Машинное обучение, ФКН ВШЭ
Семинар №14

1 Ядра

Напомним, что *ядром* мы называем некоторую функцию $K(x, z)$, представимую в виде скалярного произведения в некотором пространстве: $K(x, z) = \langle \varphi(x), \varphi(z) \rangle$, где $\varphi : X \to H$ — отображение из исходного признакового пространства в некоторое *спрямляющее пространство*.

Задача 1.1. *Рассмотрим ядро на пространстве всех подмножеств конечного множества D:*

$$K(A_1, A_2) = 2^{|A_1 \cap A_2|}.$$

Покажите, что оно соответствует отображению в $2^{|D|}$-мерное пространство

$$(\varphi(A))_U = \begin{cases} 1, U \subseteq A, \\ 0, \text{иначе}, \end{cases}$$

где U пробегает по всем подмножествам множества D.

Решение. Покажем, что при использовании указанного отображения $\varphi(A)$ скалярное произведение в спрямляющем пространстве действительно имеет указанный вид:

$$\langle \varphi(A_1), \varphi(A_2) \rangle = \sum_{U \subseteq D} (\varphi(A_1))_U (\varphi(A_2))_U .$$

Заметим, что $(\varphi(A_1))_U (\varphi(A_2))_U = 1$ только в том случае, если $(\varphi(A_1))_U = 1$ и $(\varphi(A_2))_U = 1$, т.е. если $U \subseteq A_1$ и $U \subseteq A_2$. Таким образом,

$$\langle \varphi(A_1), \varphi(A_2) \rangle = |\{U \subseteq D \,|\, U \subseteq A_1, U \subseteq A_2\}| .$$

Подсчитаем количество таких множеств. Рассмотрим некоторое $U \subseteq A_1 \cap A_2$. Заметим, что все прочие подмножества D не будут удовлетворять хотя бы одному из условий, в то время как для таким образом выбранного U выполняются оба, поэтому необходимое число — число различных подмножеств $A_1 \cap A_2$. Оно, в свою очередь, равно $2^{|A_1 \cap A_2|}$.

■

Задача 1.2. *Рассмотрим ядро*

$$K(x,z) = \prod_{j=1}^{d}(1 + x_j z_j).$$

Какому спрямляющему пространству оно соответствует?

Решение. Раскроем скобки в выражении для $K(x,z)$. Заметим, что итоговое выражение будет включать мономы всех чётных степеней от 0 до $2d$ включительно. При этом мономы степени $2k$, $k \in \{0,\dots,d\}$, формируются следующим образом: из d скобок, входящих в произведение, случайным образом выбираются k, после чего входящие в них слагаемые вида $x_j z_j$ умножаются на единицы, входящие в состав остальных $d-k$ скобок. Таким образом, в итоговое выражение входят все мономы степени $2k$ над всеми наборами из k различных исходных признаков, и только они. Запишем это формально:

$$K(x,z) = (1 + x_1 z_1)(1 + x_2 z_2)\dots(1 + x_d z_d) = \sum_{k=0}^{d} \sum_{\substack{D \subseteq \{1,\dots,d\} \\ |D|=k}} \prod_{j \in D} x_j z_j.$$

Для простоты понимания приведем вид итогового выражения для $d = 2, 3$ (несложно убедиться в его справедливости путём раскрытия скобок):

$$\begin{aligned} K((x_1,x_2),(z_1,z_2)) &= 1 + x_1 z_1 + x_2 z_2 + x_1 x_2 z_1 z_2, \\ K((x_1,x_2,x_3),(z_1,z_2,z_3)) &= 1 + x_1 z_1 + x_2 z_2 + x_3 z_3 + x_1 x_2 z_1 z_2 + \\ &\quad x_1 x_3 z_1 z_3 + x_2 x_3 z_2 z_3 + x_1 x_2 x_3 z_1 z_2 z_3. \end{aligned}$$

Таким образом, объект x в спрямляющем пространстве представим в следующем виде:

$$\varphi(x) = (1, x_1, \dots, x_d, x_1 x_2, \dots, x_1 x_d, \dots, x_{d-1} x_d, \dots, x_1 x_2 \dots x_d) = \left(\prod_{j \in D} x_j \right)_{D \subseteq \{1,\dots,d\}},$$

то есть в виде вектора мономов всех степеней над наборами различных признаков в исходном пространстве.

