

# Машинное обучение (Machine Learning)

## Метод опорных векторов (Support Vector Machine - SVM)

Уткин Л.В.

Санкт-Петербургский политехнический университет Петра Великого



# Содержание

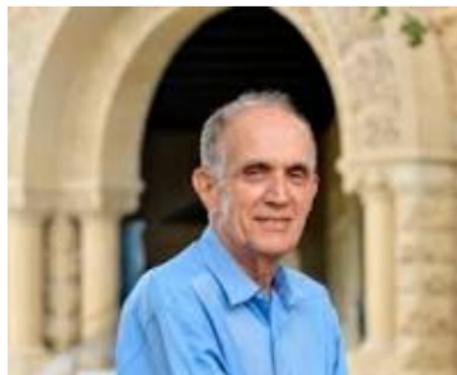
- 1 Геометрическая интерпретация метода
- 2 Прямая и двойственная задачи оптимизации
- 3 Ядра и спрямляющие пространства
- 4 Функционал риска с регуляризацией

*Презентация является компиляцией и заимствованием материалов из замечательных курсов и презентаций по машинному обучению:*

*К.В. Воронцова, А.Г. Дьяконова, Н.Ю. Золотых, С.И. Николенко, Andrew Moore, Lior Rokach, Rong Jin, Luis F. Teixeira, Alexander Statnikov и других.*

# О машине опорных векторов

“New methods always look better than old ones.  
Neural nets are better than logistic regression,  
support vector machines are better than neural nets, etc.”  
- Brad Efron



# Немного истории

- Первые идеи метода были предложены еще в 1950-е годы.
- Метод был создан на основе статистической теории обучения
- Метод стал известен и популярен после замечательной статьи (Вапник и др.) в 1992 г.
- В настоящее время метод успешно используется во многих областях.
- Метод также был модифицирован для задач регрессии.

# Что мы хотим?

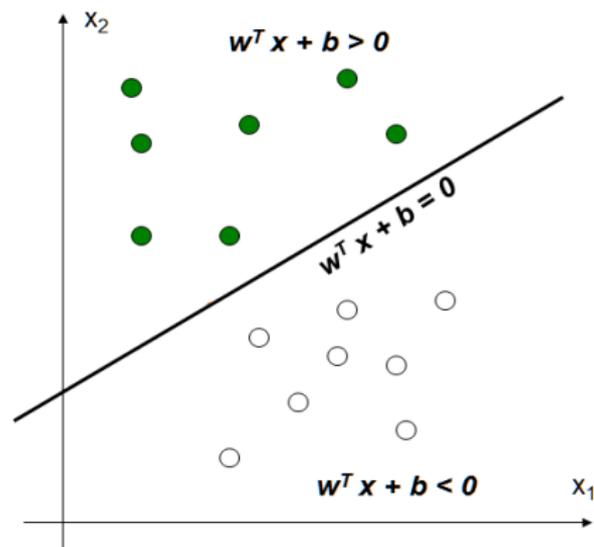
- Метод опорных векторов решает задачу классификации.
- Каждый элемент данных - точка в  $m$ -мерном пространстве  $\mathbb{R}_m$ .
- Формально: есть точки  $x_i$ ,  $i = 1, \dots, m$ , у точек есть метки  $y_i \in \{-1, +1\}$ .

