный граф похожести (similarity graph). Варианты следующие:
    - Полностью связанный
    - ε-окрестность
    - k-ближайших соседей: если вершина v_i находится среди k-ближайших соседей вершины v_j
- Элементы s_{ij} матрицы схожести графа неотрицательны и равны 0, когда объекты совершенно не похожи (не связаны, нет ребра в графе)
- Обычно матрица формируется на основе радиально-базисных функций: чаще всего Гауссовская $\varphi(x) = \exp(\frac{-d_{ij}^2}{\sigma^2})$

Spectral Clustering: основная теория ii

- Пусть W есть матрица весов (взвешенная матрица смежности), такая что $w_{ij} = s_{ij}$, если существует ребро между вершинами v_i и v_j, и 0 иначе
- Матрица D (degree matrix) есть диагональная матрица с элементами $d_i = \sum_{j=1}^{n} w_{ij}$

Spectral Clustering: основная теория iii

- Лапласианом графа называется матрица $L = D - W$ с набором интересных свойств
  - $z^\intercal L z = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(z_i - z_j)^2$ для любого $z \in R^n$
  - L симметрична и положительно полуопределена
  - У L n неотрицательных собственных чисел: $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$
  - Наименьшее собственное число λ_1 всегда равно 0. Соответствующий ему собственный вектор состоит из 1, если граф связный. Если у графа есть p компонент связанности, то кратность λ_1 равна p, а собственные векторы представляют собой индикаторные векторы

Spectral Clustering: алгоритм

Algorithm 4: Spectral Clustering

input : Матрица схожести $S \in R^{n \times n}$ и количество кластеров k

Построить граф схожести. Пусть W есть его матрица смежности;

Вычислить лапласиан L;

Найти первые k собственных векторов $u_1, \ldots, u_k$. Составить из них матрицу $U \in R^{n \times k}$;

k-means(U, k);

Spectral Clustering: комментарии i i

- Подробнее о том, почему это действительно работает, в статье [9]. Спектральная кластеризация по сути является релаксацией задачи о разделении графа на компоненты
- Но нужна грамотная настройка: тут и выбор матрицы схожести, и количества кластеров. Для данного алгоритма есть эвристика для поиска оптимального числа кластеров: построить график собственных векторов по возрастанию и “обрезать” там, где происходит существенный скачок в их значениях

Spectral Clustering: комментарии i ii

- Что касается поиска собственных векторов, то существует масса численных методов, обернутых в фрэймворки. Например, «arpack», «eigen». sklearn подтягивает методы линейной алгебры из scipy – подробнее в их доках. С простейшим итеративным методом поиска собственных векторов познакомимся в следующий раз, когда будет говорить о методе главных компонент (PCA)
- Лапласиан не единственный. Опять же, в работе [9] приводятся "нормализованные" альтернативы
- Есть в scikit-learn'e. n_components не про количество связанных компонент, а про число собственных векторов, используемых на втором этапе алгоритма (по умолчанию n_components = n_clusters)

Общая характеристика 5 подходов

Сравнение подходов: Centroid based

Преимущества:

- Простые, надежные, обычно масштабируемые
- Не требуют глубокого понимания специфики данных
- Итеративные (пересчитали центроиды и разделение на кластеры изменилось)

Недостатки

- Число кластеров?
- Нормализация/стандартизация данных может существенно изменить расклад

Сравнение подходов: Hierarchical

Преимущества:

- Изначально не требуют фиксации количества кластеров
- Зачастую никаких гиперпараметров
- Просты для понимания и имплементации

Недостатки

- Трудности с интерпретацией
- Чувствительны к выбросам
- Отсутствует итеративность (как только объект отнесен к кластеру, он из него уже не выйдет)

Сравнение подходов: Density based

Преимущества:

- Умеют в кластеры произвольной формы
- Весьма устойчивы к выбросам

Недостатки

- Трудности с очень большими и разреженными данными
- Проклятие размерности

Сравнение подходов: Grid based

Преимущества:

- Разбивают пространство данных на ячейки
- Хороши для датасетов больших размерностей, итеративные

Недостатки

- Плохо работают на данных, являющихся отображением датасета меньшей размерности в пространство большей размерности
- Требуют тщательного выбора проекций, функции оценки плотности и оптимальных разделяющих поверхностей

Сравнение подходов: Model based

Преимущества:

- Полученное разбиение интерпретируемо со статистической точки зрения
- Успешно используются в векторной квантизации [10]
- Позволяют выбирать плотности распределения компонент

Недостатки

- Параметры должны быть оценены, что требует большего количества точек данных в каждой компоненте
- Трудно настраивать

Метрики качества

Общая классификация [11]

- Разделение простое: либо работа с известной правильной разметкой, либо попытка понять, насколько хорошо произошло разбиение
- К первым относятся Adjusted Rand index, Mutual Information based scores, Homogeneity, completeness and V-measure
- Ко вторым Silhouette, Calinski-Harabasz Index, Davies-Bouldin Index и другие

