УДК 519.176

АЛГОРИТМ КЛАСТЕРИЗАЦИИ НА ОСНОВЕ МЕТОДОВ КОМБИНАТОРНОЙ ОПТИМИЗАЦИИ

М. И. Меркушев

Белорусский государственный университет,
пр. Независимости, 4, 220030, г. Минск, Беларусь, merkouchevmark@yandex.ru
Научный руководитель: О. А. Лаврова, кандидат физико-математических наук, доцент

В работе исследуется алгоритм кластеризации, разработанный на основе аппроксимационного алгоритма Гойманса-Уильямсона поиска максимального разреза графа. Алгоритм реализован в среде Wolfram Mathematica и на Python, протестирован на примерах. На основе тестовых вычислений показано, что исследуемый алгоритм сопоставим по точности с базовыми алгоритмами кластеризации, но является более медленным.

Ключевые слова: алгоритм Гойманса-Уильямсона; алгоритмы кластеризации; Wolfram Mathematica; Python.

Постановка задачи

Задача кластеризации состоит в разбиении множества объектов на подмножества (кластеры) таким образом, чтобы объекты из одного кластера были более похожи друг на друга, чем на объекты из других кластеров по какому-либо критерию.

Основная идея данной работы заключается в решении задачи кластеризации с применением результатов из теории графов и комбинаторной оптимизации. Если для некоторых точек в пространстве построить полный граф, соединяющий попарно эти точки, где веса ребер равны расстояниям между ними, мы сможем применить алгоритм поиска максимального разреза для разбиения множества точек на два максимально удаленных друг от друга кластера. При чем теория графов не ограничивает нас размерностью пространства, где находятся точки. А это значит, что количество признаков для объектов множества может быть произвольным. Соответственно, если перевести все признаки каких-либо данных в числовые величины и представить каждый из этих признаков, как координату в пространстве некоторой точки, можно решать задачу бинарной кластеризации, то есть разделения данных на два кластера.

Стоит отметить, что задача поиска максимального разреза графа относится к классу NP-полных [1]. Это означает, что при точном ее решении нельзя рассчитывать на получение быстродействующего алгоритма. Эта проблема решается созданием аппроксимационных алгоритмов, позволяющих получить приближенное решение задачи поиска максимального разреза графа за полиномиальное время.

Алгоритм Гойманса-Уильямсона

Задача поиска максимального разреза представлена задачей оптимизации вида

$$\mu = max\{\frac{1}{2}\sum_{1 \le i < j \le n} w_{ij} \cdot (1 - x_i x_j) : x_k \in \{-1,1\} \quad \forall k \in [n]\}, \qquad (1)$$

где w_{ij} это веса ребер между вершинами i и j, x_i и x_j – это метки, присваиваемые каждой вершине, n – количество вершин графа. Алгоритм Гойманса-Уильямса [1] позволяет найти приближенное решение задачи поиска максимального разреза при помощи сферической релаксации. Для сферической релаксации определяется функция

$$y(\xi) := (\xi, 0, \dots, 0) \in S^n \text{ для } \xi \in \{-1, +1\}, \qquad (2)$$

где S^n это единичная сфера в R^n. Это позволяет переписать задачу (1) в виде:

$$\mu = max\{\frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\cdot(1-< y(x_i), y(x_j) >: x_k \in \{-1,1\} \quad \forall k \in [n]\}, \qquad (3)$$

где вместо произведения $x_i x_j$ вычисляется скалярное произведение единичных векторов $< y(x_i), y(x_j) >$. Такая задача является сложной для решения, поэтому осуществляется релаксация: вместо x_k со значениями только -1 или 1 неизвестными являются точки, принадлежащий единичной сфере.

$$\mu \le max\{\frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\cdot(1-< y(i), y(j) >: y(k) \in S^n \quad \forall k \in [n]\}, \qquad (4)$$

где μ является приближением (оценкой) значения максимального разреза. Поиск данной оценки будет представлять из себя задачу полуопределенного программирования в виде:

$$\mu \le max\{\frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\cdot(1-y_{ij}) : y_{kk} = 1\ \forall k \in [n], Y = (y_{ij}) \in S_+^n\}, \qquad (5)$$

где S_+^n – это множество положительно-полуопределенных матриц размера $n \times n$.

