Отчет по заданию "Решение реальной задачи Topical classification of Biomedical Research Papers"

Малышева Екатерина

8 апреля 2012 г.

Постановка задачи

Команда организаторов собрала и с помощью алгоритма пометки (созданного командой) преобразовала 20 000 статьи, связанные с медициной, которые были опубликованы вместе с тегами - некоторыми метками (всего было 83 различных тега). Далее переработанные данные были разбиты на 2 части (с ответами и без) и предоставлены нам. Входные данные - матрица размером 10000 объектов на 25640 признаков и текстовый файл с ответами - к каким классам (из 83) принадлежит объект. В описании к задаче было сказано: "Intuitively, they can be interpreted as values of a rough membership function that measures a degree in which a term is present in a given text". После этого в моем представлении задача сводилась к следующему: были статьи, к ним был применен весьма странный алгоритм, получена признаковая матрица. По этой матрице нужно восстановить теги другой признаковой матрицы.

Ход решения

Предобработка данных

В качестве предобработки были попробованы различные нормировки:

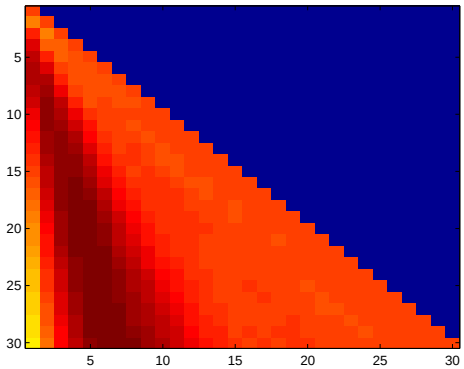
1. По столбцам
2. По строкам
3. По максимуму
4. В l_1 метрике (по сумме)
5. В l_2 метрике (по корню из суммы квадратов)

Наилучшее качество (при фиксированном алгоритме, описанном далее) было достигнуто при нормировке в l_2 метрике. В формулировке задания было указано: "The data sets are also sparse, since usually only a small fraction of the MeSH terms is assigned to a particular document by our tagging algorithm. Finally, a large number of data columns have little (or even none) non-zero

values (corresponding concepts are rarely assigned to documents). It is up to participants to decide which of them are still useful for the task."После прочтения этого абзаца была попытка удалить нулевые столбцы из обучающей выборки с последующим удалением из тестовой выборки. Но удаление ухудшило качество в среднем на 2 процента на CS.

Используемый алгоритм

За основу был взят метод ближайших соседей. Используемая метрика - косинусная (была попробована евклидова, результат давала примерно такой же, разброс на CS - 1 процент в среднем). Далее проводился подбор количества соседей и порог. Сначала порог был статическим - то есть для всех классов одинаковый. Ниже приведен график зависимости от количества соседей и порога, чем цвет темнее, тем качество лучше. Лучший результат -



коричневый - 46

В виду маломощности располагаемых ресурсов эксперимент на евклидовой метрике не проводился. Далее была попытка воспроизвести динамический порог - свой для каждого класса, в зависимости от частоты встречаемости. То есть если класс встречается часто, то порог для него выше, чем класс, который встречается реже. Алгоритм реализации примерно такой: для каждого класса считаем частоту встречаемости, получаем вектор размера 83, где каждое значение $\in [0, 1]$, далее переводим этот отрезок различными функциями в $1, 2..n$, где n - лучший результат порога для фиксированного числа соседей. Эксперимент проводился на следующих функциях:

1. $n(1 - (1 - M)^2) + const$
2. $nM + const$
3. $n(1 - e^{-M}) + const$
4. $n(1 - log_2(1 + e^{-M})) + const$
5. $n(1 - \frac{2}{1+e^{-M}}) + const$

Лучший результат (улучшение в 2 процента) получился для последней функции, которая и использовалась в финальном решении. Константа искалась CS.

Выводы

KNN - как базовый алгоритм дает неплохие результаты, если правильно подобрать параметры можно достичь результата в 50 процентов по F-мере.

Финальное решение

Нормализация всех данных по l_2 метрике. KNN с косинусной метрикой, где порог для каждого класса считается по формуле 5. 26 соседей, $t = 6, const = 2.4$, над функцией (5) round. Решение реализовано в среде программирования Matlab.

Аргументация финального решения

Данное решение было выбрано как лучшее из получившихся. Остальные алгоритмы и более точный подбор параметров не проводился, в виду маломощности используемой техники.

Что могу предоставить по решению

Код основной программы с различными вариантами подсчета динамического порога, код функций, производющих нормировку, код удаления нулевых признаков.

Советы новичкам

Скачайте liblinear, найдите товарища с мощным вычислительным ресурсом, разберитесь с использованием этой библиотеки, разработайте грамотную "сеточку"параметров, получите 52 процента и радуйтесь жизни.

Что узнала нового

Сама придумала динамический порог, очень порадовалась за это открытие. Научилась пользоваться liblinear, потыкала линейный классификатор. Задумалась о покупке нового ноутбука, на этом в половине случае при запуске меня радовала надпись "Out of memory". При наличии времени разобралась бы с SVD, его смысл мне стал понятен только после семинара. При наличии более мощной вычислительной техники (один запуск на данных 8000 на 2000 работал 14 секунд) я бы лучше подобрала параметры для функции динамического порога, количества соседей, используемой метрики.

Помощь команде

1. Выложила код, преобразующий текстовый файл в бинарную матрицу с 83 столбцами.

2. Провела достаточно полное исследование по метрическим алгоритмам классификации

Мне помогли

1. Советы Дмитрия Кондрашкина по улучшениям в скорости работы алгоритма, подсчет F-меры
2. Код Нижибицкого Евгений для удаления нулевых столбцов
3. Созданная Петром Ромовым страница обсуждения и пояснения по задаче
4. Код Ильдара Шаймарданова и поправка Марии Любимцевой для создания файла, готового к отправке
5. Моральная поддержка Лобачевой Екатерины и Огневой Дарьи при написании Short Report