*Можно ли разделить данные гиперплоскостью и какая она?*

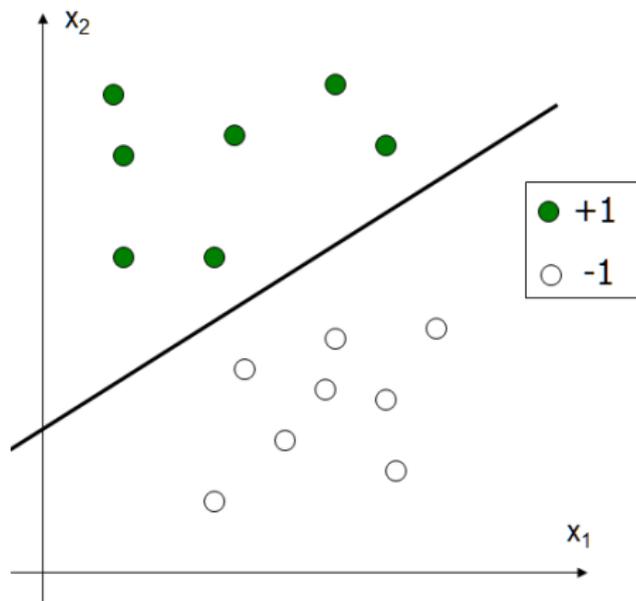
# Классификация данных

$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$  - линейная разделяющая функция  
(гиперплоскость)

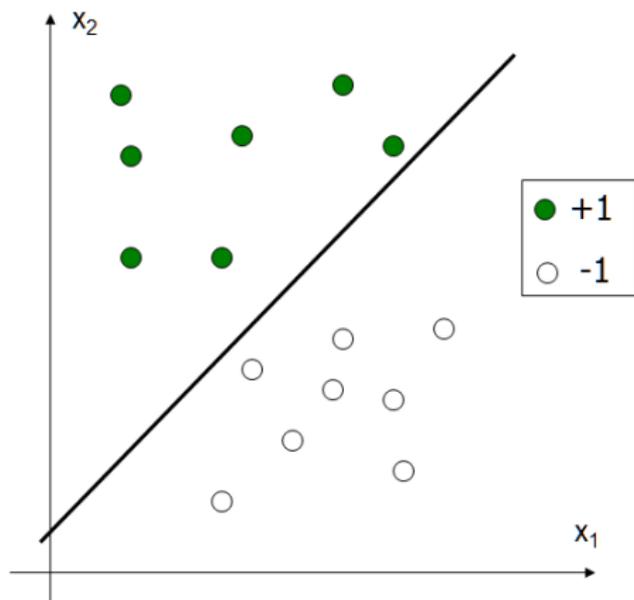
$$g(x_1, \dots, x_m) = \sum_{i=1}^m w_i x_i + b$$



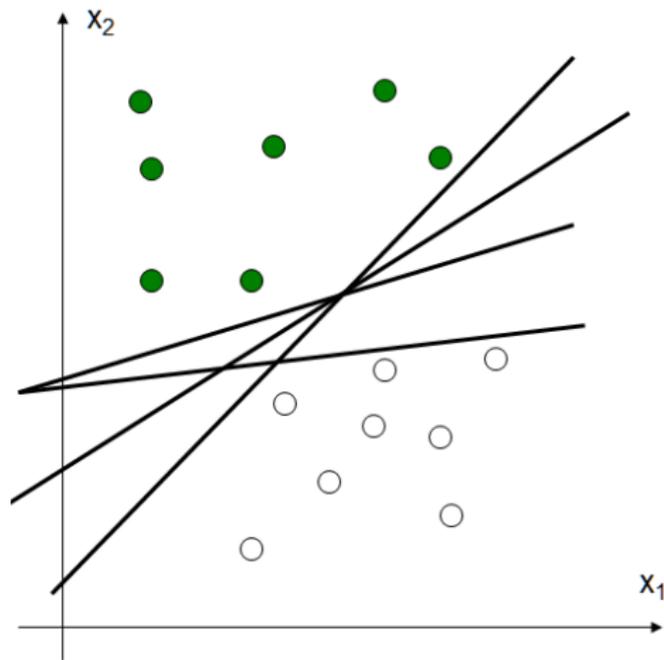
# Классификация данных (один из вариантов)



