Машинное обучение

Косатый Дмитрий

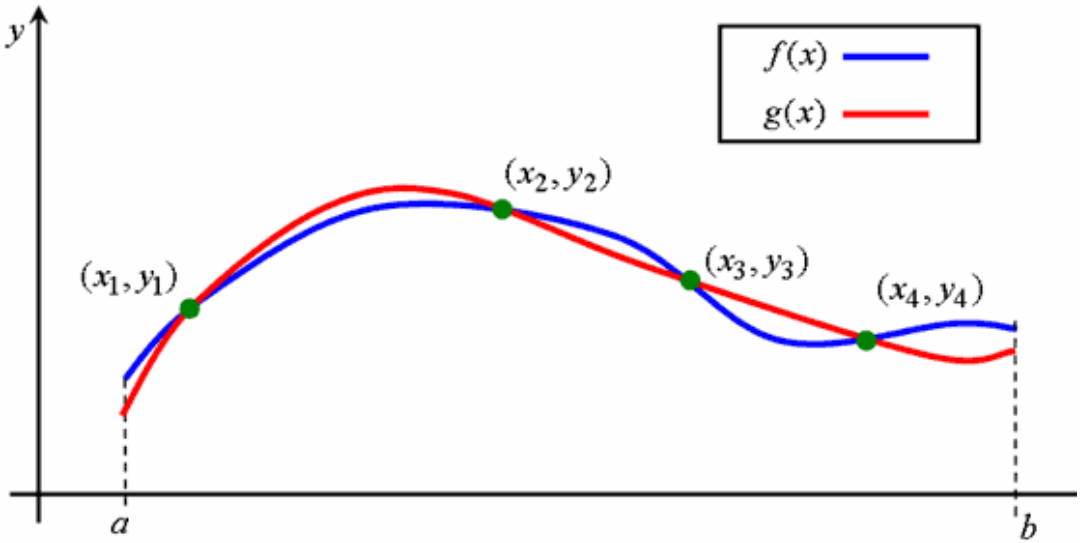
Математико-механический факультет СПбГУ

21 ноября 2015 г.

Содержание

Самые общие слова

- Задача машинного обучения сводится к задаче аппроксимации неизвестной функции
- Задача аппроксимации - это задача сопоставления одних объектов другими, более удобными объектами



Выдать клиенту кредит, или нет?

Формализация:

- Вход: x (Информация о клиенте)

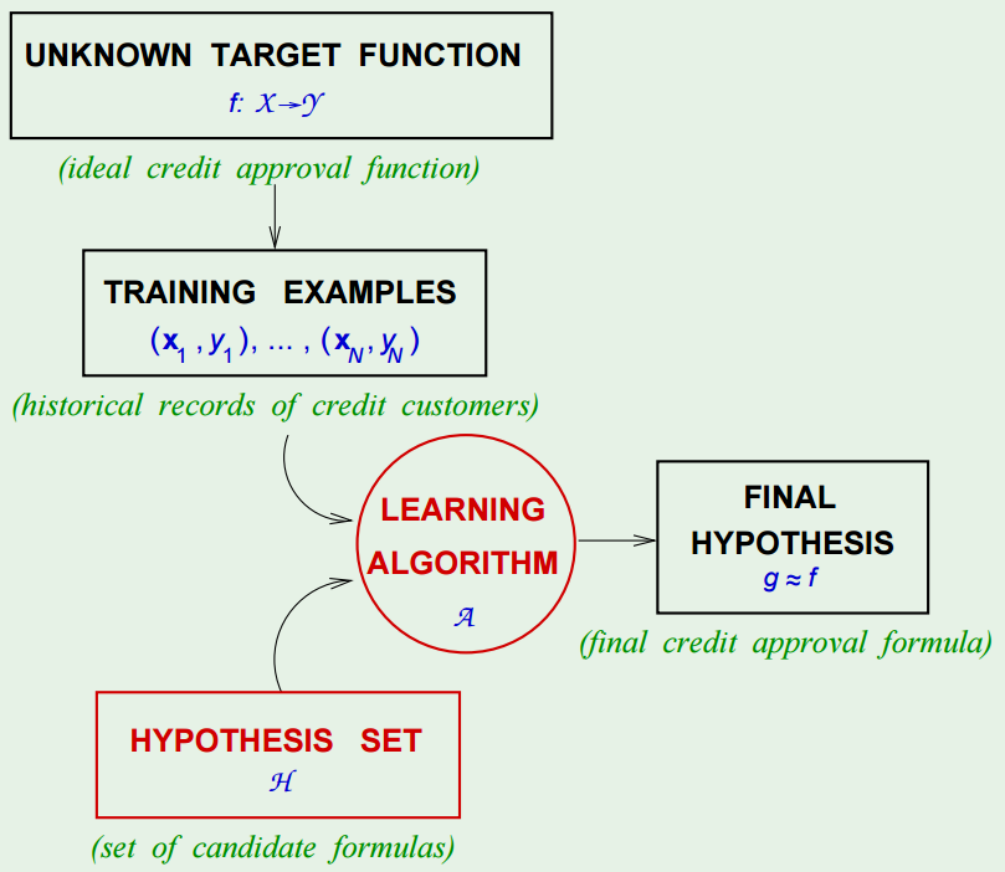
Возраст	23 года
Пол	мужской
Годовая з/п	\$30,000
Стаж работы	1 год
Задолженность	\$15,000
...	...

