Машинное обучение, ФКН ВШЭ
Семинар №14

1 Метод опорных векторов

Задача 1.1. *Рассмотрим двумерную задачу классификации ($d = 2$) с двумя классами $\mathbb{Y} = \{-1, +1\}$. В обучающей выборке 5 точек: три точки $\{(1,1),(1,2),(2,3)\}$ класса $+1$ и две точки $\{(3,1),(4,2)\}$ класса -1. Найдите линейный классификатор $a(x) = \operatorname{sign}(w_1x_1 + w_2x_2 + b)$, который будет получен в результате обучения методом опорных векторов.*

Решение. Выборка является линейно разделимой, поэтому можно решать следующую задачу:

$$\begin{cases} \frac{1}{2}(w_1^2 + w_2^2) \to \min_w \\ y_i(\langle w, x_i \rangle + b) \geqslant 1. \end{cases}$$

Запишем лагранжиан:

$$L(w, b, \lambda) = \frac{1}{2}(w_1^2 + w_2^2) + \sum_{i=1}^{5} \lambda_i \left(1 - y_i(\langle w, x_i \rangle + b)\right)$$

и условия Куна–Таккера:

$$\begin{cases} w_1 - \lambda_1 - \lambda_2 - 2\lambda_3 + 3\lambda_4 + 4\lambda_5 = 0; \\ w_2 - \lambda_1 - 2\lambda_2 - 3\lambda_3 + \lambda_4 + 2\lambda_5 = 0; \\ \lambda_1 + \lambda_2 + \lambda_3 - \lambda_4 - \lambda_5 = 0; \\ \lambda_1 = 0 \quad \text{или} \quad 1 - w_1 - w_2 - b = 0; \\ \lambda_2 = 0 \quad \text{или} \quad 1 - w_1 - 2w_2 - b = 0; \\ \lambda_3 = 0 \quad \text{или} \quad 1 - 2w_1 - 3w_2 - b = 0; \\ \lambda_4 = 0 \quad \text{или} \quad 1 + 3w_1 + w_2 + b = 0; \\ \lambda_5 = 0 \quad \text{или} \quad 1 + 4w_1 + 2w_2 + b = 0; \end{cases}$$

(мы опустили условия о выполнении ограничений в прямой и двойственной задачах).

Нарисовав выборку на плоскости, можно заметить, что объекты $(1,2)$ и $(4,2)$ не являются опорными, и поэтому соответствующие им двойственные переменные будут равны нулю: $\lambda_2 = \lambda_5 = 0$. Остальные же объекты при оптимальной разделяющей полосе будут опорными и их отступы будут равны 1. Получаем систему

$$\begin{cases} w_1 - \lambda_1 - \lambda_2 - 2\lambda_3 + 3\lambda_4 + 4\lambda_5 = 0; \\ w_2 - \lambda_1 - 2\lambda_2 - 3\lambda_3 + \lambda_4 + 2\lambda_5 = 0; \\ \lambda_1 + \lambda_2 + \lambda_3 - \lambda_4 - \lambda_5 = 0; \\ 1 - w_1 - w_2 - b = 0; \\ \lambda_2 = 0; \\ 1 - 2w_1 - 3w_2 - b = 0; \\ 1 + 3w_1 + w_2 + b = 0; \\ \lambda_5 = 0. \end{cases}$$

Здесь мы опустили все неравенства. Решая систему, получим $b = 1.5$, $w_1 = -1$, $w_2 = 0.5$, $\lambda_1 = 0.375$, $\lambda_2 = 0$, $\lambda_3 = 0.25$, $\lambda_4 = 0.625$, $\lambda_5 = 0$.

■

Задача 1.2. *Рассмотрим задачу с линейно разделимой выборкой. Допустим, мы решили двойственную задачу SVM и нашли вектор двойственных переменных λ. Покажите, что половина ширины разделяющей полосы ρ может быть вычислена по следующей формуле:*

$$\frac{1}{\rho^2} = \sum_{i=1}^{\ell} \lambda_i.$$

Решение. Поскольку выборка линейно разделима, то все объекты, для которых $\lambda_i \neq 0$, окажутся на границе разделяющей полосы. Для них будет выполнено равенство

$$y_i \left(\langle w, x_i \rangle + b \right) = 1,$$

из которого можно выразить b:

$$b = y_i - \langle w, x_i \rangle.$$

Домножим обе стороны на $\lambda_i y_i$ и просуммируем по i (заметим, что для объектов не на границе разделяющей полосы выполняется $\lambda_i y_i = 0$):

$$b \sum_{i=1}^{\ell} \lambda_i y_i = \sum_{i=1}^{\ell} \lambda_i y_i^2 - \sum_{i=1}^{\ell} \lambda_i y_i \langle w, x_i \rangle.$$

Поскольку w, b и λ здесь — решения прямой и двойственной задач, то для них выполнены условия Куна-Таккера. В частности,

$$\sum_{i=1}^{\ell} \lambda_i y_i = 0,$$

$$w = \sum_{i=1}^{\ell} \lambda_i y_i x_i.$$

Заметим также, что $y_i^2 = 1$. Воспользовавшись этими тремя равенствами, получаем:

$$0 = \sum_{i=1}^{\ell} \lambda_i - \|w\|^2.$$

Ранее мы доказали, что в SVM ширина разделяющей полосы равна $\frac{2}{\|w\|}$, поэтому

$$0 = \sum_{i=1}^{\ell} \lambda_i - \frac{1}{\rho^2}.$$

Отсюда получаем требуемое равенство.

