# Лекции по методу опорных векторов
*(черновик)*

К. В. Воронцов

9 апреля 2006 г.

## Содержание

**1 Метод опорных векторов (SVM) 2**
1.1 Метод опорных векторов в задачах классификации . . . . . 2
1.1.1 Понятие оптимальной разделяющей гиперплоскости . . . . . 2
1.1.2 Случай линейной разделимости . . . . . 3
1.1.3 Случай отсутствия линейной разделимости . . . . . 5
1.1.4 Ядра и спрямляющие пространства . . . . . 7
1.1.5 Связь SVM и нейронных сетей . . . . . 9
1.1.6 Обобщающая способность SVM . . . . . 10
1.1.7 Эффективный алгоритм построения SVM . . . . . 10
1.2 Метод опорных векторов в задачах регрессии . . . . . 10
1.3 Метод релевантных векторов (RVM) . . . . . 10

# 1 Метод опорных векторов (SVM)

Метод опорных векторов... Метод обобщённого портрета... Вапник [?]... Обобщения на случай, когда линейное разделение невозможно... Причём оказалось, что конструкция алгоритма практически не изменяется... В отличие от нейронных сетей... В отличие от линейного дискриминанта Фишера... В отличие от логистической регрессии...

Напомним основные обозначения. Рассматривается задача обучения по прецедентам $\langle X, Y, y^*, X^\ell \rangle$, где

$X$ — пространство объектов;

$Y$ — множество ответов;

$y^* : X \to Y$ — неизвестная целевая зависимость;

$X^\ell = (x_1, \ldots, x_\ell)$ — обучающая выборка;

$Y^\ell = (y_1, \ldots, y_\ell)$ — вектор ответов на обучающих объектах, $y_i = y^*(x_i)$.

Задача заключается в том, чтобы построить алгоритм $a : X \to Y$, аппроксимирующий целевую зависимость.

## 1.1 Метод опорных векторов в задачах классификации

Рассмотрим задачу классификации на два непересекающихся класса, в которой объекты описываются $n$-мерными вещественными векторами: $X = \mathbb{R}^n$, $Y = \{-1, +1\}$.

Будем строить линейный пороговый классификатор:

$$a(x) = \operatorname{sign}\left(\sum_{j=1}^{n} w_j x^j - w_0\right) = \operatorname{sign}\big(\langle w, x \rangle - w_0\big),$$

где $x = (x^1, \ldots, x^n)$ — признаковое описание объекта $x$; вектор $w = (w^1, \ldots, w^n) \in \mathbb{R}^n$ и скалярный порог $w_0 \in \mathbb{R}$ являются параметрами алгоритма.

Уравнение $\langle w, x \rangle = w_0$ описывает гиперплоскость, разделяющую классы в пространстве $\mathbb{R}^n$.

Хотя правило классификации в точности совпадает с моделью нейрона по МакКаллоку-Питтсу, критерий и методы настройки параметров в SVM радикально отличаются от персептронных (градиентных) методов обучения.

### 1.1.1 Понятие оптимальной разделяющей гиперплоскости

Предположим, что выборка линейно разделима, то есть существуют такие значения параметров $w$, $w_0$, при которых функционал числа ошибок

$$Q(w, w_0) = \sum_{i=1}^{\ell} \big[ y_i(\langle w, x_i \rangle - w_0) > 0 \big]$$

принимает нулевое значение. Но тогда разделяющая гиперплоскость не единственна, поскольку существуют и другие положения разделяющей гиперплоскости, реализующие то же самое разбиение выборки. Идея метода заключается в том, чтобы разумным образом распорядиться этой свободой выбора. Потребуем, чтобы разделяющая гиперплоскость максимально далеко отстояла от ближайших к ней точек

обоих классов. Первоначально данный принцип классификации возник из эвристических соображений: чем дальше классы отстоят от разделяющей поверхности, тем более уверенно можно классифицировать объекты. В дальнейшем принцип максимизации расстояния или *отступа* (margin) объектов от границы классов получил мощное теоретическое обоснование.

