Московский Государственный Университет
Факультет Вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Решение реальной задачи «Topical Classification of Biomedical Research Papers»

Отчет по заданию

Выполнила:
Потапенко Анна Александровна

Москва, 2012

1. Постановка задачи

Нам предложена реальная задача классификации с интернет-соревнования. Классифицируем тексты из области биомедицины по рубрикам. Представлено признаковое описание обучающих документов – сильно разреженная матрица размером количество_документов (10 000) на количество_признаков (25 640). Природа матрицы неизвестна, хотя есть предположение, что это матрица встречаемости слов в документах. Каждый документ относится к нескольким классам, метки принадлежности для обучающих документов известны и представлены в виде матрицы размером количество_документов (10 000) на количество_классов (83). Наша задача – предсказать, к каким классам относятся документы на контроле (их также 10 000). Качество классификации оценивается f-мерой.

2. Ход решения

Для внутренней проверки качества разбиваем обучающие документы на обучение (9/10) и контроль (1/10). Обучаемся на обучении и подсчитываем f-меру на контроле. Так как в ходе экспериментов усреднение качества по нескольким разбиениям не сильно отличалось от качества на одном фиксированном разбиении, часто для ускорения счета разбиваем только один раз.

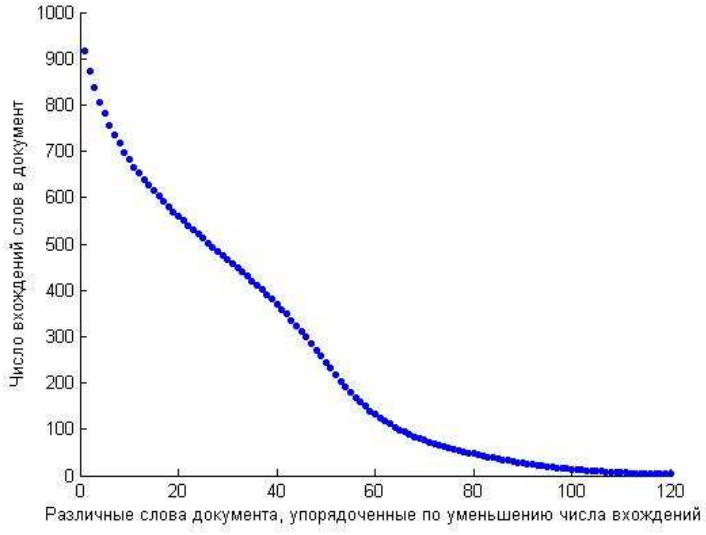
Исследование матрицы данных. Вопрос: можно ли трактовать матрицу данных как матрицу встречаемости терминов в документах.

1. Около 10 тысяч «слов» встречаются не более чем в 10 документах из 10 000. Интуитивно кажется, что таких редких слов должно быть меньше.

2. Максимальная длина документа около 60 000 слов, средняя – 30 000, минимальная 20 000. При этом максимальное число различных слов в документе – 200, среднее – 100, минимальное 60. Получается, в документе одни и те же слова повторяются очень много раз.

На графике показано, сколько раз встречаются в документе его различные слова.

Мы взяли 500 документов средней длины, упорядочили в каждом из них слова по уменьшению числа вхождений и усреднили число вхождений каждого i-го по популярности слова по выбранным документам (в разных документах это разные слова).



3. Можно было бы предположить, что у нас куча стоп-слов, и они и повторяются по много раз в каждом документе. Но нет – стоп-слова, употребительные слова языка должны встречаться в бОльшей части коллекции – а слов, которые встречаются более чем в 1000 документах из 10 000 – нет.

Таким образом, интуитивно матрица данных не похожа на матрицу встречаемости слов в документах. Однако это возможно, например, если статьи длинные, а рассматриваемые слова – это только специальные термины, а не общеупотребительные слова языка. Заметим, что многие алгоритмы классификации текстов (в частности, тематические модели) несмотря на свою терминологию, жестко не привязаны к сущностям документ-слово, и должны работать, даже если матрица данных имеет другую природу.

2.1. Метрические алгоритмы классификации текстов

Описание алгоритма:

1. Заменяем матрицу данных на матрицу весов TF-IDF.

Теперь каждый элемент матрицы это частота термина в документе, умноженная на логарифм обратной частоты термина по всей коллекции. Для этого нам необходимо произвести такие нормировки:

- делим каждый элемент на сумму по строке;
- умножаем на логарифм суммы всех элементов матрицы, деленной на сумму по столбцу.

2. Вычисляем центроиды для каждой рубрики.

