ОТ НАИВНОГО БАЙЕСА ДО ТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ I

Сергей Николенко

НИУ ВШЭ — Санкт-Петербург
6 октября 2017 г.

Random facts:

- 6 октября 1917 г. *Literary Digest* впервые назвал музыку, «вызывающую желание трястись и корчиться», словом «джаз», а 6 октября 1927 г. на экраны вышел первый в истории звуковой фильм, *The Jazz Singer*
- 6 октября 1995 г. была обнаружена планета у звезды 51 Пегаса, первая найденная людьми экзопланета у солнцеподобной звезды
- 6 октября 2005 г. Джейсон Льюис завершил первое в истории кругосветное путешествие на мускульной тяге; в ходе путешествие Льюис перенёс два случая малярии, лёгкий приступ шизофрении и нападение крокодилов
- 6 октября 2010 г. был запущен *Instagram*

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР

КАТЕГОРИЗАЦИЯ ТЕКСТОВ

- Классическая задача машинного обучения и information retrieval – категоризация текстов.
- Дан набор текстов, разделённый на категории. Нужно обучить модель и потом уметь категоризовать новые тексты.
- Атрибуты $a_1, a_2, \ldots, a_n$ – это слова, v – тема текста (или атрибут вроде «спам / не спам»).
- Bag-of-words model: забываем про порядок слов, составляем словарь. Теперь документ – это вектор, показывающий, сколько раз каждое слово из словаря в нём встречается.

NAIVE BAYES

- Заметим, что даже это – сильно упрощённый взгляд: для слов ещё довольно-таки важен порядок, в котором они идут...
- Но и это ещё не всё: получается, что $p(a_1, a_2, \dots, a_n | x = v)$ – это вероятность *в точности такого набора слов* в сообщениях на разные темы. Очевидно, такой статистики взять неоткуда.
- Значит, надо дальше делать упрощающие предположения.
- Наивный байесовский классификатор – самая простая такая модель: давайте предположим, что все слова в словаре условно независимы при условии данной категории.

NAIVE BAYES

- Иначе говоря:

$$p(a_1, a_2, \dots, a_n | x = v) = p(a_1 | x = v) p(a_2 | x = v) \dots p(a_n | x = v).$$

- Итак, наивный байесовский классификатор выбирает v как

$$v_{NB}(a_1, a_2, \dots, a_n) = \arg\max_{v \in V} p(x = v) \prod_{i=1}^{n} p(a_i | x = v).$$

- В парадигме классификации текстов мы предполагаем, что разные слова в тексте на одну и ту же тему появляются независимо друг от друга. Однако, несмотря на такие бредовые предположения, naive Bayes на практике работает очень даже неплохо (и этому есть разумные объяснения).

Многомерная модель

- Но в деталях реализации наивного байесовского классификатора есть тонкости.
- Сейчас мы рассмотрим два разных подхода к naive Bayes, которые дают разные результаты: мультиномиальный (multinomial) и многомерный (multivariate).

Многомерная модель

- В многомерной модели документ – это вектор бинарных атрибутов, показывающих, встретилось ли в документе то или иное слово.
- Когда мы подсчитываем правдоподобие документа, мы перемножаем вероятности того, что встретилось каждое слово из документа и вероятности того, что не встретилось каждое (словарное) слово, которое не встретилось.
- Получается модель многомерных испытаний Бернулли. Наивное предположение в том, что события «встретилось ли слово» предполагаются независимыми.

МНОГОМЕРНАЯ МОДЕЛЬ

- Математически: пусть $V = \{w_t\}_{t=1}^{|V|}$ – словарь. Тогда документ d_i – это вектор длины $|V|$, состоящий из битов B_{it}; $B_{it} = 1$ iff слово w_t встречается в документе d_i.
- Правдоподобие принадлежности d_i классу c_j:

$$p(d_i \mid c_j) = \prod_{t=1}^{|V|} \left(B_{it} p(w_t \mid c_j) + (1 - B_{it})(1 - p(w_t \mid c_j))\right).$$

- Для обучения такого классификатора нужно обучить вероятности $p(w_t \mid c_j)$.