**Нормировка.** Заметим, что параметры линейного порогового классификатора определены с точностью до нормировки: алгоритм $a(x)$ не изменится, если $w$ и $w_0$ одновременно умножить на одну и ту же константу. Удобно выбрать эту константу таким образом, чтобы для всех пограничных (т. е. ближайших к разделяющей гиперплоскости) объектов $x_i$ из $X^\ell$ выполнялись условия

$$\langle w, x_i \rangle - w_0 = y_i.$$

Сделать это возможно, поскольку при оптимальном положении разделяющей гиперплоскости все пограничные объекты находятся от неё на одинаковом расстоянии. Остальные объекты находятся дальше. Таким образом, для всех $x_i \in X^\ell$

$$\langle w, x_i \rangle - w_0 \begin{cases} \leqslant -1, & \text{если } y_i = -1; \\ \geqslant 1, & \text{если } y_i = +1; \end{cases} \tag{1.1}$$

Условие $-1 < \langle w, x \rangle - w_0 < 1$ задаёт полосу, разделяющую классы. Ни одна из точек обучающей выборки не может лежать внутри этой полосы. Границами полосы служат две параллельные гиперплоскости с направляющим вектором $w$. Точки, ближайшие к разделяющей гиперплоскости, лежат в точности на границах полосы. При этом сама разделяющая гиперплоскость проходит ровно по середине полосы.

TODO: *Рисунок* ToDo[1]

**Ширина разделяющей полосы.** Чтобы разделяющая гиперплоскость как можно дальше отстояла от точек выборки, ширина полосы должна быть максимальной. Пусть $x_-$ и $x_+$ — две произвольные точки классов $-1$ и $+1$ соответственно, лежащие на границе полосы. Тогда ширина полосы есть

$$\left\langle x_+ - x_-, \frac{w}{\|w\|} \right\rangle = \frac{\big(\langle w, x_+ \rangle - \langle w, x_- \rangle\big)}{\|w\|} = \frac{\big(w_0 + 1 - w_0 - 1\big)}{\|w\|} = \frac{2}{\|w\|}.$$

Итак, в случае, когда выборка линейно разделима, достаточно простые геометрические соображения приводят к следующей задаче: требуется найти такие значения параметров $w$ и $w_0$, при которых норма вектора $w$ максимальна при условии (1.1). Это задача квадратичного программирования. Она будет подробно рассмотрена в следующем разделе. Затем будет сделано обобщение на тот случай, когда линейной разделимости нет.

### 1.1.2 Случай линейной разделимости

Построение оптимальной разделяющей гиперплоскости сводится к минимизации квадратичной формы при $\ell$ ограничениях-неравенствах относительно $n+1$ переменных $w$, $w_0$:

$$\begin{cases} \langle w, w \rangle \to \min; \\ y_i\big(\langle w, x_i \rangle - w_0\big) \geqslant 1, \quad i = 1, \ldots, \ell. \end{cases}$$

По теореме Куна-Таккера эта задача эквивалентна двойственной задаче поиска седловой точки функции Лагранжа:

$$\begin{cases} \mathscr{L}(w, w_0; \lambda) = \dfrac{1}{2}\langle w, w\rangle - \sum\limits_{i=1}^{\ell} \lambda_i \Big( y_i\big(\langle w, x_i\rangle - w_0\big) - 1\Big) \to \min\limits_{w, w_0} \max\limits_{\lambda}; \\ \lambda_i \geqslant 0, \quad i = 1, \ldots, \ell; \\ \lambda_i = 0 \text{ или } y_i\big(\langle w, x_i\rangle - w_0\big) = 1, \quad i = 1, \ldots, \ell; \end{cases}$$

где $\lambda = (\lambda_1, \ldots, \lambda_\ell)$ — вектор двойственных переменных. Последнее из трёх условий называется *условием дополняющей нежёсткости*.