# Классификация данных (другой вариант)



# Классификация данных (еще много вариантов)



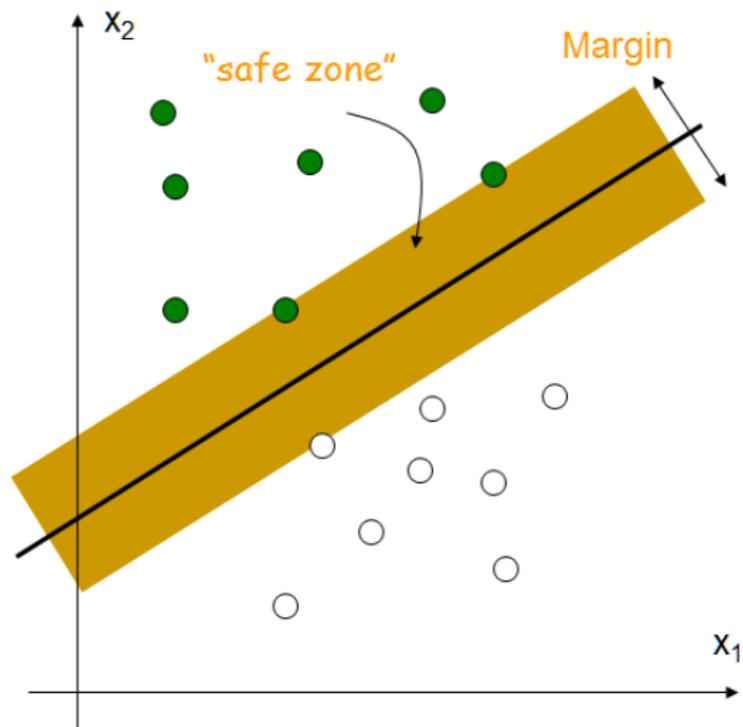
# Какой вариант оптимальный?

Оптимальная разделяющая гиперплоскость — это гиперплоскость, максимизирующая ширину **разделяющей** полосы и лежащая в середине этой полосы.

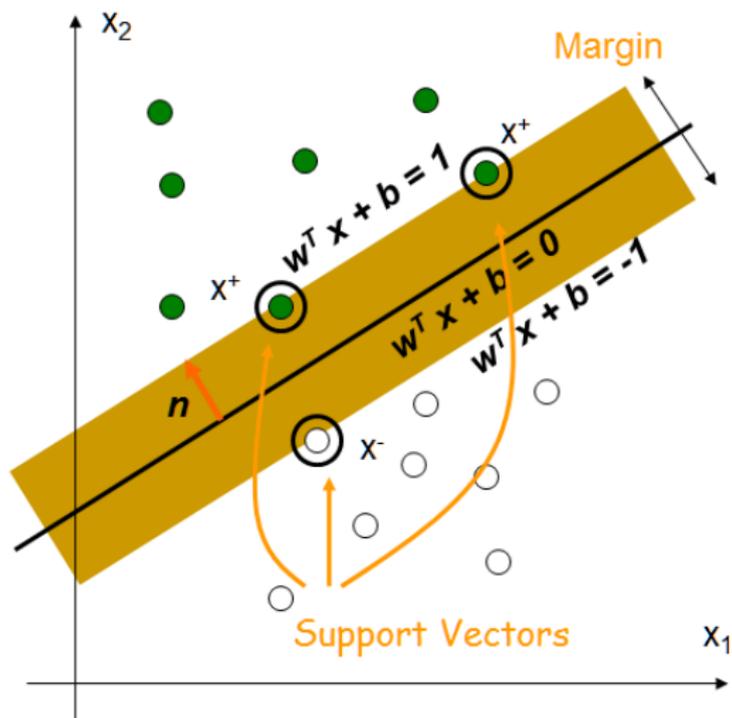
Иными словами, оптимальная разделяющая гиперплоскость максимизирует **зазор** (margin) между плоскостью и данными из обучающей выборки.

Если классы линейно разделимы и каждый содержит не менее одного элемента, то оптимальная разделяющая гиперплоскость единственна.

