Лекция 9. Байесовские методы классификации

Сергей Лебедев

sergei.lebedev@jetbrains.com

10 апреля 2015 г.

Томас Байес и Пьер-Симон Лаплас[1]

[1] http://wikipedia.org/wiki/Bayesian_probability

- Датасет *20 newsgroups*[2] содержит почти 20000 сообщений из списков рассылки Usenet.
- Примеры сообщений из списка рассылки `sci.crypt`:

```
When you find out a floppy password protect program,
could you e-mail me.
Thanks

Not to mention Computer Associates.  I'll have to be
careful to stop telling people I'm a Clipper
programmer, they might lynch me... :-)
```

- Построим классификатор, предсказывающий по тексту сообщения список рассылки, в который оно было отправлено.

[2] http://qwone.com/~jason/20Newsgroups

Вероятностная постановка задачи классификации

- Пусть X — множество объектов, а Y — множество меток классов и $X \times Y$ — вероятностное пространство с плотностью
$$p(x, y) = P_y p(x|y),$$
- тогда задача обучения классификатора по выборке $X^l = (x_i, y_i)_{i=1}^l$ сводится к поиску функции $a : X \to Y$, минимизирующей **вероятность** ошибки.
- Немного терминологии:

P_y	априорная вероятность класса y
$p(x\|y)$	функция правдоподобия класса y

- Как правило, априорные вероятности P_y и функции правдоподобия классов $p(x|y)$ не известны,
- поэтому задача классификации делится на две подзадачи.
  1. По выборке X^l из неизвестного распределения с плотностью $p(x, y)$ построить оценки вероятностей $\hat{P}_y$ и функций правдоподобия $\hat{p}(x|y)$ для каждого класса.
  2. По известным P_y и $p(x|y)$ построить функцию $a(x)$, минимизирующую вероятность ошибочной классификации.

Вопрос

Предположим, что нам известно распределение с плотностью $p(x, y)$. Как оценить вероятность ошибочной классификации для произвольного алгоритма $a : X \to Y$?

- Для каждой пары $(y, s) \in Y^2$ введём λ_{ys} — штраф за назначение класса s объекту класса y.
- На самом деле, "полезные" нам значения λ_{ys} такие, что $\lambda_{yy} = 0$ и $\lambda_{ys} > 0$ для всех $y \neq s$.
- Функционал среднего риска для алгоритма $a : X \to Y$:

$$R(a) = \sum_{s \in Y} \sum_{y \in Y} \lambda_{ys} P_y p(A_s|y) \qquad p(A_s|y) = \int_{A_s} p(x|y) dx,$$

  где $A_s = \{x \in X \mid a(x) = s\}$ — множество объектов класса y, которым алгоритм ошибочно назначил класс s.
- Если положить $\lambda_{ys} = [y \neq s]$, то $R(a)$ — это вероятность ошибки алгоритма $a(x)$. Кстати, **почему**?

- Если известны априорные вероятности P_y и функции правдоподобия $p(x|y)$, то минимум среднего риска $R(a)$ достигается при:

$$a(x) = \arg\min_{s \in Y} \sum_{y \in Y} \lambda_{ys} P_y p(x|y)$$

- А если величина штрафа зависит только от класса y, то есть $\lambda_{yy} = 0$ и $\lambda_{ys} = \lambda_y$ для всех $y \neq s$, то оптимальный алгоритм можно переписать как:

$$a(x) = \arg\max_{y \in Y} \lambda_y P_y p(x|y)$$

- Апостериорной вероятностью класса y для объекта x называют

$$P(y|x) = \frac{p(x,y)}{p(x)} = \frac{P_y p(x|y)}{\sum\limits_{s \in Y} P_s p(x|s)} \propto P_y p(x|y)$$

- Перепишем оптимальный алгоритм с использованием апостериорных вероятностей:

$$a(x) = \arg\max_{y \in Y} \lambda_y P(y|x)$$

- Если $\lambda_y = 1$, то алгоритм просто максимизирует апостериорную вероятность для объекта x.

- Задачу классификации можно интерпретировать как поиск алгоритма $a(x)$, минимизирующего вероятность ошибки или в более общем смысле средний риск $R(a)$.
- Задав штраф за ошибку классификации λ_y на объекте класса y и плотность совместного распределения $p(x,y)$, можно построить оптимальный в смысле минимизации $R(a)$ алгоритм:

$$a(x) = \arg\max_{y \in Y} \lambda_y P(y|x)$$

- Если $\lambda_y = 1$, то $a(x)$ назначает объекту x класс y, имеющий наибольшую среди всех классов апостериорную вероятность.