Необходимым условием седловой точки является равенство нулю производных Лагранжиана. Отсюда немедленно вытекают два полезных соотношения:

$$\frac{\partial \mathscr{L}}{\partial w} = w - \sum_{i=1}^{\ell} \lambda_i y_i x_i = 0 \qquad \Longrightarrow \qquad w = \sum_{i=1}^{\ell} \lambda_i y_i x_i; \tag{1.2}$$

$$\frac{\partial \mathscr{L}}{\partial w_0} = -\sum_{i=1}^{\ell} \lambda_i y_i = 0 \qquad \Longrightarrow \qquad \sum_{i=1}^{\ell} \lambda_i y_i = 0; \tag{1.3}$$

Из (1.2) следует, что искомый вектор весов $w$ должен быть линейной комбинацией векторов обучающей выборки, причём только тех, для которых $\lambda_i \neq 0$. Согласно условию дополняющей нежёсткости на этих векторах $x_i$ ограничения-неравенства обращаются в равенства: $\langle w, x_i\rangle - w_0 = y_i$. То есть именно эти векторы находятся на границе разделяющей полосы. Все остальные векторы отстоят дальше от границы классов. Для них $\lambda_i = 0$, они не участвует в сумме (1.2) и фактически оказываются неинформативными. Результат обучения $w$ не изменился бы, если бы их изначально исключили из обучающей выборки.

**Опр. 1.1.** *Если $\lambda_i > 0$ и $\langle w, x_i\rangle - w_0 = y_i$, то объект обучающей выборки $x_i$ называется опорным вектором (support vector).*

Подставляя (1.2) и (1.3) обратно в Лагранжиан, получим эквивалентную задачу квадратичного программирования, содержащую только двойственные переменные:

$$\begin{cases} -\mathscr{L}(\lambda) = -\sum\limits_{i=1}^{\ell} \lambda_i + \dfrac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell} \lambda_i \lambda_j y_i y_j \langle x_i, x_j\rangle \;\to \min\limits_{\lambda}; \\ \lambda_i \geqslant 0, \quad i = 1, \ldots, \ell; \\ \sum\limits_{i=1}^{\ell} \lambda_i y_i = 0. \end{cases} \tag{1.4}$$

Допустим, мы решили эту задачу. Тогда вектор $w$ легко вычисляется по формуле (1.2). Для определения порога $w_0$ достаточно взять произвольный опорный вектор $x_i$ и выразить $w_0$ из условия дополняющей нежёсткости: $w_0 = \langle w, x_i\rangle - y_i$. На практике для повышения численной устойчивости имеет смысл вычислять сумму по всем опорным векторам:

$$w_0 = \sum_{i=1}^{\ell} [\lambda_i > 0]\big(\langle w, x_i\rangle - y_i\big). \tag{1.5}$$

В итоге алгоритм классификации может быть записан в следующем виде:

$$a(x) = \operatorname{sign}\left(\sum_{i=1}^{\ell} \lambda_i y_i \langle x_i, x\rangle - w_0\right). \tag{1.6}$$

Обратим внимание, что реально суммирование идёт не по всей выборке, а только по опорным векторам.

Резюмируя, отметим, что пока остаются открытыми следующие вопросы:

- Как решить двойственную задачу квадратичного программирования (1.4)?
- Общие методы решения таких задач довольно трудоёмки как в смысле реализации, так и по времени выполнения. Возможно ли учесть специфику SVM для построения эффективного алгоритма?
- Что делать, если классы линейно не разделимы?

Начнём с последнего вопроса. В следующем разделе рассматривается несложная модификация двойственной задачи, позволяющая обобщить SVM на случай отсутствия линейной разделимости. После этого будет рассмотрена альтернативная техника — построение расширенного признакового пространства, в котором выборка оказывается линейно разделимой.

