



ТЕХНОСФЕРА

Лекция 7 Машина опорных векторов

Николай Анохин

6 ноября 2014 г.

План занятия

SVM

Функции ядра

SGD

Постановка задачи

Пусть дан набор объектов $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$, $\mathbf{x}_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$, $i \in 1, \dots, N$, полученный из неизвестной закономерности $y = f(\mathbf{x})$. Необходимо выбрать из семейства параметрических функций

$$H = \{h(\mathbf{x}, \theta) : \mathcal{X} \times \Theta \rightarrow \mathcal{Y}\}$$

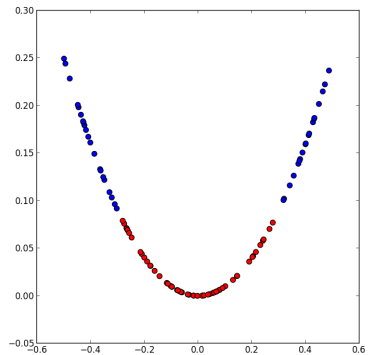
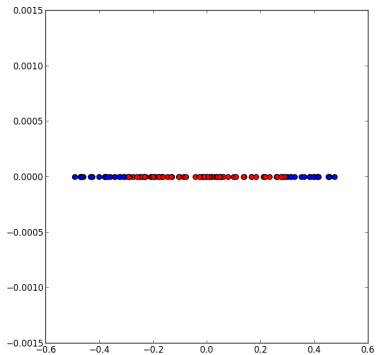
такую $h^*(\mathbf{x}) = h(\mathbf{x}, \theta^*)$, которая наиболее точно аппроксимирует $f(\mathbf{x})$.

Задачи

- ▶ Регрессия: $\mathcal{Y} = [a, b] \subset \mathbb{R}$
- ▶ Классификация: $|\mathcal{Y}| < C$

SVM

Мотивация



Обобщенные линейные модели

Обучающая выборка

$$X = (\mathbf{x}_1, \dots, \mathbf{x}_N)$$

Значения целевой переменной

$$t_1, \dots, t_N; \quad t_j \in \{-1, +1\}$$

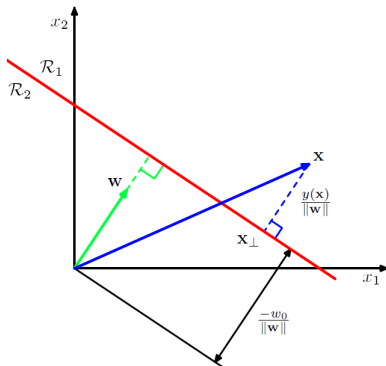
Функция принятия решения

$$y(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b$$

Разделяющая плоскость (РП)

$$y_i(\mathbf{x}_i)t_i > 0$$

Решений много: как выбрать?



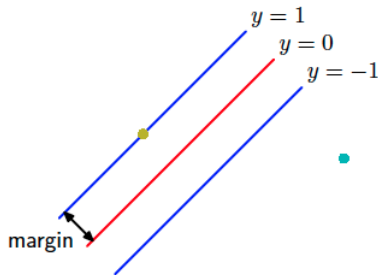
Максимальный зазор

Margin – наименьшее расстояние между РП и обучающим объектом.

$$\begin{aligned}d_j &= \frac{|y(\mathbf{x}_j)|}{\|\mathbf{w}\|} = \frac{t_j y(\mathbf{x}_j)}{\|\mathbf{w}\|} = \\&= \frac{t_j(\mathbf{w}^\top \phi(\mathbf{x}_j) + b)}{\|\mathbf{w}\|}\end{aligned}$$

Оптимальная РП

$$\arg \max_{\mathbf{w}, b} \left[\frac{1}{\|\mathbf{w}\|} \min_j t_j(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) \right]$$



Задача оптимизации

Расстояние от точки x_j до РП

$$d_j = \frac{t_j(\mathbf{w}^\top \phi(\mathbf{x}_j) + b)}{\|\mathbf{w}\|}$$

Для точки x_j , лежащей на минимальном расстоянии от РП положим

$$t_j(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) = 1$$

Задача оптимизации

$$\frac{1}{2} \|\mathbf{w}\|^2 \rightarrow \min_{\mathbf{w}, b}$$

при условиях

$$t_j(\mathbf{w}^\top \phi(\mathbf{x}_j) + b) \geq 1, \quad \forall j \in 1, \dots, N$$

Метод множителей Лагранжа $\mathbf{a} = (a_1, \dots, a_N)^\top$, $a_i \geq 0$.