МНОГОМЕРНАЯ МОДЕЛЬ

- Обучение – дело нехитрое: пусть дан набор документов $D = \{d_i\}_{i=1}^{|D|}$, которые уже распределены по классам c_j (возможно, даже вероятностно распределены), дан словарь $V = \{w_t\}_{t=1}^{|V|}$, и мы знаем биты B_{it} (знаем документы).
- Тогда можно подсчитать оптимальные оценки вероятностей того, что то или иное слово встречается в том или ином классе (при помощи лапласовой оценки):

$$p(w_t \mid c_j) = \frac{1 + \sum_{i=1}^{|D|} B_{it} p(c_j \mid d_i)}{2 + \sum_{i=1}^{|D|} p(c_j \mid d_i)}.$$

МНОГОМЕРНАЯ МОДЕЛЬ

- Априорные вероятности классов можно подсчитать как $p(c_j) = \frac{1}{|D|} \sum_{i=1}^{|D|} p(c_j \mid d_i)$.
- Тогда классификация будет происходить как

$$c = \arg\max_j p(c_j) p(d_i \mid c_j) =$$

$$= \arg\max_j \left(\frac{1}{|D|} \sum_{i=1}^{|D|} p(c_j \mid d_i) \right) \prod_{t=1}^{|V|} \left(B_{it} p(w_t \mid c_j) + (1 - B_{it})(1 - p(w_t \mid c_j)) \right) =$$

$$= \arg\max_j \left(\log(\sum_{i=1}^{|D|} p(c_j \mid d_i)) + \sum_{t=1}^{|V|} \log \left(B_{it} p(w_t \mid c_j) + (1 - B_{it})(1 - p(w_t \mid c_j)) \right) \right)$$

МУЛЬТИНОМИАЛЬНАЯ МОДЕЛЬ

- В мультиномиальной модели документ – это последовательность событий. Каждое событие – это случайный выбор одного слова из того самого «bag of words».
- Когда мы подсчитываем правдоподобие документа, мы перемножаем вероятности того, что мы достали из мешка те самые слова, которые встретились в документе. Наивное предположение в том, что мы достаём из мешка разные слова независимо друг от друга.
- Получается мультиномиальная генеративная модель, которая учитывает количество повторений каждого слова, но не учитывает, каких слов *нет* в документе.

МУЛЬТИНОМИАЛЬНАЯ МОДЕЛЬ

- Математически: пусть $V = \{w_t\}_{t=1}^{|V|}$ – словарь. Тогда документ d_i – это вектор длины $|d_i|$, состоящий из слов, каждое из которых «вынуто» из словаря с вероятностью $p(w_t \mid c_j)$.
- Правдоподобие принадлежности d_i классу c_j:

$$p(d_i \mid c_j) = p(|d_i|)|d_i|! \prod_{t=1}^{|V|} \frac{1}{N_{it}!} p(w_t \mid c_j)^{N_{it}},$$

  где N_{it} – количество вхождений w_t в d_i.
- Для обучения такого классификатора тоже нужно обучить вероятности $p(w_t \mid c_j)$.

МУЛЬТИНОМИАЛЬНАЯ МОДЕЛЬ

- Обучение: пусть дан набор документов $D = \{d_i\}_{i=1}^{|D|}$, которые уже распределены по классам c_j (возможно, даже вероятностно распределены), дан словарь $V = \{w_t\}_{t=1}^{|V|}$, и мы знаем вхождения N_{it}.
- Тогда можно подсчитать апостериорные оценки вероятностей того, что то или иное слово встречается в том или ином классе (не забываем сглаживание – правило Лапласа):

$$p(w_t \mid c_j) = \frac{1 + \sum_{i=1}^{|D|} N_{it} p(c_j \mid d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N_{is} p(c_j \mid d_i)}.$$