Таким образом, поставленную задачу (5) можно решить за полиномиальное время, например, при помощи метода внутренней точки и, при этом, с высокой точностью. Для получения значений x_i и x_j из матрицы Y используется разложение Холецкого:

$$< y\ (i), y\ (j) > = y_{ij}, \qquad (6)$$

а затем округление вектора $(y(i))$ до скаляра (x_i), которое является основной идеей Гойманса и Уильямса [1]. В результате чего получается набор меток, как результат приближения максимального разреза. При этом точность полученного приближения составит не менее 0.875 от точного решения [1].

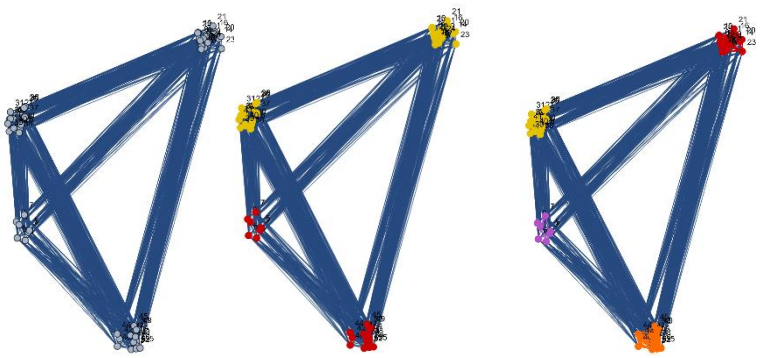
Реализация в Wolfram Mathematica

Прототип исследуемого алгоритма реализован на Wolfram Mathematica в виде пользовательской функции maxCutRelaxed[graph, trials], которая получает на вход объект graph, встроенного класса Graph, и количество попыток получения максимального разреза trials. Таким образом, алгоритм отрабатывает сразу несколько раз и выбирается разрез с максимальным значением суммарного веса попавших в разрез ребер.

Для проверки работы алгоритма был создан тестовый пример: генерируются случайно 4 группы точек, визуально расположенные в разных кластерах (рис. 1а). Количество самих точек задается также случайно.

Результат применения функции maxCutRelaxed к сгенерированным данным представлен на рис. 1б.

Для получения более двух кластеров можно применять рекурсивную кластеризацию [2]. Иными словами, данное действие можно повторять с каждым из полученных кластеров, пока не получим подходящее число подмножеств точек. При применении рекурсивной кластеризации получен следующий результат (рис. 1в).



а б в

Рис. 1. Тестовые данные (а), разбитые (б) – на 2 кластера, (в) – на 4 кластера

Недостатком реализации алгоритма в Wolfram Mathematica является низкая скорость его выполнения, связанная со временем расчета всех попарных расстояний между вершинами графа, количество которых для полного графа $N = \frac{n(n-1)}{2}$, где n – количество вершин графа.

Реализация в Python

Для сравнения исследуемого алгоритма с классическими алгоритмами кластеризации и классификации (k-means, c-means, метод опорных векторов, дерево решений и др.), была осуществлена реализация исследуемого алгоритма на Python с использованием модуля cvxgr [3]. Функция модуля algorithms.goemans_williamson_weighted(Graph) позволяет найти максимальный разрез при помощи алгоритма Гойманса-Уильямсона для объекта Graph, который реализован в виде объекта модуля networkx [4].

Алгоритм тестируется на примере из предыдущего раздела. Результаты реализаций в Wolfram Mathematica и на Python совпадают. Время выполнения алгоритма на Python составило около 5 минут.

Для дальнейшего тестирования исследуемого алгоритма используется известная задача бинарной классификации для сравнения точности и быстродействия алгоритма с классическими алгоритмами кластеризации.

Тестовые вычисления

Набор данных Titanic – Machine Learning from Disaster [5] содержит информацию о 418 пассажирах Титаника, для которых известны по 11 параметров. По заданной информации необходимо определить выжили ли пассажиры во время крушения корабля или нет, то есть решить задачу бинарной классификации. Для этого все данные о пассажирах переводятся в числовые величины при помощи библиотек pandas, numpy, sklearn. Полученные числовые величины определяют координаты точек графа, между которыми проводятся ребра с весами, равными евклидову расстоянию между соответствующими точками.

