Байесовские классификаторы

Сергей Николенко

Machine Learning — CS Club, весна 2008

Outline

Concept learning

- Обучение концептам, иногда ещё называют «формированием понятий».
- Метод обучения, принципиально похожий на деревья принятия решений, но с другими результатами
- Здесь немного не на месте, но хорошая иллюстрация.
- Это ещё один метод, результаты которого будут соответствовать MAP для тех же условий.

Два алгоритма

- Find-S — попроще и похуже.
- Candidate Elimination — посложнее и получше.

Основные принципы

- Обычная задача классификации — игры «Зенита».
- Будем рассматривать гипотезы в виде набора причин, из которых следует, что целевая функция равна 1.
- Гипотеза — набор значений атрибутов. Может содержать ? (любое значение) или $\emptyset$ (ни одного значения, пустая гипотеза).

Примеры гипотез

Для игр «Зенита» атрибуты:

$$\langle \text{Соперник}, \text{Играем}, \text{Лидеры}, \text{Дождь} \rangle$$

Примеры гипотез:

$$\langle \text{Выше}, \text{Дома}, ?, ? \rangle$$
$$\langle ?, \text{В гостях}, \text{Играют}, ? \rangle$$
$$\langle ?, ?, ?, ? \rangle$$
$$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$$

Более общие и более частные гипотезы

- На гипотезах есть естественный порядок.
- Гипотеза h_1 называется *более общей*, чем гипотеза h_2 (обозначаем $h_1 \succ h_2$), если для всякого тестового примера d, если $h_2(d) = 1$, то и $h_1(d) = 1$.
- Гипотеза $\langle \emptyset, \ldots, \emptyset \rangle$ является самой частной из всех гипотез, гипотеза $\langle ?, \ldots, ? \rangle$ — самой общей.

Find-S: идея

- Алгоритм Find-S очень прост: нужно начать с самой частной гипотезы, а затем обобщать её так, чтобы она включала в себя все тестовые примеры.
- На каждом шаге, если текущая гипотеза h неправильно классифицирует пример $(h(d) = 0)$, нужно искать минимальную h' из тех, для которых $h' \succ h$ и $h' = 0$; т.е., просто заменять неподходящие значения на ?.
- В результате получится максимально частная гипотеза, отвечающая всем тестовым данным.
- Негативные тестовые примеры вообще игнорируются.

Алгоритм Find-S

FindS(D)

- $h := \langle \emptyset, \dots, \emptyset \rangle$.
- Для всех $d \in D^+$:
  - Если $h(d) = 0$:
    - $h' := h$.
    - Для каждого атрибута a, если $h[a] \neq d[a]$, то $h'[a] := ?$.
    - $h := h'$.
- Выдать h.

Find-S: пример

- Возьмём игры «Зенита» и тестовые примеры (только положительные)

Соперник	Играем	Лидеры	Дождь	Победа
Выше	Дома	На месте	Нет	Да
Выше	Дома	Пропускают	Нет	Да
Ниже	Дома	Пропускают	Нет	Да

Find-S: пример

- $h = \langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$.
- Первый пример: Выше Дома На месте Нет
- h выдаёт 0, значит, надо обобщать. Максимально частная гипотеза просто совпадёт с этим тестовым примером.

Find-S: пример

- $h = \langle \text{Выше}, \text{Дома}, \text{На месте}, \text{Нет} \rangle$.
- Второй пример: Выше Дома Пропускают Нет
- h выдаёт 0. Максимально частная гипотеза, объединяющая h и этот пример, обобщит значение атрибута Лидеры.

Find-S: пример

- $h = \langle \text{Выше}, \text{Дома}, ?, \text{Нет} \rangle$.
- Третий пример: Ниже Дома Пропускают Нет
- h выдаёт 0. Максимально частная гипотеза, объединяющая h и этот пример, обобщит значение атрибута Соперник.

Find-S: пример

- Итого получаем гипотезу, объясняющую все тестовые примеры:
$$h = \langle ?, \text{Дома}, ?, \text{Нет} \rangle.$$

За и против

- Очень простой и быстрый алгоритм.
- Совсем не выразительный — нужно, чтобы целевая функция выражалась такой гипотезой, т.е. фактически одной веткой дерева; дизъюнкции не разрешаются.
- Но зато, если всё же выражается, то полученный результат будет правильно классифицировать и негативные примеры тоже (т.к. результат максимально частный).

Общие принципы

- Find-S позволяет найти наиболее частную гипотезу, совместную с положительными примерами.
- Что напрашивается?