# Разделяющая полоса



# Какая полоса лучше?



# Как определить ширину полосы?

Без доказательства:

$$M = \frac{2}{\|\mathbf{w}\|} = \frac{2}{w_1^2 + w_2^2 + \dots + w_m^2}$$

Формулировка задачи оптимизации:

$$\frac{2}{\|\mathbf{w}\|} \rightarrow \max_{\mathbf{w}}$$

при условии:

$$\begin{aligned} \text{для } y_i = +1 : \quad & \mathbf{w}^T \mathbf{x} + b \geq 1 \\ \text{для } y_i = -1 : \quad & \mathbf{w}^T \mathbf{x} + b \leq -1 \end{aligned}$$

# Задача оптимизации

$$\frac{\|\mathbf{w}\|}{2} = \frac{1}{2}(w_1^2 + w_2^2 + \dots + w_m^2) \rightarrow \min_{\mathbf{w}}$$

при условии:

$$y_i (\mathbf{w}^T \mathbf{x} + b) \geq 1$$

Это задача квадратической оптимизации с линейными ограничениями!

# Задача оптимизации

Функция Лагранжа:

$$\mathcal{L}(\mathbf{x}, \mathbf{w}, b, \alpha) = \frac{1}{2} \sum_{i=1}^m w_i^2 - \sum_{i=1}^n \alpha_i (y_i (\mathbf{w}^T \mathbf{x} + b) - 1)$$

$\alpha_i, i = 1, \dots, n$  - множители Лагранжа.

Необходимые условия седловой точки функции Лагранжа

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{i=1}^n \alpha_i y_i = 0, \quad \frac{\partial \mathcal{L}}{\partial w_k} = w_k - \sum_{i=1}^n \alpha_i y_i x_i^{(k)} = 0, \quad k = 1, \dots, m$$

# Двойственная задача

Подставляя условия седловой точки в функцию Лагранжа, получаем двойственную задачу

$$\max \left( \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right)$$

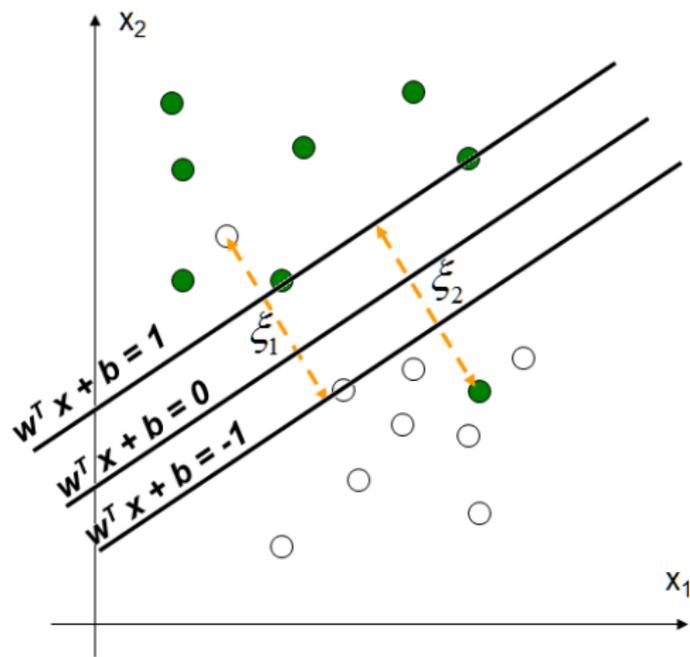
при ограничениях

$$\alpha_i \geq 0, i = 1, \dots, n, \sum_{i=1}^n \alpha_i y_i = 0.$$

Разделяющая функция:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i=1}^n \alpha_i \mathbf{x}_i^T \mathbf{x} + b$$

# Данные линейно не делимы



Вспомогательные переменные  $\xi_i$  (неотрицательные ошибки) могут быть добавлены.