Наивность

- Мы умеем строить оптимальный Байесовский классификатор $a : X \to Y$ при условии, что известны P_y и $p(x|y)$,
- но до “работающего” классификатора нам ещё довольно далеко.
- Чего нужно сделать?
  - Понять, что представляют из себя X и Y для сообщений из списков рассылки.
  - Выбрать функции правдоподобия $p(x|y)$.
  - Научиться оценивать априорные вероятности классов $\hat{P}_y$ и функции правдоподобия $\hat{p}(x|y)$ из данных.

- Пусть $V = \{v_1, \ldots, v_{|V|}\}$ – упорядоченное множество слов *aka* словарь,
- тогда сообщение можно представить в виде вектора, в котором на j-той позиции стоит 1, если v_d встречается в сообщении, и 0 в обратном случае.
- То есть, $X \equiv \{0, 1\}^{|V|}$, а Y – множество идентификаторов списков рассылки.
- Пример:
    - $V = \{\text{who}, \text{I}, \text{let}, \text{dogs}, \text{out}, \text{the}\}$
    - Сообщение «Who let the dogs out? Who, who, who, who?» будет векторизовано как $[1, 0, 1, 1, 1, 1]$.
    - Как будет векторизовано предложение «Well, if I am a dog, the party is on [...]»?

Функция правдоподобия: первое приближение

- Как определить функцию правдоподобия для сообщения, представленного в виде бинарного вектора?
- Идея: будем использовать дискретное распределение на множестве X, то есть сопоставим вероятность θ_{yx} каждому значению $x \in X$, тогда

$$p(x|y) = \theta_{yx}$$

- Какие проблемы у такой функции правдоподобия?

- Предположим, что все признаки (компоненты вектора x) независимы **при условии** y[3], тогда:

$$p(x|y) = \prod_{d=1}^{|V|} p(x_d|y)$$

- Полученный классификатор называют **наивным** Байесовским классификатором из-за наивности сделанного предположения

$$a(x) = \arg\max_{y \in Y} \lambda_y P_y \prod_{d=1}^{|V|} p(x_d|y)$$

[3] То есть вся информация о зависимостях между признаками x_d содержится в классе y.

- Распределение Бернулли – дискретное распределение на множестве $\{0, 1\}$ с параметром $\theta \in [0, 1]$ – вероятностью “успеха” и функцией вероятности:

$$\mathrm{Ber}(x; \theta) = \theta^x (1 - \theta)^{1-x}$$

- Для нашего классификатора это означает, что

$$p(x|y) = \prod_{d=1}^{|V|} \theta_{yd}^x (1 - \theta_{yd})^{1-x}$$

Вопрос

Сколько параметров у построенного классификатора и как их можно оценить?

- Метод позволяет оценивать параметры функций правдоподобия по выборке X^l.
- Найдём ОМП для параметра распределения Бернулли:

$$L(\theta) = \sum_{i=1}^{l} \ln p(x_i; \theta) \to \max_{\theta}$$

- Если функция правдоподобия дифференцируема, то условие оптимума можно записать как:

$$\frac{\partial}{\partial \theta} L(\theta) = \sum_{i=1}^{l} \frac{\partial}{\partial \theta} \ln p(x_i; \theta) = \sum_{i=1}^{l} \frac{x_i}{\theta} + \frac{1 - x_i}{\theta - 1} \equiv 0$$

$$\Rightarrow \hat{\theta}_{\text{ML}} = \frac{1}{l} \sum_{i=1}^{l} x_i$$

Промежуточный итог: наивный Байесовский классификатор

- Оценим методом максимального правдоподобия априорные вероятности классов $\hat{P}_y$ и параметры распределения Бернулли $\hat{\theta}_{yd}$:

$$\hat{P}_y = \frac{\sum\limits_{i=1}^{l} [y_i = y]}{l} \qquad \hat{\theta}_{yd} = \frac{\sum\limits_{i=1}^{l} [y_i = y] x_{id}}{\sum\limits_{i=1}^{l} [y_i = y]}$$

- Подставим оценки в оптимальный алгоритм классификации:

$$a(x) = \arg\max_{y \in Y} \lambda_y \hat{P}_y \prod_{d=1}^{|V|} \mathrm{Ber}(x_{id}; \hat{\theta}_{yd})$$

Вопрос

Является ли полученный классификатор оптимальным в смысле минимизации среднего риска $R(a)$?