### 1.1.3 Случай отсутствия линейной разделимости

Чтобы обобщить SVM на случай линейной неразделимости, позволим алгоритму допускать ошибки на обучающих объектах, но при этом постараемся, чтобы их было поменьше. Введём набор дополнительных переменных $\xi_i \geqslant 0$, характеризующих величину ошибки на объектах $x_i$, $i = 1, \ldots, \ell$ соответственно. Это позволяет смягчить ограничения-неравенства и одновременно ввести в функционал штрафное слагаемое за суммарные ошибки:

$$\begin{cases} \dfrac{1}{2}\langle w, w\rangle + C\sum\limits_{i=1}^{\ell} \xi_i \to \min\limits_{w, w_0, \xi}; \\ y_i\big(\langle w, x_i\rangle - w_0\big) \geqslant 1 - \xi_i, \quad i = 1, \ldots, \ell; \\ \xi_i \geqslant 0, \quad i = 1, \ldots, \ell. \end{cases}$$

Предполагается, что если $\xi_i = 0$, то на объекте $x_i$ ошибки нет. Если $\xi_i > 1$, то на объекте $x_i$ допускается ошибка. Если $0 < \xi_i < 1$, то объект попадает внутрь разделительной полосы, но относится алгоритмом к своему классу.

TODO: *Рисунок* ToDo[2]

Положительная константа $C$ является управляющим параметром метода и позволяет находить компромисс между максимизацией разделяющей полосы и минимизацией суммарной ошибки.

Функция Лагранжа для данной задачи имеет вид:

$$\mathscr{L}(w, w_0, \xi; \lambda, \eta) = \frac{1}{2}\langle w, w\rangle - \sum_{i=1}^{\ell} \lambda_i\Big(y_i\big(\langle w, x_i\rangle - w_0\big) - 1\Big) - \sum_{i=1}^{\ell} \xi_i\big(\lambda_i + \eta_i - C\big),$$

где $\eta = (\eta_1, \ldots, \eta_\ell)$ — вектор переменных, двойственных к переменным $\xi = (\xi_1, \ldots, \xi_\ell)$. Как и в прошлый раз, применим теорему Куна-Таккера, чтобы выписать задачу поиска седловой точки функции Лагранжа:

$$\begin{cases} \mathscr{L}(w, w_0, \xi; \lambda, \eta) \;\to\; \min\limits_{w, w_0, \xi} \max\limits_{\lambda, \eta}; \\ \xi_i \geqslant 0, \quad \lambda_i \geqslant 0, \quad \eta_i \geqslant 0, \quad i = 1, \ldots, \ell; \\ \lambda_i = 0 \text{ или } y_i\big(\langle w, x_i\rangle - w_0\big) = 1 - \xi_i, \quad i = 1, \ldots, \ell; \\ \eta_i = 0 \text{ или } \xi_i = 0, \quad i = 1, \ldots, \ell; \end{cases}$$

В последних двух строках записаны условия дополняющей нежёсткости. Необходимым условием седловой точки является равенство нулю производных Лагранжиана. Отсюда получаются три полезных соотношения:

$$\frac{\partial \mathscr{L}}{\partial w} = w - \sum_{i=1}^{\ell} \lambda_i y_i x_i = 0 \qquad \Longrightarrow \qquad w = \sum_{i=1}^{\ell} \lambda_i y_i x_i; \tag{1.7}$$

$$\frac{\partial \mathscr{L}}{\partial w_0} = -\sum_{i=1}^{\ell} \lambda_i y_i = 0 \qquad \Longrightarrow \qquad \sum_{i=1}^{\ell} \lambda_i y_i = 0; \tag{1.8}$$

$$\frac{\partial \mathscr{L}}{\partial \xi} = -\lambda - \eta + C = 0 \qquad \Longrightarrow \qquad \eta_i + \lambda_i = C, \quad i = 1, \ldots, \ell. \tag{1.9}$$

Первые два соотношения в точности такие же, как и в линейно разделимом случае. Из третьего соотношения и неравенства $\eta_i \geqslant 0$ следует ограничение $\lambda_i \leqslant C$. Отсюда, и из условий дополняющей нежёсткости вытекает, что возможны только три допустимых сочетания значений переменных $\xi_i$, $\lambda_i$, $\eta_i$. Соответственно, выборка делится на объекты трёх типов.

1. $\lambda_i = 0$; $\eta_i = C$; $\xi_i = 0$.

Объект $x_i$ классифицируется правильно и не является опорным. При этом неравенство $y_i\big(\langle w, x_i\rangle - w_0\big) > 1$ означает, что объект находится достаточно далеко от разделяющей гиперплоскости. Такие объекты будем называть *периферийными*.

