Наивный Байесовский классификатор

Александр Сизов, Сергей Николенко

Центр
Речевых
Технологий

Центр Речевых Технологий

Outline

Центр Речевых Технологий

Классификация

- В машинном обучении под классификацией понимают задачу определения категории, к которой принадлежит ранее не встречавшийся образец, на основании обучающего множества, для элементов которого эти категории известны. Это является примером обучения с учителем (supervised learning).
- Существует множество подходов к классификации: деревья принятия решений, нейронные сети, вероятностные методы и т.д.

Центр Речевых Технологий

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,

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

Центр Речевых Технологий

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.

Центр Речевых Технологий

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...

Центр Речевых Технологий

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:

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

Центр Речевых Технологий

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

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

- За счет предположения независимости параметры каждого атрибута могут быть обучены отдельно, и это значительно упрощает обучение. Особенно когда количество атрибутов велико. Например, в классификации текстов.
- В парадигме классификации текстов мы предполагаем, что разные слова в тексте на одну и ту же тему появляются независимо друг от друга.

Центр Речевых Технологий

Outline

Центр Речевых Технологий

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.

Центр Речевых Технологий

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.

Центр Речевых Технологий

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 $p(w_t \mid c_j)$.

Центр Речевых Технологий

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):

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

Центр Речевых Технологий

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

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

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

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

Центр Речевых Технологий

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.

Центр Речевых Технологий

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 $p(w_t \mid c_j)$.

Центр Речевых Технологий

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):

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

Центр Речевых Технологий

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

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

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

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

Сравнение

Многомерная	Мультиномиальная
Документ - Событие Наличие или отсутствие слов - атрибуты события	Документ - набор событий Событие - наличие индивидуального слова
Легко добавлять нетекстовые признаки	Нельзя добавлять напрямую в словарь
Плохо с этим справляются, т.к. больше вероятность, что слово возникнет в длинном тексте, вне зависимости от класса	Более точная классификация для данных с большой дисперсией длины текстов

Неявный учет отсутствующих слов

- Может показаться, что у многомерной модели есть некоторое преимущество в том, что она учитывает обоснованность (evidence) тех слов, которые не встретились в тексте.
- Однако, это не совсем так. Мультиномиальная модель неявно учитывает эту информацию в вероятностных распределениях для каждого класса.
- Например, если рассматривать тексты из области распознавания диктора, то слово *speaker* будет одним из самых популярных. Т.е. будет иметь высокую вероятность для этого класса текстов и заберет часть вероятностной массы у других слов. Соответственно, если оно не встретиться в тексте, то в нем будут только слова с меньшей вероятностью для этого класса и это снизит шансы текста быть из области распознавания диктора.

Производительность

На основе экспериментов, проведенных McCallum and Nigam (1998), можно заключить, что многомерная модель если и имеет меньшую ошибку предсказания, то только на маленьких словарях (несколько сотен слов). Когда размер словарей измеряется тысячами слов, то всегда выигрывает мультиномиальная. В среднем ее ошибка предсказаний на 27% меньше.

Центр Речевых Технологий

Outline

Классификация semi-naive методов

Хоть NB и показывает неплохие результаты, было разработано множество методов, которые пытались улучшить точность предсказаний за счет „смягчения“ предположения об условной независимости атрибутов. Эти методы можно разделить на две категории:

1. Первые применяют NB к модифицированному множеству атрибутов, которое получается в результате удаления и объединения исходных атрибутов.
2. Вторые добавляют связи между атрибутами. Т.е. учитывают зависимости между атрибутами. Эту группу можно также разделить на
   - eager learning methods, которые обучаются в фазе обучения.
   - lazy learning methods, которые откладывают обучения до начала классификации новых данных.