Adjusted Rand index [12]

- Здесь и далее обозначим через C истинное разбиение данных, а через K – разбиение на основе какого-либо алгоритма
- Пусть a и d есть общее количество пар наблюдений, которые находятся в одном/разных кластерах как в C, так и в K
- Rand Index и Adjusted Rand index

$$\text{RI} = 2\frac{a+d}{n(n-1)} = \frac{a+d}{a+b+c+d}$$

$$\text{ARI} = \frac{\text{RI} - E[\text{RI}]}{\max(\text{RI}) - E[\text{RI}]} =$$
$$= \frac{(a+b+c+d)(a+d) - [(a+b)(a+c)+(b+d)(c+d)]}{(a+b+c+d)^2 - [(a+b)(a+c)+(b+d)(c+d)]}$$

- Очевидно, чем больше, тем лучше. Значения RI в интервале от 0 до 1, ARI – [-1; 1]. Если разметка рандомная, то ARI будет близок к 0. Никаких предположений о структуре кластеров – можно сравнивать результаты разных алгоритмов

Mutual Information based scores [13]

- Энтропия C/K

$$H(C) = -\sum_{i=1}^{|C|} P^C(i)\log(P(i)), \quad P^C(i) = \frac{|C_i|}{n}$$

- Взаимная информация (Mutual information)

$$\mathsf{MI}(C,K) = \sum_{i=1}^{|C|}\sum_{j=1}^{|K|} P(i,j)\log\left(\frac{P(i,j)}{P^C(i)P^K(j)}\right), \quad P(i,j) = \frac{|C_i \cap K_j|}{n}$$

- Ее нормализованная версия

$$\mathsf{NMI}(U,V) = \frac{\mathsf{MI}(C,K)}{\mathsf{mean}(H(C),H(K))}$$

Mutual Information based scores ii

- И скорректированная нормализованная версия

$$\mathsf{AMI}(U,V) = \frac{\mathsf{MI} - E[\mathsf{MI}]}{\mathsf{mean}(H(C), H(K)) - E[\mathsf{MI}]}$$

где

$$E[\mathsf{MI}(U,V)] = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|}\sum_{n_{ij}=(a_i+b_j-N)^+}^{\min(a_i,b_j)} \frac{n_{ij}}{N}\log\left(\frac{Nn_{ij}}{a_ib_j}\right)\times$$
$$\times\frac{a_i!b_j!(N-a_i)!(N-b_j)!}{N!n_{ij}!(a_i-n_{ij})!(b_j-n_{ij})!(N-a_i-b_j+n_{ij})!}$$

Mutual Information based scores iii

- Чем больше, тем лучше
- MI, NMI лежат в интервале $[0; 1]$. При произвольной разметке не факт что будут близки к 0
- AMI это исправляет, его значения лежат в интервале от -1 до 1

Homogeneity и completeness

- Homogenity (однородность) – каждый кластер из K содержит наблюдения только из одного класса C

$$h = 1 - \frac{H(C|K)}{H(C)}$$

- Completness (полнота) – все наблюдения из класса C находятся в каком-либо кластере K

$$c = 1 - \frac{H(K|C)}{H(K)}$$

- Условная энтропия

$$H(C|K) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|} \frac{n_{c,k}}{n} \log\left(\frac{n_{c,k}}{n_k}\right)$$

где $n_{c,k}$ есть число наблюдений из класса c в кластере k, а n_k – общее число наблюдений в кластере k

V-мера

- V-мера – их гармоническое среднее

$$v_\beta = \frac{(1+\beta)hc}{\beta h + c}$$

- Значения всех трех метрик в интервале [0; 1]. Чем больше, тем лучше.
- Никаких предположений о структуре кластеров – можно сравнивать результаты разных алгоритмов кластеризации
- Та же проблема с нормализацией. Если мало наблюдений / много кластеров, рекомендуется использовать ARI

Silhouette

- Пусть a и b есть среднее расстояние между наблюдением и всеми другими точками в том же кластере/в следующем ближайшем кластере
- Коэффициент силуэта для наблюдения есть

$$s = \frac{b - a}{max(a, b)}$$

- Для выборки коэффициент силуэта задается среднее значение коэффициента для каждого наблюдения
- Значения в интервале [-1; 1]. Чем больше, тем лучше. Если коэффициент близок к 0, то это свидетельство в сторону того, что кластеры "накладываются" друг на друга

Calinski-Harabasz Index (Variance Ratio Criterion)

- Индекс Калинского–Харабэсза определяется как отношение между средней межкластерной дисперсией и средней дисперсией внутри кластеров

$$s = \frac{\mathrm{tr}(B)}{k-1} / \frac{\mathrm{tr}(W)}{n-k}$$

- Тут k – число кластеров, а матрицы B и W имеют вид

$$W = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^\mathsf{T}, \quad B = \sum_{q=1}^{k} n_q (c_q - c_X)(c_q - c_E)^\mathsf{T}$$