■

Задача 1.3. *Пусть $\{(x_i, y_i)\}_{i=1}^{\ell}$, $y_i \in \{-1, +1\}$ — произвольная выборка, а $\varphi(x)$ — отображение в спрямляющее пространство, соответствующее гауссову ядру. Покажите, что в данном спрямляющем пространстве существует линейный классификатор, безошибочно разделяющий выборку $\varphi(x_1), \dots, \varphi(x_\ell)$.*

Решение. Покажем, что вектор весов w в спрямляющем пространстве может быть найден как линейная комбинация объектов выборки $\varphi(x_1), \dots, \varphi(x_\ell)$, т.е. $w = \sum_{i=1}^{\ell} \alpha_i \varphi(x_i)$. Запишем условие верной классификации каждого из объектов выборки в спрямляющем пространстве:

$$\langle w, \varphi(x_i) \rangle = y_i,\ i = \overline{1,\ell}.$$

Заметим, что записанное нами условие является более строгим, чем необходимо, однако в дальнейшем мы покажем существование w, удовлетворяющего этим более строгим ограничениям. Преобразуем:

$$\left\langle \sum_{j=1}^{\ell} \alpha_j \varphi(x_j), \varphi(x_i) \right\rangle = y_i,\ i = \overline{1,\ell},$$

$$\sum_{j=1}^{\ell} \alpha_j \langle \varphi(x_j), \varphi(x_i) \rangle = y_i,\ i = \overline{1,\ell},$$

$$\sum_{j=1}^{\ell} \alpha_j K(x_i, x_j) = y_i,\ i = \overline{1,\ell}.$$

Таким образом, мы получили систему из ℓ линейных уравнений на $\alpha_1, \ldots, \alpha_\ell$, при этом матрицей системы является матрица Грама, являющаяся невырожденной (согласно утв. 1.3 лекции 13), а потому система имеет решение, и соответствующий вектор w существует.

■

2 Ядровой персептрон Розенблатта

Ранее мы изучали линейные модели классификации и узнали, что они отличаются друг от друга тем, какая верхняя оценка пороговой функции потерь используется в задаче оптимизации, т.е. от выбора функции $\tilde{L}(M)$:

$$Q(w, X) = \sum_{i=1}^{\ell} [y_i \langle w, x_i \rangle < 0] = \sum_{i=1}^{\ell} [M_i < 0] \leq \sum_{i=1}^{\ell} \tilde{L}(M_i) \to \min_w .$$

Модель классификации, использующая $\tilde{L}(M) = \max(-M, 0)$, называется *персептроном Розенблатта*. Для данной модели доказана теорема Новикова о сходимости, которая звучит следующим образом:

Теорема 2.1. *Пусть $\mathbb{X}$ — пространство объектов, $\mathbb{Y} = \{-1, +1\}$ — пространство ответов, и выборка $X = \{(x_i, y_i)\}_{i=1}^{\ell}$ линейно разделима. Тогда персептрон Розенблатта, обученный при помощи метода стохастического градиентного спуска, позволяет за конечное число итераций найти вектор весов w^*, безошибочно разделяющий обучающую выборку, для любого начального приближения $w^{(0)}$ и для любого темпа обучения $\eta > 0$, независимо от порядка предъявления объектов обучающей выборки.*

Задача 2.1. *Пусть дана обучающая выборка $X = \{(x_i, y_i)\}_{i=1}^{\ell}$, $y_i \in \{-1, +1\}$. Будем обучать персептрон Розенблатта при помощи стохастического градиентного спуска, при этом начальное приближение вектора весов представимо в виде линейной комбинации объектов обучающей выборки: $w^{(0)} = \sum_{i=1}^{\ell} \alpha_i x_i$.*

1. *Покажите, что вектор весов $w^{(t)}$ после t-ой итерации может быть представлен как линейная комбинация объектов обучающей выборки.*

2. *Переформулируйте алгоритм обучения персептрона таким образом, чтобы он не использовал признаковые описания объектов в явном виде, а лишь скалярные произведения между объектами обучающей выборки.*

3. *Приведите пример ядра, при использовании которого алгоритм персептрона всегда будет настраивать классификатор, не допускающий ошибок на обучающей выборке.*

Решение.