- В результате обучения наивного Байесовского классификатора на части данных *20 newsgroups* были получены следующие параметры:

$y \in Y$	$\hat{P}_y$	password	program	PGP
sci.crypt	0.4	0.8	0	1
comp.graphics	0.6	0.2	0.6	0

- Какой класс будет назначен сообщению «How should I add PGP support to my program?», если $\forall y \in Y (\lambda_y = 1)$?

$$\begin{aligned} a(x) &= \arg\max_{y \in Y} \hat{P}(y|x = [0, 1, 1]) \\ &= \arg\max_{y \in Y} \{\hat{p}(\text{sci.crypt}|x), \hat{p}(\text{comp.graphics}|x)\} \\ &= \arg\max_{y \in Y} \{0, 0\} \end{aligned}$$

- Идея: введём параметр $\alpha \geq 0$ и добавим его в ОМП для распределения Бернулли:

$$\hat{\theta}^*_{yd} = \frac{\sum\limits_{i=1}^{l} [y_i = y] x_{id} + \alpha}{\sum\limits_{i=1}^{l} [y_i = y] + 2\alpha}$$

- Если в обучающей выборке много представителей класса y, содержащих слово v_d, то $\hat{\theta}^*_{yd}$ будет стремиться к ОМП, в обратном случае $\hat{\theta}^*_{yd} \approx \frac{1}{2}$.

Вопрос

Как мотивировать использование параметра α?

- Фреквентистский подход предполагает, что параметры распределения некоторой случайной величины — это фиксированные (но, возможно, неизвестные) значения.
- Байесовский подход считает все величины случайными, то есть у параметров тоже есть распределение:

$$p(x|\theta) = \mathrm{Ber}(x|\theta) \qquad p(\theta) = \mathrm{Beta}(\theta|\alpha, \beta)$$

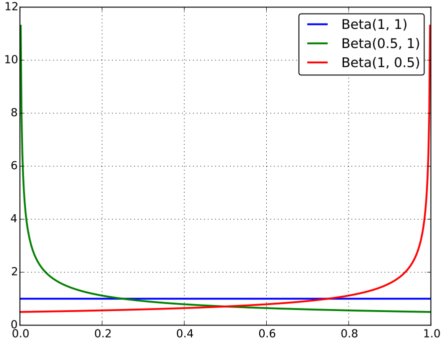
- Таким образом, при Байесовском подходе нас интересует не точечная оценка параметра $\hat{\theta}$, а его апостериорное распределение:

$$p(\theta|x) = \frac{p(\theta)p(x|\theta)}{\int p(\theta)p(x|\theta)d\theta} \propto p(\theta)p(x|\theta)$$

- В общем случае апостериорное распределение может иметь произвольный вид,
- но часто можно выбрать априорное распределение $p(\theta)$ таким образом, чтобы апостериорное распределение $p(\theta|x)$ имело тот же вид, что и априорное, только с другими параметрами.
- Чуть более формально: семейство распределений $p(\theta|\alpha)$ называется **априорным сопряжённым** для семейства функций правдоподобия $p(x|\theta)$, если апостериорное распределение $p(\theta|x, \alpha)$ остаётся в том же семействе:

$$p(\theta|x, \alpha) \propto p(\theta)p(x|\theta) = p(\theta|\alpha^*)$$

- α и α^* – это **гипер**параметры, то есть параметры распределения параметров.



$$p(\theta|\alpha,\beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha,\beta)} \qquad B(\alpha,\beta) = \int_0^1 \theta^{\alpha-1}(1-\theta)^{\beta-1}d\theta$$

- Бета-распределение является априорным сопряжённым для распределения Бернулли:

$$
\begin{aligned}
p(\theta|x,\alpha,\beta) &\propto p(\theta|\alpha,\beta)p(x|\theta) \\
&\propto \left(\theta^{\alpha-1}(1-\theta)^{\beta-1}\right)\left(\theta^{x}(1-\theta)^{1-x}\right) \\
&\propto \theta^{(\alpha+x)-1}(1-\theta)^{(\beta+1-x)-1} = \mathrm{Beta}(\theta|\alpha^*,\beta^*)
\end{aligned}
$$

- Оценим параметр θ с помощью мат. ожидания по апостериорному распределению:

$$
\begin{aligned}
\hat{\theta} &= \int\limits_0^1 \theta p(\theta|x,\alpha,\beta)d\theta = \mathbb{E}\left[\mathrm{Beta}(\theta|\alpha^*,\beta^*)\right] \\
&= \frac{\alpha^*}{\alpha^*+\beta^*} = \frac{\alpha+x}{\alpha+\beta+1}
\end{aligned}
$$

- Если $\alpha=\beta$, то $\hat{\theta}$ в точности совпадает со сглаженной оценкой $\hat{\theta}^*$!

