Байесовские классификаторы

Сергей Николенко

Академический Университет, весенний семестр 2011

Outline

Применяем теорему Байеса

- Итак, нам нужно найти наиболее вероятную гипотезу $h \in H$ при условии данных D.
- Иными словами, нужно максимизировать $p(h|D)$.
- Что нам скажет теорема Байеса?

Применяем теорему Байеса

- Итак, нам нужно найти наиболее вероятную гипотезу $h \in H$ при условии данных D.
- Иными словами, нужно максимизировать $p(h|D)$.
- Что нам скажет теорема Байеса?
- $$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

Применяем теорему Байеса

$$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

Применяем теорему Байеса

$$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

- Итого нам нужно найти гипотезу

$$h = \operatorname{argmax}_{h \in H} p(h|D).$$

- Такая гипотеза называется *максимальной апостериорной гипотезой* (maximum a posteriori hypothesis, MAP).

Применяем теорему Байеса

$$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

$$h = \operatorname{argmax}_{h \in H} p(h|D) =$$

$$= \operatorname{argmax}_{h \in H} \frac{p(D|h)p(h)}{p(D)} = \operatorname{argmax}_{h \in H} p(D|h)p(h),$$

потому что $p(D)$ от h не зависит.

Применяем теорему Байеса

$$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

Часто предполагают, что гипотезы изначально равновероятны: $p(h_i) = p(h_j)$. Тогда ещё проще:

$$h = \text{argmax}_{h \in H} p(D|h).$$

Алгоритм

- Для каждой гипотезы $h \in H$ вычислить апостериорную вероятность

$$p(h|D) = \frac{p(D|h)p(h)}{p(D)}.$$

- Выбрать гипотезу с максимальной апостериорной вероятностью:

$$h = \operatorname{argmax}_{h \in H} p(h|D).$$

Как его применять: пример

- Нужно задать $p(h)$ и $p(D|h)$.
- Пусть выполняются следующие условия.
  - В D нет шума (т.е. все тестовые примеры с правильными ответами).
  - Целевая функция c лежит в H.
  - Нет априорных причин верить, что одна из гипотез более вероятна, чем другая.

Задачи классификации

- Эти условия выполняются в задачах классификации.
- Тогда каждая гипотеза, совместимая со всеми данными, будет максимальной апостериорной гипотезой:

$$p(h|D) = \begin{cases} \frac{1}{|\mathrm{Cons}(d)|}, & \text{если } d_i = h(x_i) \text{ для всех } d_i \in D, \\ 0, & \text{в противном случае.} \end{cases}$$

Упражнение. Докажите это формально.

Постановка задачи

- До сих пор мы отвечали на вопрос: «Какова наиболее вероятная гипотеза при имеющихся данных?»
- Теперь пора ответить на вопрос «Какова наиболее вероятная классификация нового примера при имеющихся данных?»

Постановка задачи

- Казалось бы, можно просто применить максимальную апостериорную гипотезу. Почему нет?

Постановка задачи

- Казалось бы, можно просто применить максимальную апостериорную гипотезу. Почему нет?
- Пусть есть четыре гипотезы, и их апостериорные вероятности 0.2, 0.2, 0.2, 0.4. Четвёртая гипотеза — максимальная апостериорная. Но если новый пример классифицируется первыми тремя положительно, а четвёртой — отрицательно, то общая вероятность его положительной классификации 0.6, и применять MAP было бы неправильно.

Задача оптимальной классификации

Пусть имеются данные D и множество гипотез h. Для вновь поступившего примера x нужно выбрать такое значение v, чтобы максимизировать $p(v|D)$. Иными словами, наша задача — найти

$$\arg\max_{v \in V} \sum_{h \in H} p(v|h) p(h|D).$$

Оптимальный классификатор

Определение

Любой алгоритм, который решает задачу

$$\arg\max_{v \in V} \sum_{h \in H} p(v|h)p(h|D),$$

называется оптимальным байесовским классификатором (optimal Bayes classifier).

Пример

У нас уже был пример — четыре гипотезы h_i, $i = 1..4$, множество значений $V = \{0, 1\}$, и вероятности

$$\begin{aligned} &p(h_1|D) = p(h_2|D) = p(h_3|D) = 0.2, && p(h_4|D) = 0.4, \\ &p(x = 1|h_1) = p(x = 1|h_2) = p(x = 1|h_3) = 1, && p(x = 1|h_4) = 0, \\ &p(x = 0|h_1) = p(x = 0|h_2) = p(x = 0|h_3) = 0, && p(x = 0|h_4) = 1. \end{aligned}$$

Тогда

$$\sum_i p(x = 1|h_i)p(h_i|D) = 0.6, \quad \sum_i p(x = 0|h_i)p(h_i|D) = 0.4.$$

Свойства оптимального классификатора

- Он действительно оптимален: никакой другой метод не может в среднем превзойти его.
- Он может даже классифицировать данные по гипотезам, не содержащимся в H. Например, он может классифицировать по любому элементу линейной оболочки H.
- Его обычно не получается эффективно реализовать — нужно перебирать все гипотезы, а всех гипотез очень много.