Общие принципы

- Find-S позволяет найти наиболее частную гипотезу, совместную с положительными примерами.
- Что напрашивается?
- Хочется найти наиболее общую гипотезу, совместную с негативными примерами.

Общие принципы

- Find-S позволяет найти одну из возможных гипотез, а их на самом деле миллион.
- Что напрашивается?

Общие принципы

- Find-S позволяет найти одну из возможных гипотез, а их на самом деле миллион.
- Что напрашивается?
- Хочется искать два множества: множество минимальных относительно $\succ$ (наиболее частных) гипотез, совместных с данными, и множество максимальных (наиболее общих).

Общие принципы

- Find-S позволяет найти одну из возможных гипотез, а их на самом деле миллион.
- Что напрашивается?
- Хочется искать два множества: множество минимальных относительно $\succ$ (наиболее частных) гипотез, совместных с данными, и множество максимальных (наиболее общих). В этом случае то, что получится, будет границами всего множества допустимых гипотез, и любая другая гипотеза, совместная с тестовыми данными, будет находиться между этими двумя.

Сеттинг

- Итак, есть два множества: G — множество максимальных (общих) гипотез, и S — множество минимальных (частных) гипотез.
- Когда приходит новый пример d, есть несколько возможностей:

Сеттинг

- Итак, есть два множества: G — множество максимальных (общих) гипотез, и S — множество минимальных (частных) гипотез.
- Когда приходит новый пример d, есть несколько возможностей:
- $t_d = 1$
- В этом случае, если оказывается, что гипотеза из G неверно классифицирует d, её нужно удалить.
- А если неверно классифицируют гипотезы из S, их нужно соответственно обобщить. Но обобщить так, чтобы подходила хоть какая-то из гипотез из G.

Сеттинг

- Итак, есть два множества: G — множество максимальных (общих) гипотез, и S — множество минимальных (частных) гипотез.
- Когда приходит новый пример d, есть несколько возможностей:
- $t_d = 0$
- В этом случае, если оказывается, что гипотеза из S неверно классифицирует d, её нужно удалить.
- А если неверно классифицируют гипотезы из G, их нужно соответственно специализировать (добавить все минимальные специализации, совместные с d и такие, чтобы была соответствующая гипотеза в h).

Алгоритм Candidate Elimination

`CandidateElimination`(D)

- $H := \{\langle\emptyset, \ldots, \emptyset\rangle\}$.
- $G := \{\langle ?, \ldots, ?\rangle\}$.
- Для всех $d \in D$:
  - Если $t_d = 1$:
    - Для каждой $h \in G$, если $h(d) = 0$, $G := G \setminus \{h\}$.
    - Для каждой $h \in S$, если $h(d) = 0$, $S := S \setminus \{h\} \cup \mathrm{Gen}(h)$, где $\mathrm{Gen}(h) = \min\{h' \mid h' \succ h, h'(d) = 1, \exists h_g \in G : \ h_g \succ h'\}$.
  - Если $t_d = 0$:
    - Для каждой $h \in S$, если $h(d) = 1$, $S := S \setminus \{h\}$.
    - Для каждой $h \in G$, если $h(d) = 1$, $G := G \setminus \{h\} \cup \mathrm{Spe}(h)$, где $\mathrm{Spe}(h) = \max\{h' \mid h \succ h', h'(d) = 0, \exists h_s \in S : \ h' \succ h_s\}$.
- Выдать G, H.

Candidate elimination: пример

- Возьмём игры «Зенита» и тестовые примеры (в том числе негативные)

Соперник	Играем	Лидеры	Дождь	Победа
Выше	Дома	На месте	Нет	Да
Ниже	В гостях	Пропускают	Нет	Нет
Ниже	Дома	Пропускают	Да	Нет
Выше	Дома	Пропускают	Нет	Да

Candidate elimination: пример

- $H = \{\langle\emptyset, \emptyset, \emptyset, \emptyset\rangle\}$, $G := \{\langle ?, ?, ?, ?\rangle\}$.
- Первый пример: Выше Дома На месте Нет Да
- $H[0]$ выдаёт 0, значит, надо обобщать. Максимально частная гипотеза просто совпадёт с этим тестовым примером.