# Переформулирование задачи оптимизации

$$\frac{\|\mathbf{w}\|}{2} = \frac{1}{2}(w_1^2 + w_2^2 + \dots + w_m^2) \rightarrow \min_{\mathbf{w}}$$

при условии:  $y_i (\mathbf{w}^T \mathbf{x} + b) \geq 1, i = 1, \dots, n.$

⇓

$$\frac{\|\mathbf{w}\|}{2} + C \sum_{i=1}^n \xi_i$$

при условии:

$$y_i (\mathbf{w}^T \mathbf{x} + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n.$$

$C$  - штрафной параметр

# Двойственная задача для линейно неразделимых данных

$$\max_{\alpha} \left( \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right)$$

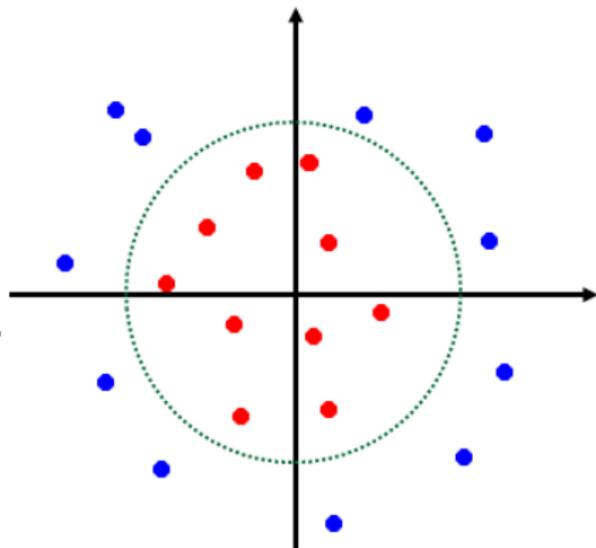
при ограничениях

$$0 \leq \alpha_i \leq C, i = 1, \dots, n, \sum_{i=1}^n \alpha_i y_i = 0.$$

Разделяющая функция:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i=1}^n \alpha_i \mathbf{x}_i^T \mathbf{x} + b$$

# Сложный случай

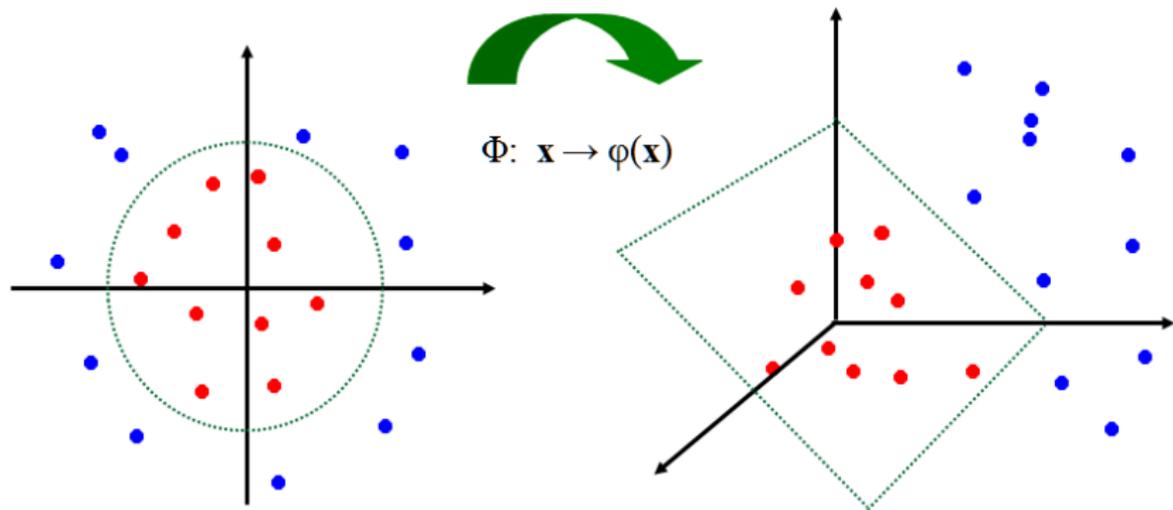


Как построить разделительную плоскость?

# Ядра и спрямляющие пространства

Необходимо перейти от исходного пространства  $X$  в другое большей размерности, называемое **спрямляющим пространством**, с помощью некоторого отображения  $\Phi$ .