После понижения размерности (для визуализации) при помощи метода главных компонент (модуль sklearn.decomposition.PCA [6]) и применения алгоритма Гойманса-Уильямсона граф, представляющий набор данных, изображен на рис. 2.

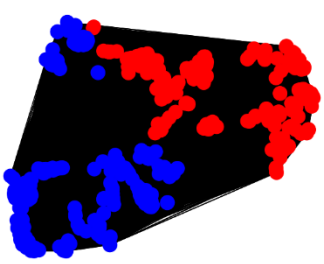


Рис. 2. Результат кластеризации набора данных Titanic исследуемым алгоритмом

Разрез был получен за 841 секунду (примерно за 14 минут), что существенно превышает время выполнения алгоритма кластеризации k-means. Алгоритм k-means для бинарной кластеризации выполняется за несколько секунд, результат кластеризации представлен на рис. 3.

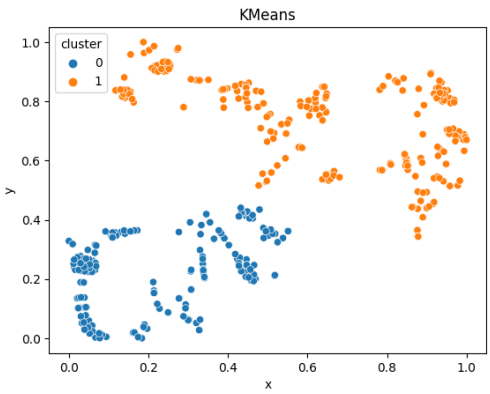


Рис. 3. Результат кластеризации набора данных Titanic алгоритмом k-means

Чтобы сравнить исследуемый алгоритм по точности, результаты кластеризации были выложены на kaggle [4]. Доля правильно определенных объектов кластера для алгоритма k-means составила 0.67224. Для исследуемого алгоритма оценка незначительно выше 0.67464, что говорит о сравнимой эффективности обоих алгоритмов. Однако, для алгоритмов кластеризации random forest, gradient boosting и других, перечисленных ранее алгоритмов, реализованных в модуле sklearn, средним результатом точности является оценка порядка 0.74. Можно сделать вывод о сопоставимости по точности исследуемого алгоритма с классическими алгоритмами кластеризации.

Вывод

Исследуемый алгоритм кластеризации имеет ряд преимуществ и недостатков. С одной стороны, он позволяет находить оптимальное разбиение реальных данных на два кластера по некоторому критерию сходства, используя эффективный алгоритм комбинаторной и полуопределенной оптимизации. С другой стороны, исследуемый алгоритм кластеризации имеет ряд ограничений, таких, как приближенное построение решения, низкая скорость выполнения алгоритма, связанная с расчетом начальных данных, а также сложность обобщения алгоритма для количества кластеров, большего двух. Указанные ограничения являются основаниями для дальнейшего анализа и улучшения алгоритма.

Библиографические ссылки

1. The Goemans-Williamson Algorithm [Электронный ресурс] / Simon Fraser University. URL: https://www.sfu.ca/~mdevos/notes/semidef/GW.pdf (дата обращения: 20.03.2024).

2. Blum A, Hopcroft J., Kannan R. Foundations of Data Science / 4.01.2018. P. 229 [Электронный ресурс]. URL: https://www.cs.cornell.edu/jeh/book.pdf (дата обращения: 23.10.2023).

3. Mehta H. Cvxgraphalgs [Электронный ресурс]. URL: https://pypi.org/project/cvxgraphalgs/ (дата обращения: 20.03.2024).

4. Titanic Machine Learning from Disaster [Электронный ресурс] / Kaggle. URL: https://www.kaggle.com/competitions/titanic/ (дата обращения: 20.03.2024).

5. NetworkX [Электронный ресурс]. URL: https://networkx.org/documentation/stable/index.html (дата обращения: 20.03.2024).

6. Scikit-learn [Электронный ресурс]. URL: https://scikit-learn.org/stable/ (дата обращения: 20.03.2024).