1. Отметим, что

$$\frac{\partial \tilde{L}(M_i)}{\partial w} = \begin{cases} -y_i x_i, M_i < 0, \\ \nexists, M_i = 0, \\ 0, M_i > 0. \end{cases}$$

   Для простоты во втором случае положим градиент равным $-y_i x_i$. Таким образом, имеем $\frac{\partial \tilde{L}(M_i)}{\partial w} = -y_i x_i [y_i \langle w, x_i \rangle \leq 0]$.

   Пусть на t-ой итерации стохастического градиентного спуска для обновления весов был выбран i_t-ый объект выборки, тогда на этой итерации выполняются следующие шаги:

   (а) вычисляется значение $M_i = y_i \langle w^{(t-1)}, x_i \rangle$ для текущего классификатора;

   (б) если $M_i > 0$, т.е. объект классифицируется верно, то обновление весов не производится; в противном случае веса обновляются следующим образом:

$$w^{(t)} = w^{(t-1)} + \eta x_i y_i,$$

   где η — темп обучения.

   Таким образом, на каждой итерации к линейной комбинации объектов обучающей выборки прибавляется один из объектов с некоторым весом, а потому вектор весов на любой итерации можно представить как линейную комбинацию объектов обучающей выборки.

2. В предыдущем пункте мы показали, что алгоритм обучения персептрона состоит из двух чередующихся шагов: вычисления отступа выбранного объекта и обновления весов. Запишем отступ M_i, используя знания из предыдущего пункта:

$$M_i = y_i \langle w, x_i \rangle = y_i \left\langle \sum_{i=j}^{\ell} \alpha_j x_j, x_i \right\rangle = y_i \sum_{j=1}^{\ell} \alpha_j \langle x_i, x_j \rangle.$$

   Таким образом, мы можем определить, допускает ли алгоритм ошибку на объекте x_i, используя лишь скалярные произведения. В случае, если ошибка имеет место, нам необходимо лишь обновить значение α_i:

$$\alpha_i^{(t)} = \alpha_i^{(t-1)} + \eta y_i.$$

3. Рассмотрим гауссово ядро. Как мы показали ранее, выборка в соответствующем спрямляющем пространстве будет линейно разделима. Тогда согласно теореме Новикова за конечное число итераций в процессе обучения персептрона при помощи стохастического градиентного спуска будет получен вектор весов, позволяющий безошибочно разделить обучающую выборку.

■

3 Ядра для объектов сложной структуры

§3.1 All-subsequences kernel

Рассмотрим ядро, часто используемое при работе с текстами. Обозначим за Σ используемый алфавит, за Σ^* — множество всех возможных последовательностей над алфавитом Σ. Для некоторой строки $s \in \Sigma^*$ обозначим за $|s|$ её длину. Для некоторого набора позиций $\mathbf{i} = (i_1, \ldots, i_k), 1 \leq i_1 \leq \ldots i_k \leq |s|$, строки s обозначим за $s(\mathbf{i})$ подпоследовательность символов строки s на позициях $\mathbf{i}$. Кроме того, обозначим за $s[a:b], a, b = \overline{1, |s|}$, подстроку строки s, состоящую из символов на позициях от a до b включительно. В случае $a > b$ положим $s[a:b]$ равной пустой строке ε.

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

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

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 $\min\left(C^k_{|s|}, |\Sigma|^k\right)$.

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

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

Тогда

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

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

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

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

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

При разбиении суммы мы воспользовались тем фактом, что при наличии совпадающих подпоследовательностей в строках 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 $K(s[1:i],t[1:j])$.

	j	0	1	2	…	j	…	$\lvert t\rvert$
i		ε	t_1	t_2	…	t_j	…	$t_{\lvert t\rvert}$
0	ε	1	1	1	…	1	…	1
1	s_1	1	$\ddots$	$\ddots$				$\vdots$
2	s_2	1	$\ddots$	$\ddots$				
$\vdots$	$\vdots$	$\vdots$				$\vdots$		
i	s_i	1				$K(s[1:i],t[1:j])$		$\vdots$
$\vdots$	$\vdots$	$\vdots$					$\ddots$	$\vdots$
$\lvert s\rvert$	$s_{\lvert s\rvert}$	1	…			…	…	$K(s,t)$

При этом согласно реккурентной формуле имеем:

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

поэтому таблицу можно заполнять по строкам сверху вниз слева направо. Можно заметить, для вычисления $DP(i,j)$ требуются значения $DP(i-1,k)$, $k = \overline{0,j}$, 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 $O(|s||t|)$.