■

Задача 1.3. *Пусть $(w, b, \xi_1, \dots, \xi_\ell)$ — оптимальное решение прямой задачи SVM. Предположим, что $\xi_3 > 0$. Выразите отступ объекта x_3 для обученного линейного классификатора через значения $(\xi_1, \dots, \xi_\ell)$.*

Решение. Заметим, что, поскольку $\xi_3 > 0$, то объект x_3 является опорным нарушителем. Отсюда следует, что $\lambda_3 = C$. Напомним, что для двойственной задачи можно записать условия дополняющей нежесткости:

$$\lambda_3[y_3\left(\langle w, x_3\rangle + b\right) - 1 + \xi_3] = 0,$$

откуда можно получить, что $y_3\left(\langle w, x_3\rangle + b\right) - 1 + \xi_3 = 0 \Leftrightarrow M_3 = y_3\left(\langle w, x_3\rangle + b\right) = 1 - \xi_3$.

■

Задача 1.4. *Пусть мы решили двойственную задачу SVM и получили решение $(\lambda_1, \dots, \lambda_\ell)$. Пусть мы также восстановили оптимальный порог b. Выразите сумму $\sum_{i=1}^{\ell} \xi_i$ оптимальных значений параметров $\xi_1, \dots, \xi_\ell$ для прямой задачи.*

Решение. Напомним, что имеет место

$$\mu_i \xi_i = 0 \Leftrightarrow (\mu_i = 0) \text{ или } (\xi_i = 0),$$

поэтому имеет смысл рассматривать лишь те объекты, для которых $\mu_i = 0$. Из $\lambda_i + \mu_i = C$ имеем $\lambda_i = C \neq 0$. Отсюда и из $\lambda_i[y_i\left(\langle w, x_i\rangle + b\right) - 1 + \xi_i] = 0$ имеем

$$y_i\left(\langle w, x_i\rangle + b\right) - 1 + \xi_i = 0 \Leftrightarrow \xi_i = 1 - y_i\left(\langle w, x_i\rangle + b\right) =$$
$$= 1 - y_i\left(\left\langle \sum_{j=1}^{\ell} \lambda_j y_j x_j, x_i \right\rangle + b\right) = 1 - y_i\left(\sum_{j=1}^{\ell} \lambda_j y_j \langle x_i, x_j\rangle + b\right).$$

Если учесть, что для объектов с $\xi_i = 0$ выполняется $\lambda_i = C$, отсюда имеем:

$$\sum_{i=1}^{\ell} \xi_i = \sum_{\substack{i=1\\ \lambda_i = C}}^{\ell} \left(1 - y_i\left(\sum_{j=1}^{\ell} \lambda_j y_j \langle x_i, x_j\rangle + b\right)\right) =$$
$$= \sum_{\substack{i=1\\ \lambda_i = C}}^{\ell} 1 - \sum_{\substack{i=1\\ \lambda_i = C}}^{\ell} \sum_{j=1}^{\ell} y_i y_j \lambda_j \langle x_i, x_j\rangle - b \sum_{\substack{i=1\\ \lambda_i = C}}^{\ell} y_i.$$

■

2 Ядра для объектов сложной структуры

§2.1 All-subsequences kernel

Рассмотрим ядро, часто используемое при работе с текстами. Введём некоторые понятия и обозначения:

- Σ — алфавит, некоторое множество элементов, называемых *символами*;
- Σ^* — множество всех возможных последовательностей (называемых *строками*; включая пустую строку ε) над алфавитом Σ;
- $|s|$— длина строки s;
- $\overline{s_1 s_2 \dots s_k}$ — конкатенация символов или строк $s_1, s_2, \dots, s_k$;
- $s(i)$ — подпоследовательность символов строки s на позициях $i = (i_1, \dots, i_k), 1 \le i_1 < \dots < i_k \le |s|$, т.е. строка $\overline{s_{i_1} \dots s_{i_k}}$;
- $s[a:b] = \begin{cases} s((a, a+1, \dots, b)), a \le b, \\ \varepsilon, a > b. \end{cases}$

Для произвольной строки над алфавитом Σ рассмотрим следующее отображение в спрямляющее пространство:

$$(\varphi(s))_u = |\{i : s(i) = u\}|, u \in \Sigma^*,$$

т.е. $(\varphi(s))_u$ — количество вхождений строки u в строку s в качестве её подпоследовательности. Соответствующее ядро задаётся следующим образом:

$$K(s,t) = \langle \varphi(s), \varphi(t) \rangle = \sum_{u \in \Sigma^*} (\varphi(s))_u (\varphi(t))_u.$$

Тем не менее, вычисление ядра путём формирования признаковых описаний объектов слишком трудозатратно, даже если учитывать лишь подпоследовательности, действительно входящие в строку в исходном пространстве. В частности, для подпоследовательностей длины k количество ненулевых компонент признакового описания строки s в спрямляющем пространстве можно оценить как $\min\left(C_{|s|}^k, |\Sigma|^k\right)$.