2. $0 < \lambda_i < C$; $\eta_i > 0$; $\xi_i = 0$.

Объект $x_i$ классифицируется правильно и лежит на границе разделяющей полосы: $y_i\big(\langle w, x_i\rangle - w_0\big) = 1$, Такие объекты, как и раньше, будем называть *опорными*.

3. $\lambda_i = C$; $\eta_i = 0$; $\xi_i > 0$.

Объект $x_i$ переходит через границу разделяющей полосы: $y_i\big(\langle w, x_i\rangle - w_0\big) = 1 - \xi_i < 1$, Если $0 < \xi_i < 1$, то он всё ещё классифицируется правильно, если же $\xi_i > 1$, то алгоритм относит его к чужому классу. Такие объекты будем называть *аутсайдерами*.

В силу соотношения (1.9) в Лагранжиане обнуляются все члены, содержащие переменные $\xi_i$ и $\eta_i$, и он принимает тот же вид, что и в случае линейной разделимости. Параметры разделяющей поверхности $w$ и $w_0$, согласно формулам (1.7) и (1.8), также выражаются только через двойственные переменные $\lambda$. Таким образом, задача снова сводится к квадратичному программированию относительно двойственных переменных $\lambda$. Причём единственное отличие от линейно разделимого случая состоит

в появлении ограничения сверху $\lambda \leqslant C$:

$$\begin{cases} -\mathscr{L}(\lambda) = -\sum\limits_{i=1}^{\ell} \lambda_i + \dfrac{1}{2} \sum\limits_{i=1}^{\ell} \sum\limits_{j=1}^{\ell} \lambda_i \lambda_j y_i y_j \langle x_i, x_j \rangle \;\to\; \min\limits_{\lambda}; \\ 0 \leqslant \lambda_i \leqslant C, \quad i = 1, \ldots, \ell; \\ \sum\limits_{i=1}^{\ell} \lambda_i y_i = 0. \end{cases} \tag{1.10}$$

На практике для построения SVM решают именно эту задачу, а не (1.4), так как гарантировать линейную разделимость выборки в общем случае не представляется возможным.

Для алгоритма классификации сохраняется формула (1.6), с той лишь разницей, что теперь ненулевыми $\lambda_i$ обладают не только опорные объекты, но и объекты-аутсайдеры. В определённом смысле это недостаток SVM, поскольку аутсайдерами часто оказываются шумовые выбросы, и построенное на них решающее правило оказывается неустойчивым.

TODO: *Рекомендации по подбору константы $C$.* ToDo[3]

### 1.1.4 Ядра и спрямляющие пространства

Существует ещё один путь к решению проблемы линейной неразделимости. Это переход от исходного пространства признаковых описаний объектов $X$ к новому пространству $H$ с помощью некоторого преобразования $\psi\colon X \to H$. Если пространство $H$ имеет достаточно высокую размерность, то можно надеяться, что в нём выборка окажется линейно разделимой (легко показать, что если выборка $X^\ell$ не противоречива, то всегда найдётся пространство размерности не более $\ell$, в котором она будет линейно разделима). Пространство $H$ называют *спрямляющим*.

Если предположить, что признаковыми описаниями объектов являются векторы $\psi(x_i)$, а не векторы $x_i$, то построение SVM проводится практически так же, как и ранее. Единственное отличие состоит в том, что скалярное произведение $\langle x, x' \rangle$ в пространстве $X$ всюду заменяется на скалярное произведение $\langle \psi(x), \psi(x') \rangle$ в пространстве $H$. Отсюда вытекает естественное требование: пространство $H$ должно быть наделено скалярным произведением, в частности, подойдёт любое евклидово или гильбертово пространство.

**Опр. 1.2.** *Ядром (kernel function) называется функция $K\colon X \times X \to \mathbb{R}$, представимая в виде $K(x, x') = \langle \psi(x), \psi(x') \rangle$ при некотором $\psi\colon X \to H$, где $H$ — пространство со скалярным произведением.*

Постановка задачи (1.10), и сам алгоритм классификации (1.6) зависят только от скалярных произведений объектов, но не от самих признаковых описаний. Это означает, что при построении SVM скалярное произведение $\langle x, x' \rangle$ можно формально заменить ядром $K(x, x')$. Поскольку ядро в общем случае нелинейно, такая замена приводит к существенному расширению класса допустимых алгоритмов $a : X \to Y$.