Наивный Байесовский классификатор: резюме

- Предположение о независимости признаков при условии класса упрощает обучение Байесовского классификатора.
- Несмотря на сомнительность этого предположения, наивный Байесовский классификатор имеет свои преимущества:
  - не делает предположений о форме функции правдоподобия $p(x_d|y)$, например, для классификации текста также можно использовать мультиномиальное распределение,
  - этот классификатор просто реализовать и использовать,
  - его можно обучать по потоку данных (например, Twitter).

Регрессия

- Пусть $Y = \{-1, +1\}$, тогда наивный вариант оптимального алгоритма можно переписать как

$$a(x) = \arg\max_{y \in Y} \lambda_y P(y|x) = \operatorname{sign}\left(\lambda_{+1} P(+1|x) - \lambda_{-1} P(-1|x)\right)$$
$$= \operatorname{sign}\left(\frac{P(+1|x)}{P(-1|x)} - \frac{\lambda_{+1}}{\lambda_{-1}}\right)$$

- Рассмотрим отношение

$$\frac{P(+1|x)}{P(-1|x)} = \frac{P_+ p(x|+)}{P_- p(x|-)} = \frac{P_+}{P_-} \prod_{d=1}^{|V|} \frac{\theta_{+d}^{x_d} (1 - \theta_{+d})^{1-x_d}}{\theta_{-d}^{x_d} (1 - \theta_{-d})^{1-x_d}}$$
$$= \exp\left(\ln \frac{P_+}{P_-} + \sum_{d=1}^{|V|} x_d \ln \frac{\theta_{+d}}{\theta_{-d}} + (1 - x_d) \ln \frac{1 - \theta_{+d}}{1 - \theta_{-d}}\right)$$
$$= \exp\left(\sum_{d=1}^{|V|} x_d \left(\ln \frac{\theta_{+d}}{\theta_{-d}} - \ln \frac{1 - \theta_{+d}}{1 - \theta_{-d}}\right) - \operatorname{const}(x)\right)$$

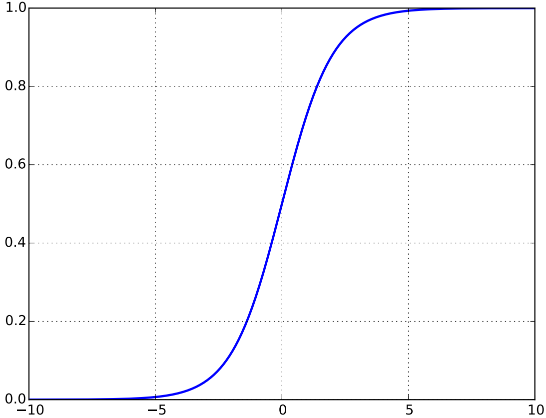
- Получили, что наивный Байесовский классификатор линеен:

$$\frac{P(+1|x)}{P(-1|x)} = e^{\langle w,x\rangle} \Rightarrow a(x) = \operatorname{sign}\left(e^{\langle w,x\rangle} - \underbrace{\ln\frac{\lambda_{+1}}{\lambda_{-1}}}_{w_0}\right)$$

- Выразим апостериорные вероятности в терминах $\langle w, x\rangle$, воспользовавшись формулой полной вероятности:

$$\begin{array}{l} P(+1|x) = \dfrac{1}{1+e^{-\langle w,x\rangle}} \\ P(-1|x) = \dfrac{1}{1+e^{+\langle w,x\rangle}} \end{array} \Rightarrow P(y|x) = \frac{1}{1+e^{-\langle w,x\rangle y}} \doteq \sigma(\langle w,x\rangle y)$$

- Можно обобщить этот результат на более широкий класс распределений, но мы этим заниматься не будем.