МУЛЬТИНОМИАЛЬНАЯ МОДЕЛЬ

- Априорные вероятности классов можно подсчитать как $p(c_j) = \frac{1}{|D|}\sum_{i=1}^{|D|} p(c_j \mid d_i)$.
- Тогда классификация будет происходить как

$$
\begin{aligned}
c &= \arg\max_j p(c_j)p(d_i \mid c_j) = \\
&= \arg\max_j \left(\frac{1}{|D|}\sum_{i=1}^{|D|} p(c_j \mid d_i)\right) p(|d_i|)|d_i|! \prod_{t=1}^{|V|} \frac{1}{N_{it}!} p(w_t \mid c_j)^{N_{it}} = \\
&= \arg\max_j \left(\log\left(\sum_{i=1}^{|D|} p(c_j \mid d_i)\right) + \sum_{t=1}^{|V|} N_{it} \log p(w_t \mid c_j)\right).
\end{aligned}
$$

Как можно обобщить наивный байес

- В наивном байесе есть два сильно упрощающих дело предположения:
  - мы знаем метки тем всех документов;
  - у каждого документа только одна тема.
- Мы сейчас уберём оба эти ограничения.
- Во-первых, что можно сделать, если мы не знаем метки тем, т.е. если датасет неразмеченный?

КЛАСТЕРИЗАЦИЯ

- Тогда это превращается в задачу кластеризации.
- Её можно решать EM-алгоритмом (Expectation-Maximization, используется в ситуациях, когда есть много скрытых переменных, причём если бы мы их знали, модель стала бы простой):
  - на E-шаге считаем ожидания того, какой документ какой теме принадлежит;
  - на M-шаге пересчитываем наивным байесом вероятности $p(w \mid t)$ при фиксированных метках.
- Это простое обобщение.

ТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Как ещё можно обобщить наивный байес

- А ещё в наивном байесе у документа только одна тема.
- Но это же не так! На самом деле документ говорит о многих темах (но не слишком многих).
- Давайте попробуем это учесть.

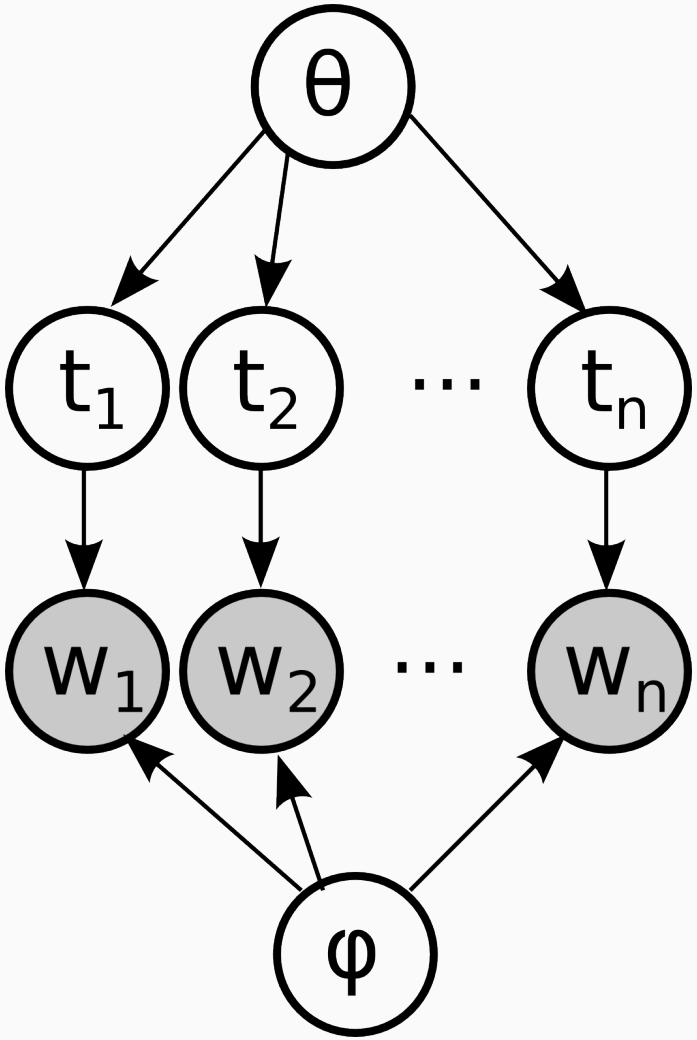
PLSA

- Рассмотрим такую модель:
  - каждое слово в документе d порождается некоторой темой $t \in T$;
  - документ порождается некоторым распределением на темах $p(t \mid d)$;
  - слово порождается именно темой, а не документом: $p(w \mid d, t) = p(w \mid d)$;
  - итого получается такая функция правдоподобия:

$$p(w \mid d) = \sum_{t \in T} p(w \mid t) p(t \mid d).$$

- Эта модель называется probabilistic latent semantic analysis, pLSA (Hoffmann, 1999).