Более того, можно вообще не строить спрямляющее пространство $H$ в явном виде, и вместо подбора отображения $\psi$ заниматься непосредственно подбором ядра.

Можно пойти дальше и вовсе отказаться от использования признаковых описаний объектов. Во многих практических задачах объекты изначально задаются информацией об их попарном взаимоотношении, например, об отношении сходства. Если эта информация допускает представление в виде двуместной функции $K(x, x')$, удовлетворяющей аксиомам скалярного произведения, то задача может решаться методом SVM. Для такого подхода недавно был придуман термин *беспризнаковое распознавание* (featureless recognition), хотя многие давно известные метрические алгоритмы классификации (kNN, RBF и др.) также не требуют знания признаковых описаний.

В связи с использованием ядер возникают два вопроса. Любая ли функция двух аргументов $K(x, x')$ может исполнять роль ядра? И как подбирать ядро в каждой конкретной задаче?

Исчерпывающий ответ на первый вопрос даёт теорема Мерсера. Оказывается, функция $K$ должна быть симметричной и неотрицательно определённой, и этого достаточно, чтобы она была ядром. Выходит, что класс ядер достаточно широк.

На второй вопрос универсального ответа до сих пор нет. Существует несколько «стандартных» ядер, которые при ближайшем рассмотрении приводят к построению алгоритмов, эквивалентных по своей структуре полиномиальным разделяющим поверхностям, двухслойным нейронным сетям, методу потенциальных функций (RBF-сетям), и другим известным алгоритмам. Таким образом, ядра претендуют на роль универсального языка для описания широкого класса алгоритмов обучения по прецедентам. Возникает парадоксальная ситуация. С одной стороны, ядра — одно из самых красивых и плодотворных изобретений в машинном обучении. С другой стороны, до сих пор не найдено эффективного общего подхода к их построению.

### Теорема Мерсера.

**Опр. 1.3.** *Функция $K\colon X \times X \to \mathbb{R}$ неотрицательно определена, если для любой выборки $X^p = \{x_1, \ldots, x_p\}$ из $X$ матрица $K = \|K(x_i, x_j)\|$ размера $p \times p$ неотрицательно определена, то есть $a^\mathsf{T} K a \geqslant 0$ для любого $a \in \mathbb{R}^p$.*

**Конструктивные способы построения ядер.** Следующие правила порождения позволяют строить ядра в практических задачах.

1. Произвольное скалярное произведение $K(x, x') = \langle x, x' \rangle$ является ядром.
2. Для любой функции $\psi : X \to \mathbb{R}$ произведение $K(x, x') = \psi(x)\psi(x')$ — ядро.
3. Константа $K(x, x') = 1$ является ядром.
4. Линейная комбинация ядер с неотрицательными коэффициентами $K(x, x') = \alpha_1 K_1(x, x') + \alpha_2 K_2(x, x')$ является ядром.
5. Произведение ядер $K(x, x') = K_1(x, x')K_2(x, x')$ является ядром.
6. Композиция произвольной функции $\varphi : X \to X$ и ядра $K(x, x') = K_0(\varphi(x), \varphi(x'))$ является ядром.
7. TODO: *предел локально-равномерно сходящейся последовательности ядер...* ToDo[4]