Значения признаков для центроида рубрики - это среднее арифметическое значений признаков относящихся к этой рубрике документов.

3. Для каждого классифицируемого документа подсчитываем расстояния до всех центроидов, используя евклидову меру. Получаем матрицу близости документов к рубрикам (размер: количество классифицируемых документов на количество рубрик)

4. Преобразуем матрицу близости в матрицу принадлежности: относим каждый документ к 5 наиболее близким рубрикам.

Качество описанного алгоритма 0.39.

Сравнение метрик. Попробуем использовать различные метрики для подсчета расстояний от классифицируемых объектов до центроидов. Полученные результаты:

- 1. Евклидова метрика - 0.39
- 2. Манхэттенское расстояние - 0.17
- 3. Косинусная мера - 0.38
- 4. Расстояние Чебышева - 0.18

Совмещение алгоритмов. Евклидова метрика и косинусная мера дают неплохое качество классификации. Логично их каким-то образом совместить. На практике разные совмещения не дали существенного улучшения: результат 0.4, условие отнесения документа к рубрике "одна из 5 ближайших по евклидовой мере И близость по косинусной мере больше порога (0.024)". Возможно, причина в схожести совмещаемых алгоритмов.

Преобразование матрицы близости объектов к классам в матрицу принадлежности. Я отношу документ к 5 наиболее близким центроидам (рубрикам). Если аккуратно подобрать пороговое значение, то с условием "расстояние меньше порогового" результаты подобные, но не лучше.

Учет «скученности» рубрик. Чем ближе расположены представители рубрик к своим центроидам, тем большей близости мы должны требовать от классифицируемых объектов. Пробовала учитывать среднее расстояние от представителей до центроида, максимальное, медиану, расстояние от к-ого соседа центроида - качество классификации только портится.

2.2 Применение тематических моделей (topic modelling)

В ходе построения тематической модели определяются темы, и документы распределяются по этим темам. Попробуем использовать это в нашей задаче. Будем работать с одним из самых популярных алгоритмов – LDA (Latent Dirichlet Allocation).

Сэмплирование Гиббса (GS)

Одна из быстрых реализаций метода LDA - это сэмплирование Гиббса. Алгоритм можно посмотреть на слайде:

Алгоритм сэмплирования Гиббса

Вход: коллекция $X = \{(d, w)\colon d \in D,\ w \in d\}$; параметры α, β;

Выход: оценки $\hat{p}(w|t)$, $\hat{p}(t|d)$;

$n_{wt} := \beta$; $n_{td} := \alpha$ для всех $d \in D$, $w \in W$, $t \in T$;
$n_t := \beta|W|$; $n_d := \alpha|T|$ для всех $d \in D$, $t \in T$;
для $i := 1, \ldots, M$ — генерация M сэмплов
  для всех документов $d \in D$ и всех слов $w \in d$
    $\tilde{p}(t|d, w) := (n_{wt}/n_t) \cdot (n_{td}/n_d)$ для всех $t \in T$;
    если $i \geqslant 2$ **то** $t := t_{dw}$; $--n_{wt}$; $--n_{td}$; $--n_t$; $--n_d$;
    выбрать t из ненормированного распределения $\tilde{p}(t|d, w)$;
    $t_{dw} := t$; $++n_{wt}$; $++n_{td}$; $++n_t$; $++n_d$;
$\hat{p}(w|t) := n_{wt}/n_t$ для всех $w \in W$, $t \in T$;
$\hat{p}(t|d) := n_{td}/n_d$ для всех $d \in D$, $t \in T$;

Модификация GS для задачи классификации

Будем рассматривать обучение и контроль как единую коллекцию документов. Коллекция частично размечена - мы знаем принадлежности обучающих документов к темам. Такую постановку можно трактовать как задачу с частичным обучением - semi-supervised learning. Все что нужно изменить в GS - это учесть известную информацию в начальных приближениях. Я делала это так:

Длины документов n_d - просто суммируем матрицу данных по строкам.

Число слов в документе посвященное теме n_{td} (для обучения) - 0, если документ не отнесен к этой теме; длина документа делить на число тем, к которым он отнесен по классификации, если отнесен.

Число слов w, относящихся к теме n_{wt} (подсчитанное по обучению) - каждое слово каждого документа распределяем в равных долях по темам, к которым отнесен документ.

Число слов в документе посвященное теме n_{td} (для контроля) - считаем, что каждое слово документа принадлежит теме t в доле n_{wt} / n_w, где n_w - сколько всего таких слов в обучении.