# Ядра и спрямляющие пространства



# Спрямяющие пространства (пример)

- Чтобы в двумерном пространстве с координатами  $(x_1, x_2)$  решить задачу классификации квадратичной функцией, надо перейти в пятимерное пространство:

$$(x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2)$$

- Если решить задачу линейного разделения в этом новом пространстве, тем самым решится задача квадратичного разделения в исходном.
- Отображение

$$\varphi : \mathbb{R}^2 \rightarrow \mathbb{R}^5 : \varphi(x_1, x_2) = (x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2)$$

- Вектор в  $\mathbb{R}^5$  теперь соответствует квадратичной кривой общего положения в  $\mathbb{R}^2$ , а функция классификации выглядит как

$$f(\mathbf{x}) = \text{sign}(\varphi(\mathbf{w})\varphi(\mathbf{x}) - b)$$

# Ядра и спрямляющие пространства

Разделяющая плоскость в спрямляющем пространстве:

$$g(\mathbf{x}) = \sum_{i=1}^n \alpha_i \mathbf{x}_i^T \mathbf{x} + b \rightarrow g(\mathbf{x}) = \sum_{i=1}^n \alpha_i \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}) + b$$

Произведение  $\varphi(\mathbf{x}_i)^T \varphi(\mathbf{x})$  играет необыкновенную роль.  
Ядро

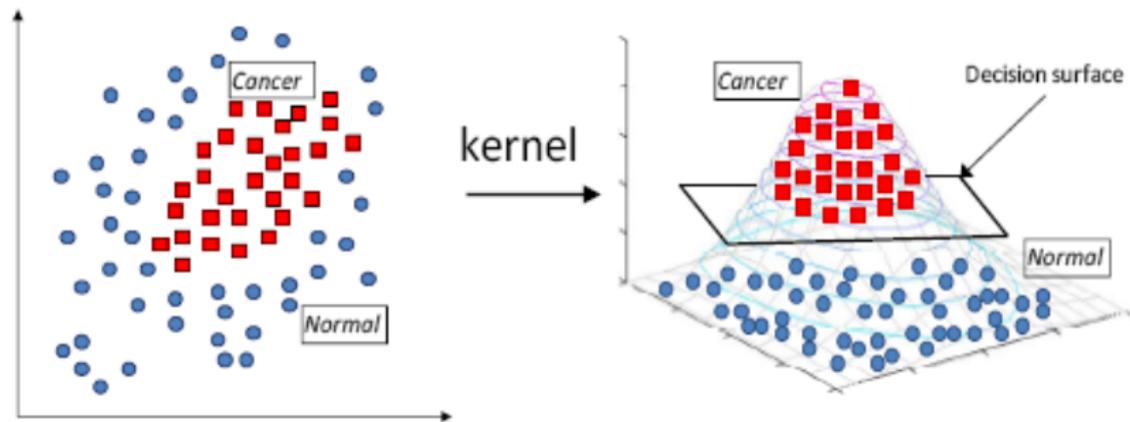
$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \varphi(\mathbf{x}_j).$$

***The Kernel Trick!***

# Kernel Trick

- Данные отображаются в пространство большей размерности для того, чтобы сделать их линейно разделимыми.
- Так как только поточечное произведение  $\mathbf{x}_i^T \mathbf{x}_j$  используется в двойственной форме SVM, то нет необходимости придумывать такое отображение  $\varphi(\mathbf{x}_i)$  в явном виде. Достаточно использовать ядро  $K(\mathbf{x}_i, \mathbf{x}_j)$  из определенного класса ядер.