8. TODO: *Композиция ядра и произвольной функции... В частности, экспонента от ядра — ядро.* ToDo5

9. TODO: *Если $\rho(x, x')$ — метрика, то ...* ToDo6

**Примеры ядер.**

**Пример 1.1.** Пусть $X = \mathbb{R}^2$. Рассмотри ядро $K(u, v) = \langle u, v\rangle^2$, где $u = (u_1, u_2)$, $v = (v_1, v_2)$, и попробуем понять, какое спрямляющее пространство и преобразование $\psi$ ему соответствуют. Представим квадрат скалярного произведения следующим образом:

$$\langle u, v\rangle^2 = \langle (u_1, u_2), (v_1, v_2)\rangle^2 = (u_1v_1 + u_2v_2)^2 = u_1^2v_1^2 + u_2^2v_2^2 + 2u_1v_1u_2v_2 = \\ = \left\langle (u_1^2, u_2^2, \sqrt{2}u_1u_2), (v_1^2, v_2^2, \sqrt{2}v_1v_2)\right\rangle.$$

Итак, ядро $K$ удалось представить в виде скалярного произведения в пространстве $H = \mathbb{R}^3$. Преобразование $\psi\colon \mathbb{R}^2 \to \mathbb{R}^3$ имеет вид $\psi\colon (u_1, u_2) \mapsto (u_1^2, u_2^2, \sqrt{2}u_1u_2)$. Линейной поверхности в пространстве $H$ соответствует квадратичная поверхность в исходном пространстве $X$. Данное ядро позволяет линейно разделять внутреннюю и наружную часть эллипса, что невозможно в исходном двумерном пространстве.

TODO: *Рисунок* ToDo7

**Пример 1.2.** Усложним ситуацию. Пусть теперь $X = \mathbb{R}^n$,

$$K(u, v) = \langle u, v\rangle^d.$$

Тогда компонентами вектора $\psi(u)$ являются различные произведения $(u_1)^{d_1} \cdots (u_n)^{d_n}$ при всевозможных целых неотрицательных $d_1, \ldots, d_n$, удовлетворяющих условию $d_1 + \cdots + d_n = d$. Число таких мономов, а следовательно и размерность пространства $H$, равно $C^d_{n+d-1}$. Пространство $H$ изоморфно пространству всех полиномов, состоящих из мономов степени $d$ от переменных $u_1, \ldots, u_n$.

**Пример 1.3.** Если $X = \mathbb{R}^n$,

$$K(u, v) = \big(\langle u, v\rangle + 1\big)^d,$$

то $H$ — пространство всех мономов степени *не выше* $d$ от переменных $u_1, \ldots, u_n$. В этом случае пространство $H$ изоморфно пространству всех полиномов степени $d$. Линейная разделимость множеств в этом пространстве эквивалентна полиномиальной разделимости множеств в исходном пространстве $X$.

### 1.1.5 Связь SVM и нейронных сетей

**Преимущества SVM** перед методом стохастического градиента.

- Вместо многоэкстремальной задачи решается задача квадратичного программирования, имеющая единственное решение. Методы оптимизации в этом случае существенно более эффективны.

- Автоматически определяется число нейронов скрытого слоя. Оно равно числу опорных векторов.

- Принцип оптимальной разделяющей гиперплоскости приводит к явной максимизации отступов обучающих объектов, что способствует повышению качества классификации. Градиентные нейросетевые методы выбирают положение разделяющей гиперплоскости произвольным образом, «как придётся».

#### Недостатки SVM

- Метод опорных векторов неустойчив по отношению к шуму в исходных данных. Если обучающая выборка содержат шумовые выбросы, они будут существенным образом учтены при построении разделяющей гиперплоскости. Этого недостатка лишён *метод релевантных векторов* (relevance vector machine, RVM), см. 1.3.

- До сих пор не разработаны общие методы построения спрямляющих пространств или ядер, наиболее подходящих для конкретной задачи. Построение адекватного ядра является искусством и, как правило, опирается на априорные знания о предметной области. На практике вполне «разумные» функции $K(x, x')$, выведенные из содержательных соображений, как правило, оказываются положительно определёнными.

- В общем случае, когда линейная разделимость не гарантируется, приходится подбирать управляющий параметр алгоритма $C$.

### 1.1.6 Обобщающая способность SVM

Минимизация числа опорных векторов.
Максимизация ширины полосы.

### 1.1.7 Эффективный алгоритм построения SVM

Алгоритм настройки SVM методом активных ограничений.

## 1.2 Метод опорных векторов в задачах регрессии

## 1.3 Метод релевантных векторов (RVM)