Опишем более эффективный способ вычисления ядра. Преобразуем вклад $(\varphi(s))_u$ в значение $K(s,t)$:

$$(\varphi(s))_u (\varphi(t))_u = \sum_{i: s(i) = u} 1 \cdot \sum_{j: t(j) = u} 1 = \sum_{(i,j):\, u = s(i) = t(j)} 1.$$

Тогда

$$K(s,t) = \langle \varphi(s), \varphi(t) \rangle = \sum_{u \in \Sigma^*} \sum_{u = s(i) = t(j)} 1 = \sum_{(i,j):\, s(i) = t(j)} 1.$$

Для эффективного вычисления ядра будем использовать рекуррентную формулу — вычислим значение ядра $K(sa, t)$, где a — символ, дописанный в конец рассматриваемой ранее строки:

$$K(\overline{sa}, t) = \sum_{(i,j):\, \overline{sa}(i)=t(j)} 1.$$

В этом случае для набора i возможны 2 случая: i целиком содержится в s либо последний элемент i является символом a. Таким образом, имеем:

$$\sum_{(i,j):\overline{sa}(i)=t(j)} 1 = \sum_{(i,j):s(i)=t(j)} 1 + \sum_{u:t=\overline{uav}} \sum_{(i,j):\, s(i)=u(j)} 1.$$

При разбиении суммы мы воспользовались тем фактом, что при наличии совпадающих подпоследовательностей в строках $\overline{sa}$ и t с участием символа a в первой из них этот символ должен также встречаться на некоторой позиции в строке t.

Описанные преобразования позволяют нам сформулировать реккурентную формулу для вычисления ядра:

$$K(s, \varepsilon) = 1,$$
$$K(sa, t) = K(s, t) + \sum_{k:\, t_k=a} K(s, t[1:k-1]).$$

Верны также и аналогичные симметричные формулы в силу симметричности ядра.

Таким образом, для вычисления значения ядра можно составить таблицу размера $(|s|+1)(|t|+1)$. Обозначим за $DP(i, j)$ значение в позиции (i, j), $i = \overline{0, |s|}$, $j = \overline{0, |t|}$, этой таблицы и будем заполнять таблицу таким образом, чтобы в позиции (i, j) находилось значение $K(s[1:i], t[1:j])$.

$$\begin{array}{cc|ccccccc}
 & j & 0 & 1 & 2 & \dots & j & \dots & |t| \\
i & & \varepsilon & t_1 & t_2 & \dots & t_j & \dots & t_{|t|} \\
\hline
0 & \varepsilon & 1 & 1 & 1 & \dots & 1 & \dots & 1 \\
1 & s_1 & 1 & \ddots & \ddots & & & & \vdots \\
2 & s_2 & 1 & \ddots & \ddots & & & & \\
\vdots & \vdots & \vdots & & & & \vdots & & \\
i & s_i & 1 & & & & K(s[1:i], t[1:j]) & & \vdots \\
\vdots & \vdots & \vdots & & & & & \ddots & \vdots \\
|s| & s_{|s|} & 1 & \dots & & & \dots & \dots & K(s,t)
\end{array}$$

При этом согласно реккурентной формуле имеем:

$$DP(0, j) = DP(i, 0) = 1, i = \overline{0, |s|}, j = \overline{0, |t|},$$
$$DP(i, j) = DP(i-1, j) + \sum_{k \le j:\, t_k = s_i} DP(i-1, k-1),$$

поэтому таблицу можно заполнять по строкам сверху вниз слева направо. Можно заметить, для вычисления $DP(i, j)$ требуются значения $DP(i-1, k)$, $k = \overline{0, j-1}$, а потому заполнение позиции (i, j) таблицы требует $O(j)$ операций, откуда следует, что вычисление значения ядра $K(s,t) = DP(|s|,|t|)$ требует $O(|s||t|^2)$ операций.

Заметим, что при заполнении i-ой строки таблицы сумма в реккурентной формуле для $DP(i, j)$ использует один и тот же символ s_i, причём каждая последующая сумма включает в себя предыдущие, а потому они могут быть вычислены динамически заранее для i-ой строки путём прохода по строке t, поиска символов s_i и прибавления соответствующего слагаемого суммы в случае успешного нахождения. Обозначив полученный вектор сумм за P, можем вычислять значение в позиции (i, j) таблицы по следующей формуле:

$$DP(i, j) = DP(i-1, j) + P(j).$$

Отметим, для вычисления значений сумм для i-ой строки таблицы требуется $O(|t|)$ операций, а потому полученный алгоритм вычисления ядра $K(s,t)$ имеет сложность $O(|s||t|)$.