Центр Речевых Технологий

Зависимость между атрибутами

Naive Bayes использует все атрибуты для классификации. Однако, когда два атрибута сильно зависимы друг от друга, то NB может переоценить их совместное влияние и это может увеличить ошибку предсказания.
Рассмотрим это на примере. Пусть имеются три атрибута, тогда:

$$p(x = v|a_1, a_2, a_3) \propto p(v)p(a_1|v)p(a_2|v)p(a_3|v).$$

Если добавить $a_4 = a_2$, то

$$p(x = v|a_1, a_2, a_3, a_4) \propto p(v)p(a_1|v)p(a_2|v)^2p(a_3|v).$$

Backwards Sequential Elimination (BSE)

- Метод BSE оставляет только подмножество изначальных атрибутов.
- Это достигается с помощью последовательного удаления атрибутов, каждый раз выбирая тот, удаление которого сильнее всего улучшит точность классификатора.
- Процесс останавливается, когда уже больше нельзя достичь улучшения точности.
- Классификация производится как и в NB, только на выбранном множестве атрибутов ($\bar{a} = a_{g_1}, \ldots, a_{g_h}$):

$$v_{BSE}(a_{g_1}, \ldots, a_{g_h}) = \arg\max_{v \in V} p(x = v) \prod_{i=g_1}^{g_h} p(a_i | x = v).$$

Forward Sequential Selection (FSS)

FSS отличается от предыдущего метода только тем, что начинает с пустого множества атрибутов и на каждом шаге добавляет по одному атрибуту.

Backward Sequential Elimination and Joining

- Еще одним подходом при обнаружении зависимостей между атрибутами, является создание составных атрибутов. BSEJ использует точность предсказаний в качестве критерия объединения двух атрибутов. В результате получается новых атрибут, принимающий значения из их Декартова произведения.
- Во время работы алгоритм объединяет или удаляет атрибуты, исходя из того, какое действие сильнее всего улучшит точность предсказаний.
- Процесс останавливается, когда уже больше нельзя достичь улучшения точности.

Backward Sequential Elimination and Joining

- Если $a_{l_1}, \ldots, a_{l_q}$ — оставшееся подмножество исходных атрибутов, а $join_{g_1}, \ldots, join_{g_h}$ — новые атрибуты, то

$$v_{BSEJ}(a_{l_1}, \ldots, a_{l_q}, join_{g_1}, \ldots, join_{g_h}) =$$

$$\arg\max_{v \in V} p(x = v) \prod_{i=g_1}^{g_h} p(join_i | x = v) \prod_{r=l_1}^{l_q} p(a_r | x = v).$$

Центр Речевых Технологий

x-зависимость

Определение

Байесовский классификатор называется x-зависимым (x-dependent), если каждый атрибут зависит от класса и от не более чем x других атрибутов.

NB и три предыдущих метода являются 0-зависимым. Следующие 5 методов, которые мы рассмотрим, будут x-зависимыми ($x \geq 1$).

Центр Речевых Технологий

Tree Augment Naive Bayes (TAN)

Этот метод устанавливает зависимости между атрибутами на основе следующего алгоритма:

- Для каждого класса v вычисляется $I(a_i, a_j|v)$ - взаимная информация между всеми парами атрибутов.
- Затем стоится неориентированный граф, вершинами которого являются атрибуты, а ребро между a_i и a_j имеет вес $I(a_i, a_j|v)$.
- В этом графе находится максимальное покрывающее дерево. После чего в произвольном порядке расставляются направления ребер.

Tree Augment Naive Bayes (TAN)

Этот метод устанавливает зависимости между атрибутами на основе следующего алгоритма:

- Для каждого класса v вычисляется $I(a_i, a_j|v)$ - взаимная информация между всеми парами атрибутов.
- Затем стоится неориентированный граф, вершинами которого являются атрибуты, а ребро между a_i и a_j имеет вес $I(a_i, a_j|v)$.
- В этом графе находится максимальное покрывающее дерево. После чего в произвольном порядке расставляются направления ребер.
- Если $\pi(a_i)$ - это родитель a_i. Тогда

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

SuperParent TAN (SP-TAN)

SP-TAN — это вариация TAN, в которой используется другой подход для построения функции $\pi(a_i)$.