Число слов w, относящихся к теме n_{wt} (досчитанное по контролю) - Каждое слово на контроле добавляет к n_{wt} величину n_{wt} / n_w, заметим, что распределение слова по темам при этом не меняется.

Число слов в теме n_t - просто суммируем матрицу n_{wt} по столбцам.

Далее реализуем описанный на слайде алгоритм. Замечу, что в данном случае длина документа очень велика, поэтому внутренний цикл я вела не по всем словам документа, а по его различным словам, а значит, выбирала единую тему на все вхождений конкретного слова в документ. Это не очень хорошо, потому что обновление счетчиков, связанное с распределением большого числа вхождений одного и того же слова может заметно сказаться на распределении p(t|d) - сместить его. Чтобы этого избежать, стоит разбить внутренний цикл по различным словам на два таких цикла. В первом пересчитывать распределение p(t|w,d), в другом выбирать тему и обновлять счетчики.

Итак, получился алгоритм классификации. Результаты осмысленные, но не очень хорошие – f-мера 0.41.

Использование GS для создания нового признакового пространства

Другой подход к использованию сэмлирования Гиббса в нашей задаче: сэмлирование Гиббса с обычными начальными приближениями производит классификацию документов по некоторому количеству тем (например, 200), никак не относящимся к нашим целевым рубрикам. Мы получаем

новое описание документов в пространстве из 200 признаков и применяем какой-либо алгоритм классификации. При использовании линейного классификатора без оптимизации параметров (из LIBLINEAR) был получен результат около 0.45.

2.3. Линейные классификаторы, использование библиотеки LIBLINEAR

По советам с вкладки «обсуждение» я решила обратиться к библиотеке LIBLINEAR и использовать 83 независимых классификатора SVM. Каждый из них определяет принадлежность к определенному классу, если ни один из классификаторов не выбирает документ, то мы относим его к 5 самым популярным рубрикам. Аккуратный подбор параметров для каждого классификатора я не осуществляла. Почитав help и примерно поняв суть параметров, я подобрала их руками – это были '-c 0.2 –w1 4 –w2 1'. Предварительная обработка данных включает удаление везде нулевых признаков и деление всей матрицы на подобранное значение (6000). Различные нормировки по столбцам или строкам пользы не принесли. Это решение стало финальным, так как давало лучший результат: f-мера 0.521 по предварительной оценке сайта.

4. Заключение

При решении задачи были попробованы разные подходы и реализованы разные идеи, эти эксперименты было интересно проводить, и на них ушло много времени и сил. Однако высокого результата они не принесли, и финальной стала программа, написанная за несколько часов. Подобную картину я заметила у многих одногруппников. Из такого опыта получается какой-то грустный совет новичкам: не изобретать велосипед, не писать много когда, а почувствовать какой известный алгоритм может сработать, найти его реализацию в интернете и применить. Но можно сделать и другой вывод: это была первая реальная задача для многих из нас, поэтому, возможно, нам не хватало опыта и чутья, и мы пробовали применить заведомо бесполезные в данном случае методы. Тогда надо продолжать экспериментировать и не забывать при этом думать.

По решению задачи я могу предоставить все коды, подробные описания своих мыслей и действий. В частности, я долго разбиралась с LDA-GS, по многим наблюдениям понятно, что моя реализация алгоритма работает не до конца верно. При наличии времени я бы хотела окончательно с ней разобраться. Кроме того, я нашла хорошую готовую реализацию (GibbsLDA++) и научилась обрабатывать с её помощью наши данные, однако довести эти эксперименты до результатов классификации не успела. Интересно также подумать о теоретической проблеме применения тематических моделей к задаче классификации.

Польза группе и группы

1. Описала свои наблюдения о матрице данных, продолжила исследование ребят с метрическими классификаторами на основе TF-IDF и центроидов, окончательно убедившись, что этот метод не даст высоких результатов, описала возможность применения LDA к задаче классификации.
2. Спасибо всем, кто участвовал в обсуждении задачи! Я почерпнула оттуда полезные вещи, хотя порой количество умных слов наоборот демотивировало.

Отдельное спасибо Андрею Остапцу за подробное описание алгоритма с линейными классификаторами, Дмитрию Кондрашкину за полезные наработки кода и Евгению Заку за моральную поддержку :)

Список использованной литературы:

1. К.В. Воронцов – Математические методы обучения по прецедентам (теория обучения машин).
2. David M. Blei, Andrew Y. Ng, Michael I. Jordan – Latent Dirichlet Allocation
3. machinelearning.ru (презентация К.В. Воронцова о тематических моделях)
4. страница обсуждения задачи