  где C_q – наблюдения из кластера q, c_q – центр кластера q, c_X – центр всего набора данных X, а n_q – число наблюдений в кластере q
- Значение индекса тем больше, чем разделеннее кластеры и чем более сгруппированы в кластерах наблюдения. Ну и применять для случая сферических/эллиптических кластеров

Davies-Bouldin Index

- Индекс Дэвиса–Болдина оценивает среднее “сходство” между кластерами, где сходство есть мера, которая сравнивает расстояние между кластерами с размером самих кластеров
- Пусть s_i есть среднее расстояние между каждой точкой кластера i и центроида этого кластера, а d_{ij} есть расстояние между центроидами кластеров. Определим схожесть между кластерами следующим образом

$$R_{ij} = \frac{s_i + s_j}{d_{ij}}$$

- Тогда индекс имеет вид

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij}$$

- Чем индекс меньше, тем лучше. Близок к 0 – отличное разделение. Недостаток тот же, что и у индекса Калинского–Харабэсза

Кратко о выборе числа кластеров

Elbow method

- Вполне естественно полагать, что при хорошей кластеризации (в простом случае сферических/эллиптических кластеров) общее внутрикластерное расстояние

$$W(K) = \sum_{i=1}^{k} \sum_{j \in K_k} ||x_j - \mu_i||_2 \longrightarrow \min_K$$

  должно быть как можно меньше
- В то же время, минимум метрики достигается тогда, когда каждое наблюдение формирует отдельный кластер
- Идея метода локтя проста: смотрим на график метрики для различного количества кластеров и выбираем то, после которого увеличение числа кластеров не дает существенного снижения значения метрики
- Формализуя, правило следующее:

$$D(k) = \frac{|W(K_k) - W(K_{k+1})|}{|W(K_{k-1}) - W(K_k)|} \longrightarrow \min_k$$

Выбор числа кластеров

- Рассмотренные ранее коэффициенты Silhouette, Calinski-Harabasz Index, Davies-Bouldin Index также можно использовать для определения числа кластеров
- Всем желающим подробнее изучить эту тему предлагаю взглянуть на библиотеку “NbClust” в R [14]. В ней представлено более 30 различных коэффициентов

Использованные источники i

1. *Гирдюк Д.* Репозиторий с материалами для занятий. URL: https://github.com/dmgirdyuk/PtW_ML_Unsupervised_learning.
2. A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis. /. A. Fahad [и др.] // IEEE Transactions on Emerging Topics in Computing. 2014. Т. 2, № 3. С. 267—279.
3. *Tiruveedhula S.*, *Rani C.*, *Narayana V.* A Survey on Clustering Techniques for Big Data Mining. // Indian Journal of Science and Technology. 2016. Февр. Т. 9. С. 1—12. DOI: 10.17485/ijst/2016/v9i3/75971.

4. *Benabdellah A. C.*, *Benghabrit A.*, *Bouhaddou I.* A survey of clustering algorithms for an industrial context. // Procedia Computer Science. 2019. Т. 148. С. 291—302. ISSN 1877-0509. DOI: https://doi.org/10.1016/j.procs.2019.01.022. URL: http://www.sciencedirect.com/science/article/pii/S1877050919300225.
5. *Comaniciu D.*, *Meer P.* Mean shift: a robust approach toward feature space analysis. // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2002. Т. 24, № 5. С. 603—619.
6. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. /. M. Ester [и др.] // KDD. 1996.

7. OPTICS: Ordering Points to Identify the Clustering Structure. /. M. Ankerst [и др.] // Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. Philadelphia, Pennsylvania, USA : Association for Computing Machinery, 1999. С. 49—60. (SIGMOD '99). ISBN 1581130848. URL: https://doi.org/10.1145/304182.304187.

8. *Воронцов К.* Презентация по разделению смеси распределений из курса лекций Воронцова К.В. URL: http://www.machinelearning.ru/wiki/images/a/a9/Voron-ML-BTC-EM-slides.pdf.

9. *Luxburg U. V.* A tutorial on spectral clustering. // Statistics and Computing. 2007. Т. 17. С. 395—416.

10. *Hastie T.*, *Tibshirani R.*, *Friedman J.* The elements of statistical learning: data mining, inference and prediction. 2-е изд. Springer, 2009. URL: http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

11. Кластеризация в scikit-learn. URL: https://scikit-learn.org/stable/modules/clustering.html.

12. *Steinley D.* Properties of the Hubert-Arabie adjusted Rand index. // Psychol Methods. 2004. Т. 9, № 3. DOI: 10.1037/1082-989X.9.3.386.

13. *Vinh N. X.*, *Epps J.*, *Bailey J.* Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. // Journal of Machine Learning Research. 2010. Т. 11, № 95. С. 2837—2854. URL: http://jmlr.org/papers/v11/vinh10a.html.

14. NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. /. M. Charrad [и др.] // Journal of Statistical Software. 2014. Т. 61, № 6. С. 1—36. URL: http://www.jstatsoft.org/v61/i06/.