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{j=1}^N a_j [t_j (\mathbf{w}^\top \phi(\mathbf{x}_j) + b) - 1]$$

Дифференцируем по \mathbf{w} и b

$$\mathbf{w} = \sum_{j=1}^N a_j t_j \phi(\mathbf{x}_j), \quad 0 = \sum_{j=1}^N a_j t_j$$

Подставляем \mathbf{w} и b в лагранжиан

Сопряженная задача

Сопряженная задача

$$\tilde{L}(\mathbf{a}) = \sum_{j=1}^N a_j - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N a_i a_j t_i t_j \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) \rightarrow \max_{\mathbf{a}}$$

при условиях

$$a_j \geq 0, \quad \forall j \in 1, \dots, N$$

$$\sum_{j=1}^N a_j t_j = 0$$

Наблюдения

- ▶ $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$ – неотрицательно-определенная функция
- ▶ лагранжиан $\tilde{L}(\mathbf{a})$ – выпуклая и ограниченная сверху функция

Классификация

Функция принятия решения

$$y(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b = \sum_{j=1}^N a_j t_j \phi(\mathbf{x}_j)^\top \phi(\mathbf{x}) + b = \sum_{j=1}^N a_j t_j k(\mathbf{x}_j, \mathbf{x}) + b$$

Условия Karush-Kuhn-Tucker

$$\begin{aligned} a_j &\geq 0 \\ t_j y(\mathbf{x}_j) - 1 &\geq 0 \\ a_j \{t_j y(\mathbf{x}_j) - 1\} &= 0 \end{aligned}$$

Опорным векторам $\mathbf{x}_j \in S$ соответствуют $a_j > 0$

$$b = \frac{1}{N_s} \sum_{i \in S} \left(t_i - \sum_{j \in S} a_j t_j k(\mathbf{x}_i, \mathbf{x}_j) \right)$$

Линейно-разделимый случай

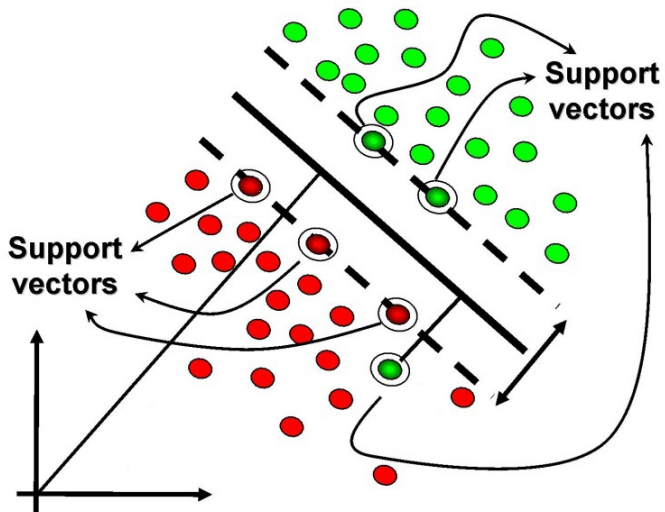
Задача

Дана обучающая выборка

	x_1	x_2	t
\mathbf{x}_1	1	-2	1
\mathbf{x}_2	1	2	-1

Найти оптимальную разделяющую плоскость, используя сопряженную задачу оптимизации

Линейно-неразделимый случай



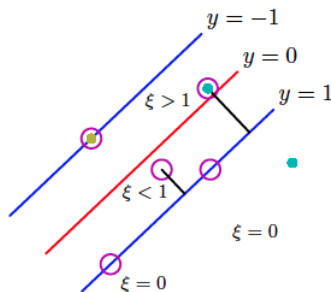
Смягчение ограничений

Переменные $\xi_j \geq 0$ (slacks):

$$\xi_j = \begin{cases} 0, & \text{если } y(\mathbf{x}_j)t_j \geq 1 \\ |t_j - y(\mathbf{x}_j)|, & \text{иначе} \end{cases}$$