Алгоритм Гиббса

- Как можно ускорить процесс? Алгоритм Гиббса:
  - Выбрать случайную гипотезу $h \in H$ согласно распределению их апостериорных вероятностей.
  - Классифицировать новый случай x согласно h.
- То есть мы заменяем взвешенную сумму по всем гипотезам на случайную гипотезу, выбранную по соответствующему распределению.

Алгоритм Гиббса

- Как можно ускорить процесс? Алгоритм Гиббса:
  - Выбрать случайную гипотезу $h \in H$ согласно распределению их апостериорных вероятностей.
  - Классифицировать новый случай x согласно h.
- Ошибка алгоритма Гиббса при определённых не слишком жёстких условиях лишь вдвое больше ошибки оптимального классификатора!
- Правда, доказать это не так просто, и мы сейчас не будем; см. (Haussler, Kearns, Shapire, 1994).

Общая идея

- Наивный байесовский классификатор (naive Bayes classifier, idiot's Bayes) применяется в тех же случаях — для классификации данных.
- Он особенно полезен в ситуациях, когда разных атрибутов очень много; например, в классификации текстов.

Вывод формул

Дано:

- Каждый пример x принимает значения из множества V и описывается атрибутами $\langle a_1, a_2, \ldots, a_n \rangle$.
- Нужно найти наиболее вероятное значение данного атрибута, т.е.

$$v_{\text{MAP}} = \arg\max_{v \in V} p(x = v | a_1, a_2, \ldots, a_n).$$

- По теореме Байеса,

$$v_{\text{MAP}} = \arg\max_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \arg\max_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

Вывод формул

- По теореме Байеса,

$$v_{\text{MAP}} = \arg\max_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \arg\max_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

- Оценить $p(x = v)$ легко: будем оценивать частоту его встречаемости.
- Но оценить разные $p(a_1, a_2, \ldots, a_n | x = v)$ не получится — их слишком много; нам нужно каждый случай уже пронаблюдать несколько раз, чтобы получилось как надо.

Вывод формул

- По теореме Байеса,

$$v_{\text{MAP}} = \arg\max_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \arg\max_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

- Пример: классификация текстов.
- Атрибуты $a_1, a_2, \ldots, a_n$ – это слова, v – тема текста (или атрибут вроде «спам / не спам»).
- Тогда $p(a_1, a_2, \ldots, a_n | x = v)$ – это вероятность *в точности такого набора слов* в сообщениях на разные темы. Очевидно, такой статистики взять неоткуда.
- Заметим, что даже это – сильно упрощённый взгляд: для слов ещё важен порядок, в котором они идут...

Вывод формул

- По теореме Байеса,

$$v_{\text{MAP}} = \arg\max_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \arg\max_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

- Поэтому давайте предположим условную независимость атрибутов при условии данного значения целевой функции. Иначе говоря:

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

Вывод формул

- По теореме Байеса,

$$v_{\text{MAP}} = \arg\max_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \arg\max_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

Итак, наивный байесовский классификатор выбирает v как

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

- В парадигме классификации текстов мы предполагаем, что разные слова в тексте на одну и ту же тему появляются независимо друг от друга.
- Бред, конечно...

Насколько хорош naive Bayes

- На самом деле наивный байесовский классификатор гораздо лучше, чем кажется.
- Его оценки вероятностей оптимальны, конечно, только в случае независимости.
- Но сам классификатор оптимален в куда более широком классе задач.

Насколько хорош naive Bayes

- Есть два (в том числе формальных) общих объяснения этому факту.
  1. Атрибуты, конечно, зависимы, но их зависимость одинакова для разных классов и «взаимно сокращается» при оценке вероятностей. Грамматические и семантические зависимости между словами одни и те же и в тексте про футбол, и в тексте о байесовском обучении.
  2. Для *оценки вероятностей* наивный байесовский классификатор очень плох, но как *классификатор* гораздо лучше. Например, возможно, что на самом деле $p(x = v_0 \mid D) = 0.51$ и $p(x = v_1 \mid D) = 0.49$, а наивный классификатор выдаст $p(x = v_0 \mid D) = 0.99$ и $p(x = v_1 \mid D) = 0.01$; но классификация от этого не изменится.
- Мы сейчас не будем этим подробно заниматься; см. [Domingos and Pazzani, 1997; Zhang, 2004].

Outline

Два подхода

- В деталях реализации наивного байесовского классификатора прячется небольшой дьяволёнок.
- Сейчас мы рассмотрим два разных подхода к naive Bayes, которые дают разные результаты: мультиномиальный (multinomial) и многомерный (multivariate).
- Разница особенно отчётливо проявляется в классификации текстов. Она заключается в том, как именно порождается документ (это называется *генеративной моделью*).
- В дальнейшем мы будем использовать терминологию из мира текстов и документов.

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.
- Для применения требуется зафиксировать словарь, а количество повторений каждого слова теряется.

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

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

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

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

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 оптимальные оценки вероятностей того, что то или иное слово встречается в том или ином классе (тоже сгладив по Лапласу):

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

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

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

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

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

Thank you!

Спасибо за внимание!