Candidate elimination: пример

- $H = \{\langle \text{Выше}, \text{Дома}, \text{На месте}, \text{Нет} \rangle\}$, $G := \{\langle ?, ?, ?, ? \rangle\}$.
- Второй пример:
  Ниже В гостях Пропускают Нет Нет
- $G[0]$ выдаёт 0 — надо специализировать. Максимально общие гипотезы, которые обобщают $H[0]$ и выдают 0:

$$\langle \text{Выше}, ?, ?, ? \rangle, \langle ?, \text{Дома}, ?, ? \rangle, \langle ?, ?, \text{На месте}, ? \rangle.$$

Candidate elimination: пример

- $H = \{\langle \text{Выше}, \text{Дома}, \text{На месте}, \text{Нет} \rangle\}$,
  $G = \{\langle \text{Выше}, ?, ?, ? \rangle, \langle ?, \text{Дома}, ?, ? \rangle, \langle ?, ?, \text{На месте}, ? \rangle\}$.
- Третий пример: Ниже Дома Пропускают Да Нет
- $G[1]$ выдаёт 1; $\text{Spe}(G[1]) =$
  $\{\langle \text{Выше}, \text{Дома}, ?, ? \rangle, \langle ?, \text{Дома}, \text{На месте}, ? \rangle, \langle ?, \text{Дома}, ?, \text{Нет} \rangle\}$.

Candidate elimination: пример

- $H = \{\langle \text{Выше}, \text{Дома}, \text{На месте}, \text{Нет} \rangle\}$,

$$G = \{\langle \text{Выше}, ?, ?, ? \rangle, \langle ?, ?, \text{На месте}, ? \rangle, \langle ?, \text{Дома}, ?, \text{Нет} \rangle\}.$$

- Четвёртый пример:

Выше	Дома	Пропускают	Нет	Да

- $G[1]$ выдаёт 0, поэтому её удаляем.
- $H[0]$ надо обобщить, получим $\langle \text{Выше}, \text{Дома}, ?, \text{Нет} \rangle$.

Candidate elimination: пример

- Итого получаем максимально частную гипотезу, объясняющую все тестовые примеры:

$$h = \langle ?, \text{Дома}, ?, \text{Нет} \rangle$$

и набор максимально общих гипотез, совместных со всеми данными:

$$G = \{ \langle \text{Выше}, ?, ?, ? \rangle, \langle ?, \text{Дома}, ?, \text{Нет} \rangle \}.$$

За и против

- Хорошо работает, если целевая функция содержится во множестве возможных гипотез.
- В противном случае может сойтись к пустому множеству.
- Это фактически единственный недостаток (но очень серьёзный, потому что множество гипотез довольно невыразительное).

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

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

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

Применяем теорему Байеса

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

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

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

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:

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

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

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.
- Как мы уже выясняли, когда анализировали деревья принятия решений,

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

- То есть каждая гипотеза, совместимая со всеми данными — максимальная апостериорная гипотеза. Пока вроде бы ничего нового.

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

$$\mathrm{argmax}_{v \in V} \sum_{h \in H} p(v|h)p(h|D).$$

Оптимальный классификатор

Определение

Любой алгоритм, который решает задачу

$$\mathrm{argmax}_{v\in V} \sum_{h\in H} p(v|h)p(h|D),$$

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

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

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.
- Результаты метода сравнимы с результатами обучения нейронных сетей или деревьев принятия решений.

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.

$$v_{\mathrm{MAP}} = \mathrm{argmax}_{v \in V} p(x = v | a_1, a_2, \ldots, a_n).$$

- По теореме Байеса,

$$v_{\mathrm{MAP}} = \mathrm{argmax}_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$

$$= \mathrm{argmax}_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

Вывод формул

- По теореме Байеса,

$$v_{\mathrm{MAP}} = \mathrm{argmax}_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \mathrm{argmax}_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

- 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,

$$v_{\mathrm{MAP}} = \mathrm{argmax}_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$
$$= \mathrm{argmax}_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

- 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,

$$v_{\mathrm{MAP}} = \mathrm{argmax}_{v \in V} \frac{p(a_1, a_2, \ldots, a_n | x = v) p(x = v)}{p(a_1, a_2, \ldots, a_n)} =$$

$$= \mathrm{argmax}_{v \in V} p(a_1, a_2, \ldots, a_n | x = v) p(x = v).$$

Итак, наивный байесовский классификатор выбирает v как

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

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.
- Мы сейчас не будем этим подробно заниматься; см. (Domingos and Pazzani, 1997).

Спасибо за внимание!

- Lecture notes и слайды будут появляться на моей homepage:
  http://logic.pdmi.ras.ru/~sergey/index.php?page=teaching
- Присылайте любые замечания, решения упражнений, новые численные примеры и прочее по адресам: sergey@logic.pdmi.ras.ru, snikolenko@gmail.com
- Заходите в ЖЖ smartnik.