Определение

Сирота (orphan) — это любой атрибут, у которого нет родителя.

Определение

Если провести дуги из вершины a_i до всех атрибутов из множества сирот, то a_i будет называться супер-родителем (SuperParent).

SuperParent TAN (SP-TAN)

Определение

Если вершину a_i по очереди соединять с каждым сиротой и тестировать точность предсказаний в таком графе, то сирота с лучшей связью называется любимым потомком (FavoriteChild)

Алгоритм построения зависимостей между атрибутами следующий:

1. Выбираем супер-родителя, дающего лучшие предсказания.
2. Выбираем любимого потомка для этого супер-родителя.
3. Если ребро между ними улучшает предсказания, то фиксируем его и возвращаемся к п.1

SuperParent TAN (SP-TAN)

Классификация в SP-TAN производится также как и в TAN:

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

SuperParent TAN (SP-TAN)

Классификация в SP-TAN производится также как и в TAN:

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

Тем не менее можно выделить несколько отличий между TAN и SP-TAN:

- TAN добавляет N-1 ребро, тогда как SP-TAN может иметь и меньше зависимостей между атрибутами.
- TAN выбирает направления ребер произвольно, а в SP-TAN они всегда идут от супер-родителя к любимому потомку.

Центр Речевых Технологий

NBTree

NBTree — это комбинация NBC и дерева принятие решений. Идея состоит в том, что на основании значений некоторых атрибутов мы разделяем данные, так, что получается дерево. А затем в каждом листе этого дерева создаем локальный NBC.

NBTree

NBTree — это комбинация NBC и дерева принятие решений. Идея состоит в том, что на основании значений некоторых атрибутов мы разделяем данные, так, что получается дерево. А затем в каждом листе этого дерева создаем локальный NBC.

Определение

Полезность узла (utility of a node) — точность предсказаний NBC в этом узле, определяемая с помощью 5-fold cross-validation test.

NBTree

NBTree — это комбинация NBC и дерева принятие решений. Идея состоит в том, что на основании значений некоторых атрибутов мы разделяем данные, так, что получается дерево. А затем в каждом листе этого дерева создаем локальный NBC.

Определение

Полезность узла (utility of a node) — точность предсказаний NBC в этом узле, определяемая с помощью 5-fold cross-validation test.

Определение

Полезность разделения (utility of a split) — взвешенная сумма полезностей получаемых в итоге разделения узлов. Веса узлов пропорциональны количеству попавших в них данных.

Центр Речевых Технологий

Рекурсивный алгоритм работы NBTree

- Для каждого атрибута a_i вычислить полезность разделения по этому атрибуту $u(a_i)$.
- Пусть $j = \arg\max_i u(a_i)$.
- Если $u(a_j)$ не является значимо лучше, чем полезность текущего узла, создать NBC в текущем узле и вернуться на уровень выше. (Разделение является значимым, если относительное уменьшение ошибки не менее 5% и разделяемый атрибут содержит не менее 30 экземпляров).
- Разделить данные на основании атрибута a_j.
- Для каждого дочернего узла рекурсивно вызвать алгоритм, на той части данных, которые совпадают со значением атрибута.

Центр Речевых Технологий

Классификация с помощью NBTree

Пусть $S = S_1, \ldots, S_g$ — множество тестовых атрибутов на пути к листу дерева, а $R = R_1, \ldots, R_{n-g}$ — множество оставшихся атрибутов, тогда:

$$v_{NBTree}(R|s) = \arg\max_{v \in V} p(x = v|s) \prod_{i=1}^{n-g} p(r_i|x = v, s).$$

Где $s \in S, r_j \in R_j$.