# Ядра и спрямляющие пространства



# Небольшое видео

(SVMtrick1.mp4)

# Типовые ядра

Примеры наиболее популярных ядер:

1 Линейное:  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$

2 Полиномиальное:  $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$

3 Гауссово (радиальная функция):

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{k=1}^m (x_i^{(k)} - x_j^{(k)})^2}{2\sigma^2}\right)$$

4 Сигмоидальная («нейронная») функция:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

# Двойственная задача для линейно неразделимых данных

$$\max_{\alpha} \left( \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \right)$$

при ограничениях

$$0 \leq \alpha_i \leq C, i = 1, \dots, n, \sum_{i=1}^n \alpha_i y_i = 0.$$

Разделяющая функция:

$$g(\mathbf{x}) = \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}_j) + b$$

# Алгоритм SVM

- 1 Выбрать ядро  $K(\mathbf{x}_i, \mathbf{x}_j)$
- 2 Выбрать значение штрафа  $C$
- 3 Выбрать значение параметра ядра  $(\sigma, \rho, \beta_0, \beta_1)$
- 4 Решить задачу квадратического программирования (есть много программных пакетов) и получить опорные векторы  $\alpha_i$
- 5 Построить разделяющую функцию  $g$ , используя опорные векторы
- 6 Проверить качество классификации, используя скользящий контроль (cross-validation)
- 7 Если качество не удовлетворяет, то переход к шагу 2, иначе завершение алгоритма.

# Некоторые советы по выбору ядра и параметров

- Выбор ядра: Гауссово или полиномиальное
- Выбор параметра Гауссова ядра  $\sigma$ :
  - эвристика:  $\sigma$  выбирается как расстояние между ближайшими точками в разных классах
  - используя скользящий контроль в общем случае

# Эмпирический функционал риска и SVM

- Эмпирический функционал риска:

$$Q(X) = \frac{1}{n} \sum_{i=1}^n L(x_i, y_i).$$

- Функция потерь специального вида (петлевая функция - hinge-loss function):

$$L(x_i, y_i) = \max(0, 1 - y_i g(x_i)).$$

- Задача минимизации функционала риска:

$$\frac{1}{n} \sum_{i=1}^n L(x_i, y_i) = \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i g(x_i, \mathbf{w}, b)) \rightarrow \min_{b, \mathbf{w}}$$

- **Некорректная задача:** имеет бесконечное множество решений, если  $g$  нелинейная (можно провести бесконечно много кривых между точками разных классов)

# Эмпирический функционал риска и слагаемое Тихонова

Регуляризационное (сглаживающее) слагаемое Тихонова:

$$\frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i g(x_i, \mathbf{w}, b)) \rightarrow \min_{b, \mathbf{w}}$$

Заменяем  $\xi_i = \max(0, 1 - y_i g(x_i, \mathbf{w}, b))$  и вводим штраф  $C$



Задача оптимизации:

$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \rightarrow \min_{b, \mathbf{w}}$$

при ограничении

$$1 - y_i g(x_i, \mathbf{w}, b) \geq \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n$$

# Верхняя граница ошибки

Используем метод перекрестного контроля с одним отделяемым элементом (leave-one-out cross-validation, LOO).

$$\text{ошибка LOO} \leq \frac{\text{количество опорных векторов}}{\text{количество обучающих примеров}}$$

# Ключевые особенности SVM

- 1 Максимизирует отступ между положительными и отрицательными объектами
- 2 Штрафует ошибки в случае неразделимой выборки
- 3 Только опорные векторы определяют решение
- 4 Отображение объектов при помощи ядер в новое нелинейное пространство

# Преимущества и недостатки SVM

- Преимущества SVM:
  - Задача выпуклого квадратичного программирования имеет единственное решение.
  - Позволяет рассматривать различные виды нелинейности, изменяя ядра или их параметры.
- Недостатки SVM:
  - Неустойчивость к шуму.
  - Нет общих подходов к оптимизации ядра под задачу.
  - Приходится подбирать параметр  $C$ .
  - Нет отбора признаков.

# Программная реализация в R

- <https://cran.r-project.org/web/views/MachineLearning.html>
- Package 'kernlab': функция: **ksvm**
- Package 'e1071': функция: **svm**
- Package 'wsvm': функция: **wsvm**

см. детальное описание пакетов в А. Karatzoglou, D. Meyer, К. Hornik "Support Vector Machines in R".

# Вопросы

?