

Сравнение двух методов кластеризации, основанных на марковских цепях

Роман Гученко

19 сентября 2020 г.

Марковская цепь описывает такой процесс случайного блуждания на графе, который не помнит прошлого и на каждой итерации «прыгает» из узла в узел согласно фиксированной матрице переходных вероятностей¹. У марковских цепей много приложений. В частности, их можно приспособить для решения задачи кластеризации. Мы рассмотрим два метода.

Метод первый. Более очевидный. Вычисляем матрицу M попарных similarity² между объектами³. Выбираем минимальное similarity α , после превышения которого объекты будем считать похожими; что больше α в матрице заменяем единицами, что меньше — нулями. Далее нормируем матрицу по столбцам, чтобы она стала матрицей переходных вероятностей для некоторой марковской цепи. Нам нужна предельная матрица переходных вероятностей⁴. Полагаем $M_0 = M$. Вычисляем $M_i = M_{i-1}M$ до сходимости, которая наступает тогда, когда M_i распадается⁵ на набор стоящих на диагонали подматриц, состоящих из одинаковых столбцов, где каждая из подматриц соответствует своей изолированной компоненте связности в графе марковской цепи⁶. По каждой из таких подматриц строим кластер.

¹Подробнее смотри в [1].

²Посмотрите в поиске, например, про Jaccard similarity.

³Ну, или матрицу попарных расстояний, а потом переводим ее в similarity matrix, например, с помощью преобразования $1/(1 + \rho(x_1, x_2))$.

⁴На самом деле не обязательно нужна предельная. Может хватить и матрицы переходных вероятностей через несколько шагов. Это зависит от задачи. Мы просто хотим связать цепочки объектов A , B , C по транзитивности, которой может не быть в исходной матрице M . То есть если A похож на B , а B на C , но A не похож на C , мы все равно хотим, чтобы они попали в один кластер.

⁵Возможно, после перестановки столбцов.

⁶Для каждой компоненты связности мы получаем стационарное распределение в одинаковых столбцах соответствующей подматрицы.

Метод второй. Менее очевидный. Рассмотрен в диссертации [2]⁷. Вычисляем similarity matrix, нормируем ее по столбцам. Далее по очереди делаем следующее: умножаем матрицу саму на себя⁸, возводим все элементы в заданную степень⁹ (которую автор алгоритма назвал inflation параметром) и снова нормируем по столбцам. И так до сходимости, пока матрица не станет идемпотентной. Далее убираем все нулевые и одинаковые строчки из итоговой матрицы, а по оставшимся строчкам строим кластеры по ненулевым элементам в них.

Оба метода зависят по сути от одного параметра¹⁰, определяющего размер итоговых кластеров, и в целом очень похожи. Цель курсовой — разобраться с обоими и провести их повсеместное сравнение.

Теперь по баллам:

- Разобравшись в теории, реализовав оба алгоритма с использованием dense матриц, протестировав их на нескольких небольших наборах модельных данных¹¹ при разных параметрах и сделав выводы об их возможностях, особенностях и разнице в поведении, вы принесли 40/20 очков Гриффиндору в случае 1/2 курса соответственно.
- Разобрались со sparse матрицами? Поняли, что в первом алгоритме можно не дожидаться сходимости к стационарным распределениям для компонент связности и придумали другое условие остановки¹²? Попробовали все это на настоящих данных¹³? Значит, заработали еще 60/30 очков сверху (по 20/10 за каждый вопрос).

⁷Также есть смысл посмотреть презентацию по ссылке: https://sites.cs.ucsb.edu/~xuan/classes/CS595D-2009winter/MCL_Presentation2.pdf.

⁸То есть реализуем один шаг случайного блуждания на графе с текущей матрицей переходных вероятностей.

⁹Зачем это делается? С этим предстоит разобраться, прочитав текст диссертации.

¹⁰В первом методе это α , во втором — inflation.

¹¹Которые будут выданы.

¹²Или вообще другой алгоритм?

¹³Например, вот на этих, рассмотрев, скажем, товары как множества пользователей, покупавших их, и посчитав Jaccard similarity каждого с каждым. Вообще, если у вас уже есть на примете что-то, что вы хотели бы покластеризовать, не стесняйтесь и пробуйте!

Список литературы

- [1] Ширяев А.Н. Вероятность - 1. Москва : МЦНМО, 2007, 552 стр.
- [2] Van Dongen, S. Graph Clustering by Flow Simulation. PhD Thesis, University of Utrecht, The Netherlands, 2000.
<https://micans.org/mcl/>