Lazy Bayesian Rules (LBR)

- Результаты работы LBR похожи на NBTree, только LBR использует lazy learning и для каждого тестового примера генерирует новый путь в дереве решений.
- Классификация производится аналогично NBTree:

$$v_{LBR}(R|s) = \arg\max_{v \in V} p(x = v|s) \prod_{i=1}^{n-g} p(r_i|x = v, s).$$

- Так как NBTree строит одно дерево для всей обучающей выборки, то в нем могут образоваться листья, содержащие всего несколько элементов из выборки. И это может ухудшить классификацию. С этой проблемой и борется LBR.
- LBR хорошо подходит, когда надо классифицировать немного элементов.

Averaged One-Dependence Estimators (AODE)

AODE усредняет предсказания всех подходящих 1-зависимых классификаторов. Это делается, чтобы:

- не производить выбор модели (model selection),
- сохранить эффективность 1-зависимых классификаторов.

Пусть E — тестовый элемент. Для $\forall$ значения атрибута a_j,

$$p(x = v, E) = p(x = v, a_j)p(E|x = v, a_j).$$

Поэтому,

$$p(x = v, E) = \frac{\sum_{j:1 \le j \le n \wedge F(a_j) \ge m} p(x = v, a_j)p(E|x = v, a_j)}{|j : 1 \le j \le n \wedge F(a_j) \ge m|},$$

где $F(a_j)$ — частота a_j в базе обучения, а $m = 30$.

Averaged One-Dependence Estimators (AODE)

$$p(x = v, E) = \frac{\sum_{j:1 \le j \le n \wedge F(a_j) \ge m} p(x = v, a_j) p(E|x = v, a_j)}{|j : 1 \le j \le n \wedge F(a_j) \ge m|},$$

где $F(a_j)$ — частота a_j в базе обучения, а $m = 30$.

- Если $p(a_j)$ — мала, то оценка $p(E|x = v, a_j)$ может быть ненадежной. Для этого выбирается порог m.
- Классификация производится по формуле:

$$v_{AODE}(a_1, a_2, \ldots, a_n|x = v) =$$

$$= \arg\max_{v \in V} \sum_{j:1 \le j \le n \wedge F(a_j) \ge m} p(x = v, a_j) \prod_{h=1}^{n} p(a_h|x = v, a_j).$$

Центр Речевых Технологий

Вычислительная сложность

	Обучение		Классификация	
Алгоритм	Время	Память	Время	Память
NB	$O(nt)$	$O(knv)$	$O(kn)$	$O(knv)$
BSE и FSS	$O(tkn^2)$	$O(tn + knv)$	$O(kn)$	$O(knv)$
BSEJ	$O(tkn^3)$	$O(tn + kv^n)$	$O(kn)$	$O(kv^n)$
TAN	$O(tn^2 + kn^2v^2 + n^2 \log n)$	$O(k(nv)^2)$	$O(kn)$	$O(knv^2)$
SP-TAN	$O(tkn^3)$	$O(tn + k(nv)^2$	$O(kn)$	$O(knv^2)$
NBTree	$O(t^2kn^2/v)$	$O(tk(n - log_v t)v)$	$O(kn)$	$O(tk(n - \log_v t)v)$
LBR	$O(tn)$	$O(tn)$	$O(tkn^2)$	$O(tn)$
AODE	$O(tn^2)$	$O(k(nv)^2)$	$O(kn^2)$	$O(k(nv)^2)$

k — количество классов
n — количество атрибутов
v — среднее количество значений для атрибута
t — количество элементов в обучающей базе

Центр Речевых Технологий

Сравнение качества работы

В результате крупномасштабных экспериментов (Zheng and Webb,2005) были получены следующие результаты:

- Алгоритмы с наименьшим смещением: NBTree, BSJE, LBR, (AODE, TAN), SP-TAN.
- Алгоритмы с наименьшей дисперсией: NB, AODE, (LBR, SP-TAN, BSE).
- При выборе алгоритма можно исходить из того, что для больших наборов обучающих данных основной вклад в ошибку вносит смещение, а при малых — дисперсия.
- AODE — неплохой „среднячок“.