PLSA: ГРАФИЧЕСКАЯ МОДЕЛЬ ДОКУМЕНТА



PLSA

- Как её обучать? Мы можем оценить $p(w \mid d) = \frac{n_{wd}}{n_d}$, а нужно найти:
  - $\phi_{wt} = p(w \mid t)$;
  - $\theta_{td} = p(t \mid d)$.
- Максимизируем правдоподобие

$$p(D) = \prod_{d \in D} \prod_{w \in d} p(d, w)^{n_{dw}} = \prod_{d \in D} \prod_{w \in d} \left[\sum_{t \in T} p(w \mid t) p(t \mid d) \right]^{n_{dw}}.$$

- Как максимизировать такие правдоподобия?

- EM-алгоритмом. На E-шаге ищем, сколько слов w в документе d из темы t:

$$n_{dwt} = n_{dw} p(t \mid d, w) = n_{dw} \frac{\phi_{wt}\theta_{td}}{\sum_{s \in T} \phi_{ws}\theta_{sd}}.$$

- А на M-шаге пересчитываем параметры модели:

$$n_{wt} = \sum\nolimits_d n_{dwt}, \qquad n_t = \sum\nolimits_w n_{wt}, \quad \phi_{wt} = \tfrac{n_{wt}}{n_t},$$
$$n_{td} = \sum\nolimits_{w \in d} n_{dwt}, \quad \theta_{td} = \tfrac{n_{td}}{n_d}.$$

- Вот и весь вывод в pLSA.

PLSA

- Можно даже не хранить всю матрицу n_{dwt}, а двигаться по документам, каждый раз добавляя n_{dwt} сразу к счётчикам n_{wt}, n_{td}.
- Чего тут не хватает?
  - Во-первых, разложение такое, конечно, будет сильно не единственным.
  - Во-вторых, параметров очень много, явно будет оверфиттинг, если корпус не на порядки больше числа тем.
  - А совсем хорошо было бы получать не просто устойчивое решение, а обладающее какими-нибудь заданными хорошими свойствами.
- Всё это мы можем решить как?

- Правильно, регуляризацией. Есть целая наука о разных регуляризаторах для pLSA (К.В. Воронцов).
- В общем виде так: добавим регуляризаторы R_i в логарифм правдоподобия:

$$\sum_{d \in D} \sum_{w \in d} n_{dw} \ln \sum_{t \in T} \phi_{wt} \theta_{td} + \sum_i \tau_i R_i(\Phi, \Theta).$$

- Тогда в EM-алгоритме на M-шаге появятся частные производные R:

$$n_{wt} = \left[\sum_{d \in D} n_{dwt} + \phi_{wt}\frac{\partial R}{\partial \phi_{wt}}\right]_+,$$

$$n_{td} = \left[\sum_{w \in d} n_{dwt} + \theta_{td}\frac{\partial R}{\partial \theta_{td}}\right]_+$$

- Чтобы доказать, EM надо рассмотреть как решение задачи оптимизации через условия Каруша-Куна-Такера.

PLSA

- И теперь мы можем кучу разных регуляризаторов вставить в эту модель:
  - регуляризатор сглаживания (позже, это примерно как LDA);
  - регуляризатор разреживания: максимизируем KL-расстояние между распределениями ϕ_{wt} и θ_{td} и равномерным распределением;
  - регуляризатор контрастирования: минимизируем ковариации между векторами ϕ_t, чтобы в каждой теме выделилось своё лексическое ядро (характерные слова);
  - регуляризатор когерентности: будем награждать за слова, которые в документах стоят ближе друг к другу;
  - и так далее, много всего можно придумать.

СПАСИБО!

Спасибо за внимание!