Сигмоида $\sigma(z)$ — S-образная функция, отображающая вещественные числа в отрезок $[0, 1]$

- Логистическая регрессия — это линейный классификатор с логарифмической функцией потерь.
- Чтобы в этом убедиться, выпишем логарифм правдоподобия $L(w)$ выборки и сравним его с функционалом эмпирического риска $Q(w)$[4].

$$L(w) = \ln \prod_{i=1}^{l} p(x_i, y_i) = \sum_{i=1}^{l} \underbrace{\ln P_y}_{\text{const}(w)} - \ln \sigma(\langle w, x_i \rangle y_i)$$

$$\to \max_w$$

$$Q(w) = \sum_{i=1}^{l} [M_i(w) < 0] \leq \sum_{i=1}^{l} \mathcal{L}(M_i(w)) = \sum_{i=1}^{l} \ln(1 + e^{-M_i(w)})$$

$$\to \min_w$$

[4] Напоминание: $M_i(w) \doteq \langle w, x_i \rangle y_i$ — отступ для объекта i.

- Максимизировать логарифм правдоподобия аналитически не представляется возможным, но можно воспользоваться, например, методом стохастического градиента[5]:

$$\begin{aligned} w^{(t+1)} &= w^{(t)} + \alpha x_i y_i \ln(1 - \sigma(\langle w, x_i \rangle y_i)) \\ &= w^{(t)} + \alpha x_i y_i \ln(1 - P(y_i|x_i)), \end{aligned}$$

- Разумеется, стоит аккуратно подходить к выбору α и использовать регуляризацию, чтобы избежать переобучения $\|w\| \to \infty$.

[5] N. B. Существуют и другие способы численной оптимизации, которые можно применить для этой задачи, например, метод Ньютона-Рафсона.

Пример реализации стохастического градиента для ЛР

```python
import numpy as np
from scipy.special import expit


def fit_lr(X, y, *, alpha, k=1, n_iter=1000):
    n_samples, n_features = X.shape
    w = init_weights(n_features)
    errors = []
    for i in range(n_iter):
        indices = np.random.choice(
            n_samples, k, replace=False)
        Xk, yk = X[indices, :], y[indices]

        Mk = Xk.dot(w) * yk
        errors.append(np.log1p(np.exp(-Mk)).mean())
        grad = expit(-Mk[:, np.newaxis])
        w += alpha * \
            (Xk * yk[:, np.newaxis] * grad).sum(axis=0)
    return w, errors
```

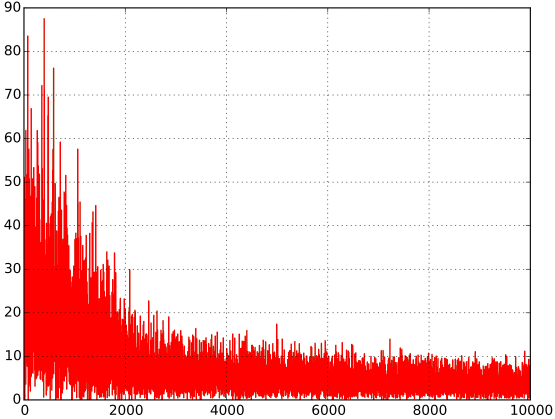
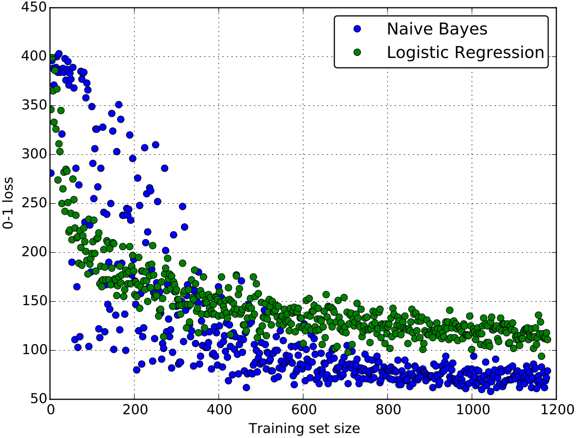


График функционала эмпирического риска для запуска стохастического градиентного подъёма с параметрами $\alpha = 10^{-6}$ и $k = 10$ на данных списков рассылки sci.crypt и comp.graphics

- Логистическую регрессию можно рассматривать
  - как линейный классификатор с логарифмической функцией потерь,
  - как наивный Байесовский классификатор.
- В пределе обе интерпретации эквивалентны, на практике же линейный подход
  - может лучше работать, когда предположения наивного Байесовского классификатора не выполняются[6];
  - может переобучаться, если размер обучающей выборки невелик.
- N. B. Мы обсудили только вариант логистической регрессии для двух классов, но, разумеется, модель можно обобщить и на количество классов ≥ 2.

[6] Кстати, за счёт чего ЛР так может?

Времяшоу



Зависимость количества неверных предсказаний от размера обучающей выборки, построенная по данным списков рассылки sci.crypt и comp.graphics