Задача оптимизации

$$C \sum_{j=1}^N \xi_j + \frac{1}{2} \|\mathbf{w}\|^2 \rightarrow \min_{\mathbf{w}, b}$$



Сопряженная задача

Сопряженная задача

$$\tilde{L}(\mathbf{a}) = \sum_{j=1}^N a_j - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N a_i a_j t_i t_j \phi(\mathbf{x}_i)^{\top} \phi(\mathbf{x}_j) \rightarrow \max_{\mathbf{a}}$$

при условиях

$$0 \leq a_j \leq C, \quad \forall j \in 1, \dots, N$$

$$\sum_{j=1}^N a_j t_j = 0$$

Наблюдения

- ▶ $a_j = 0$ – правильно проклассифицированные объекты
- ▶ $a_j = C$ – опорные векторы внутри отступа
- ▶ $0 < a_j < C$ – опорные векторы на границе

Классификация

Функция принятия решения

$$y(\mathbf{x}) = \sum_{j=1}^N a_j t_j k(\mathbf{x}_j, \mathbf{x}) + b$$

Константа b

$$b = \frac{1}{N_{\mathcal{M}}} \sum_{i \in \mathcal{M}} \left(t_i - \sum_{j \in S} a_j t_j k(\mathbf{x}_i, \mathbf{x}_j) \right)$$

Задача регрессии

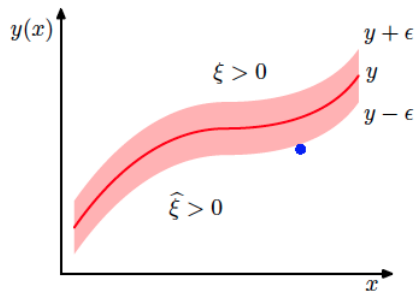
Переменные $\xi_j \geq 0$, $\hat{\xi}_j \geq 0$ (slacks):

$$t_j \leq y(\mathbf{x}_j) + \epsilon + \xi_n$$

$$t_j \geq y(\mathbf{x}_j) - \epsilon - \hat{\xi}_n$$

Задача оптимизации

$$C \sum_{j=1}^N (\hat{\xi}_j + \xi_j) + \frac{1}{2} \|\mathbf{w}\|^2 \rightarrow \min_{\mathbf{w}, b}$$



Численные методы оптимизации

- ▶ Chunking (Vapnik, 1982)
- ▶ Decomposition (Osuna, 1996)
- ▶ Sequential Minimal Optimization (Platt, 1999)

Функции ядра

Функции ядра

$\phi(\mathbf{x})$ – функция преобразования \mathbf{x} из исходного пространства в спрямляющее пространство

Проблема: количество признаков может быть очень велико

Идея Kernel Trick

В процессе тренировки и применения SVM исходные векторы \mathbf{x} используются только как аргументы в скалярном произведении $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$. Но в этом случае можно избежать вычисления $\phi(\mathbf{x})$!

Теорема Мерсера

Теорема

Функция $k(\mathbf{x}, \mathbf{z})$ является ядром тогда и только тогда, когда она

- ▶ симметрична

$$k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x})$$

- ▶ неотрицательно определена

$$\int_{\mathbf{x} \in \mathbf{X}} \int_{\mathbf{z} \in \mathbf{X}} k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0, \quad \forall g(\mathbf{x}) : \mathbf{X} \rightarrow \mathbb{R}$$

Задача

Пусть $\mathbf{x} \in \mathbb{R}^2$, а преобразование $\phi(\mathbf{x})$

$$\phi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1x_2, x_2^2).$$

Проверить, что функция $k(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x}^\top \mathbf{z})^2$ является функцией ядра для данного преобразования.