Центр Речевых Технологий

Outline

Центр Речевых Технологий

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.

Центр Речевых Технологий

Объяснения качества NB

- 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.
  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.

Центр Речевых Технологий

Недостатки объяснений

- Однако, объяснение в пунке 2 достаточно поверхностное, потому что не показывает, что мешает сильным зависимостям повлиять на классификацию.
- Ключевым моментом является то, что нужно знать, как зависимости влияют на классификацию и при каких условиях они не влияют на классификацию.
- Сейчас будт показано, что классификация NB зависит, не от самих зависимостей между переменными, а от распределения этих зависимостей.

Повторение

Предположим, что у нас есть только два класса: $V = \{+, -\}$. $E = (a_1, \ldots, a_n))$ — пример для классификации. Тогда E сопоставляется классу $C = +$ тогда и только тогда, когда

$$f_b(E) = \frac{p(C = +|E}{p(C = -|E)} \geq 1$$

Если сделать наивное Байесовское предположение, то получим

$$f_{nb}(E) = \frac{p(C = +}{p(C = -)} \prod_{i=1}^{n} \frac{p(a_i|C = +)}{p(a_i|C = -)} \geq 1$$

Центр Речевых Технологий

Augmented naive Bayes

- NB — это простейшая форма Байесовской сети, в которой все атрибуты независимы друг от друга при условии класса.
- Если несколько расширить эту структуру и включить в явном виде зависимости между атрибутами, то получим augmented naive Bayes network(ANB).
- Пусть G — это ANB, тогда

$$p_G(a_1, \ldots, a_n, v) = p(v) \prod_{i=1}^{n} p(a_i | \pi(a_i), v)$$

- Доказано, что любая Байесовская сеть может быть представлена в виде ANB. То есть любое совместное распределение может быть представленно в виде ANB.

Центр Речевых Технологий

Augmented naive Bayes

- Пусть G — это ANB, тогда

$$p_G(a_1, \ldots, a_n, v) = p(v) \prod_{i=1}^{n} p(a_i | \pi(a_i), v)$$

- Доказано, что любая Байесовская сеть может быть представлена в виде ANB. То есть любое совместное распределение может быть представленно в виде ANB.
- Таким образом, мы можем выбрать ANB как истинное вероятностное распределение, и попытаться установить, при каких условиях NB классифицирует в точности также, как и истинное распределение.

Центр Речевых Технологий

Эквивалентность

Определение 1

Пусть есть образец E. Два классификатора f_1 и f_2 называются равными на E при условии 0-1 потерь (zero-one loss), если $f_1(E) \geq 1 \Leftrightarrow f_2(E) \geq 1$, обозначается $f_1(E) \doteq f_2(E)$. Если это выполнятеся для любого E, то f_1 и f_2 называются равными при условии 0-1 потерь, $f_1 \doteq f_2$.

Центр Речевых Технологий

Локальная зависимость

Определение

Зависимость между узлом и его родителем называется локальной зависимостью узла.

Определение 2

Для узла a в графе G, являющемся ANB, производные локальной зависимости (local dependence derivative) узла a в классах $\{+, -\}$ определяются как:

$$dd_G^+(a|\pi(a)) = \frac{p(a|\pi(a), +)}{p(a|+)},$$

$$dd_G^-(a|\pi(a)) = \frac{p(a|\pi(a), -)}{p(a|-)}.$$

Центр Речевых Технологий

Отношение производных

Определение 3

Для узла a в графе G, являющемся ANB, отношение производных локальной зависимости (local dependence derivative ratio) узла a в классах $\{+, -\}$ определяттся как:

$$ddr_G(a) = \frac{dd_G^+(a|\pi(a))}{dd_G^-(a|\pi(a))}.$$

Центр Речевых Технологий

Распределение зависимостей

Теорема

Пусть для атрибутов $(A_1, \ldots, A_n)$ *задан ANB граф* G *и соответствующий ему NB граф* G_{nb}. *Предположим, что* f_b *и* f_{nb} *— классификаторы, соответствующие* G *и* G_{nb}. *Для любого образца* $E = (a_1, \ldots, a_n)$ *выполняется*

$$f_b(a_1, \ldots, a_n) = f_{nb}(a_1, \ldots, a_n) \prod_{i=1}^{n} ddr_G(a_i),$$

где $\prod_{i=1}^{n} ddr_G(a_i)$ *называется фактором распределения зависимостей образца* E *и обозначается* $DF_G(E)$.

Центр Речевых Технологий

Теорема 2

Теорема

Пусть дан образец $E = (a_1, \ldots, a_n)$, *ANB граф* G *равен соответствующему naive Bayes графу* G_{nb} *при условии 0-1 потерь, т.е.* $f_b(E) \doteq f_{nb}(E)$ *тогда и только тогда, когда либо* $f_b(E) \geq 1$ *и* $DF_G(E) \leq f_b(E)$ *либо,* $f_b(E) < 1$ *и* $DF_G(E) > f_b(E)$.

Центр Речевых Технологий

Результаты

1. Когда $DF_G(E) = 1$ зависимости в ANB графе G не влияют на классификацию. $DF_G(E) = 1$ в трех случаях:
   - зависимостей между атрибутами не существует.
   - для $\forall$ атрибута $A \in G$, $ddr_G(a) = 1$, т.е. локальное распределение каждого атрибута распределено равномерно в обоих классах.
   - Влияние, которое оказывают некоторые локальные зависимости в поддержку класса $V = +$ сокращается влиянием, оказываемым другими локальными зависимостями в поддержку класса $V = -$.

Центр Речевых Технологий

Результаты

1. Когда $DF_G(E) = 1$ зависимости в ANB графе G не влияют на классификацию. $DF_G(E) = 1$ в трех случаях:
   - зависимостей между атрибутами не существует.
   - для $\forall$ атрибута $A \in G$, $ddr_G(a) = 1$, т.е. локальное распределение каждого атрибута распределено равномерно в обоих классах.
   - Влияние, которое оказывают некоторые локальные зависимости в поддержку класса $V = +$ сокращается влиянием, оказываемым другими локальными зависимостями в поддержку класса $V = -$.
2. Чтобы выполнялось $f_b(E) \doteq f_{nb}(E)$, не требуется $DF_G(E) = 1$. Этот факт объясняет, почему NB дает хорошие результаты даже для data set-ов с сильными зависимостями между атрибутами.

Центр Речевых Технологий

Результаты

1. Когда $DF_G(E) = 1$ зависимости в ANB графе G не влияют на классификацию. $DF_G(E) = 1$ в трех случаях:
   - зависимостей между атрибутами не существует.
   - для $\forall$ атрибута $A \in G$, $ddr_G(a) = 1$, т.е. локальное распределение каждого атрибута распределено равномерно в обоих классах.
   - Влияние, которое оказывают некоторые локальные зависимости в поддержку класса $V = +$ сокращается влиянием, оказываемым другими локальными зависимостями в поддержку класса $V = -$.
2. Чтобы выполнялось $f_b(E) \doteq f_{nb}(E)$, не требуется $DF_G(E) = 1$. Этот факт объясняет, почему NB дает хорошие результаты даже для data set-ов с сильными зависимостями между атрибутами.
3. Зависимости в графе ANB влияют на классификацию, только в случае, когда не выполняются условия теоремы 2.

Центр Речевых Технологий

Результаты

Таким образом, теорема 2 дает нам необходимые и достаточные условия оптимальности наивного Байесовского классификатора для образца E. Если условия теоремы выполняются для всех E, то $f_b \doteq f_{nb}$. Т.е. NB является глобально оптимальным.

Центр Речевых Технологий

Thank you!

Спасибо за внимание!