Некоторые стандартные функции ядра

- ▶ Линейное ядро

$$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{z}$$

- ▶ Полиномиальное ядро степени d

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\top \mathbf{z} + r)^d$$

- ▶ Radial Basis Function

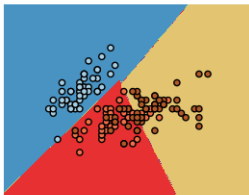
$$k(\mathbf{x}, \mathbf{z}) = e^{-\gamma \|\mathbf{x} - \mathbf{z}\|^2}$$

- ▶ Sigmoid

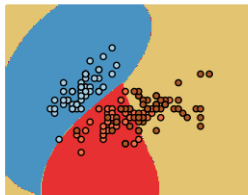
$$k(\mathbf{x}, \mathbf{z}) = \tanh(\gamma \mathbf{x}^\top \mathbf{z} + r)$$

Опять ирисы

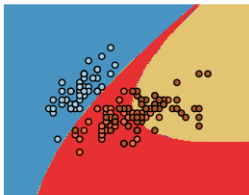
SVC with linear kernel



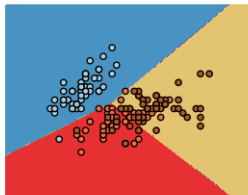
SVC with RBF kernel



SVC with polynomial (degree 3) kernel



LinearSVC (linear kernel)



SGD

Связь с линейными моделями

Задача оптимизации

$$\sum_{j=1}^N \xi_j + \frac{1}{2} \|w\|^2 \sim \sum_{j=1}^N E(y(\mathbf{x}_j), t_j) + \lambda \|w\|^2 \rightarrow \min_{w, b}$$

Hinge loss

$$E(y_j, t_j) = \begin{cases} 1 - y_j t_j, & \text{если } y_j t_j < 1 \\ 0, & \text{иначе} \end{cases}$$

Stochastic Gradient Descent

Градиентный спуск

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta(k) \frac{1}{N} \sum_{n=1}^N \nabla_{\mathbf{w}} l(\mathbf{x}_n, \mathbf{w}, t_n)$$

Стохастический градиентный спуск

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta(k) \nabla_{\mathbf{w}} l(\mathbf{x}_k, \mathbf{w}, t_k)$$

Усредненный стохастический градиентный спуск $\bar{\mathbf{w}}_k = \frac{1}{k} \sum_{j=1}^k \mathbf{w}_j$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta(k) \nabla_{\mathbf{w}} l(\mathbf{x}_k, \mathbf{w}, t_k), \quad \bar{\mathbf{w}}_{k+1} = \frac{k}{k+1} \bar{\mathbf{w}}_k + \frac{1}{k+1} \mathbf{w}_{k+1}$$

Сходимость: $\sum_k \eta_k^2 < \infty$, $\sum_k \eta_k = \infty$

SGD tips

- ▶ Использовать SGD, когда обучение модели занимает слишком много времени
- ▶ Перемешать тренировочную выборку
- ▶ Следить за training error и **validation error**
- ▶ Проверять, правильно ли вычисляется градиент

$$Q(z, w + \delta) \approx Q(z, w) + \delta g$$

- ▶ Подобрать η_0 на небольшой выборке

$$\eta_k = \eta_0(1 + \eta_0 \lambda k)^{-1}, \quad \lambda - \text{параметр регуляризации}$$

SVM – итоги

- + Нелинейная разделяющая поверхность
- + Глобальная оптимизация
- + Разреженное решение
- + Хорошая обобщающая способность
 - Не поддерживает $p(C_k|\mathbf{x})$
 - Чувствительность к выбросам
 - Нет алгоритма выбора ядра
 - Медленное обучение

Эксперименты над SVM

В задаче рассматриваются 4 алгоритма генерации обучающих выборок. Требуется разработать функцию, автоматически подбирающую параметры модели SVM в зависимости от полученной выборки.

Инструкция по выполнению задания

1. Скачать код <http://bit.ly/1y26EF4>
2. Запустить хелп, при желании изучить

```
$ python svm.py -h  
$ python svm.py cc -h
```

3. Запустить какой-нибудь пример, разобраться с картинкой

```
$ python svm.py gauss
```

4. Работать над кодом (функция `select_model`)

Вопросы