- Выход: y (Хороший/плохой клиент?)
- Целевая функция: $f : x \to y$ (Идеальная формула одобрения кредита)
- Данные: $(x_1, y_1), \ldots, (x_N, y_N)$ (Архивные записи)

$\Downarrow$

- Гипотеза: $g : X \to Y$ (Формула для использования)

Диаграмма машинного обучения



Линейная классификация

Вернёмся к примеру кредитного скоринга:

- $x = x\,(x_1, \ldots, x_d)$ – входной вектор тех самых признаков, которые характеризуют клиента
  y – решение о выдачи/не выдачи кредита
  w – неизвестный вектор весов, который мы определяем в процессе обучения
- Модель линейной классификации:

$$h(x) = sign\left(\sum_{i=1}^{d} w_i x_i - threshold\right) = sign(w^T x)$$

Линейная регрессия

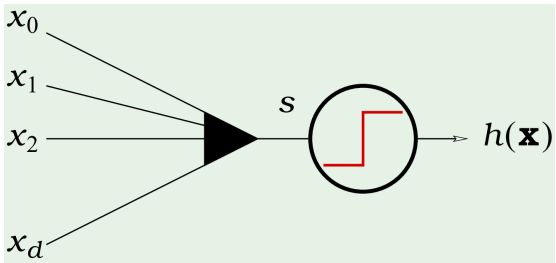
В терминах того же примера

- $x = x\,(x_1, \ldots, x_d)$ – входной вектор тех самых признаков, которые характеризуют клиента
  y – выходной размер кредита
  w – неизвестный вектор весов, который мы определяем в процессе обучения
- Модель линейной регрессии: $h(x) = \sum_{i=0}^{d} w_i x_i = w^T x$

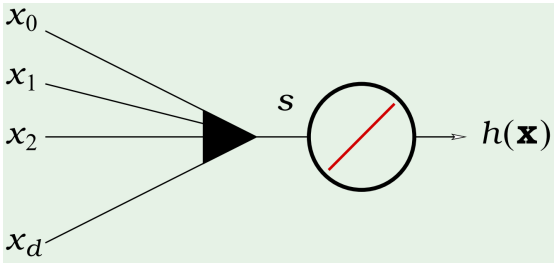
Логистическая регрессия

$$s = \sum_{i=0}^{d} w_i x_i = w^T x$$

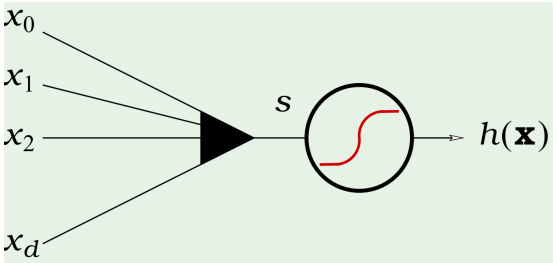
Линейная классификация
$h(x) = sign(s)$



Линейная регрессия
$h(x) = s$

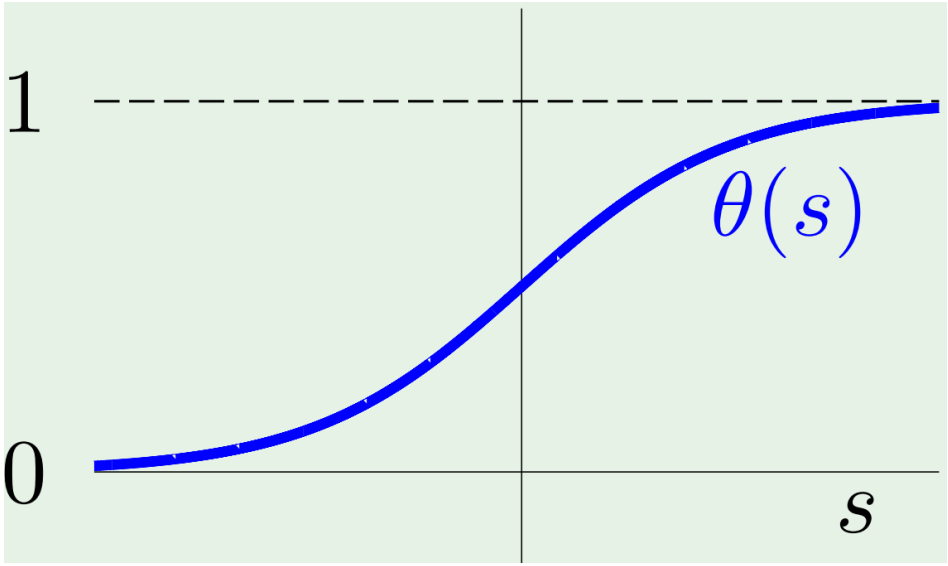


Логистическая регрессия
$h(x) = \theta(s)$



$\theta(s) = \frac{1}{1+e^{-s}}$ - логистическая функция

Логистическая регрессия



$$P(y|x) = \begin{cases} h(x), y = +1 \\ 1 - h(x), y = -1 \end{cases}$$

Метод наименьших квадратов

Рассмотрим МНК для линейной регрессии:

- Хотим, чтобы $h(x)$ как можно лучше аппроксимировала $f(x)$
- В нашем распоряжении тренировочная выборка $(x_1, y_1), \ldots, (x_N, y_N)$
- Тогда функцию ошибки $E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} (h(x_n) - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 = \frac{1}{N} ||Xw - y||^2$ сводим к минимому методом наименьших квадратов
- $\nabla_w (E_{in}(h)) = \nabla_w \left(\frac{1}{N} ||Xw - y||^2\right) = \frac{2}{N} X^T (Xw - y) = 0$, $X^T X w = X^T y$
- В итоге, $w = (X^T X)^{-1} X^T y$

Метод градиентного спуска

Рассмотрим метод градиентного спуска для логистической регрессии:

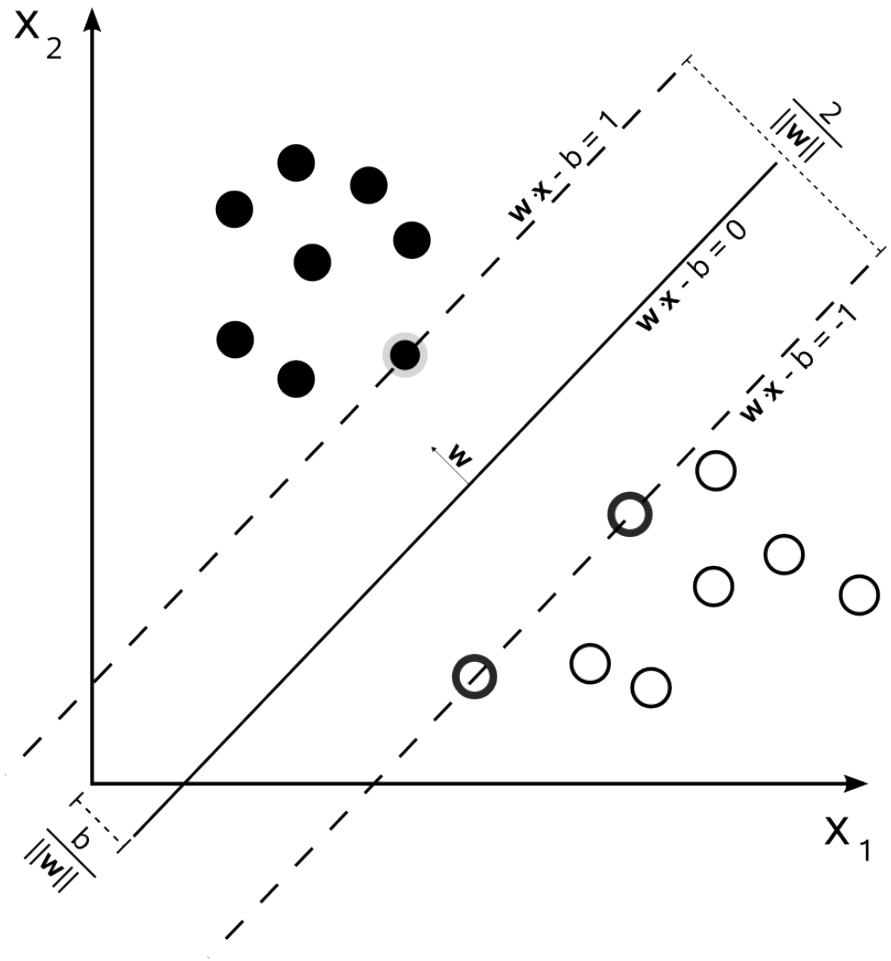
- $w_{t+1} = w_t - \eta \nabla E_{in}(w_t))$, $E_{in} \to min$
  - w – это вектор весов; начальное значение можно брать произвольным, например, $w_0 = (0, \ldots, 0)$
  - η – это шаг; чем меньше, тем дольше обучение
  - $E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} \ln\left(1 + e^{-y_n w^T x_n}\right)$
  - $\nabla E_{in} = -\frac{1}{N} \sum_{n=1}^{N} \frac{y_n x_n}{1+e^{y_n w^T(t) x_n}}$

Случай линейно разделимых классов

- Имеем два класса данных
- SVM ищет разделяющую поверхность, максимально удалённую от любых точек данных
- Классификатор с большим зазором снижает неопределённость решения

Случай линейно разделимых классов

- Имеем обучающее множество $(\vec{x}_i, y_i)$, где $\vec{x}_i$ - p-мерный вектор, $y_i \in [-1, 1]$ - метка класса
- Уравнение разделяющей гиперплоскости: $\vec{w}^T \vec{x} = b$, где b - это параметр сдвига, $\vec{w}^T$ - это вектор весов



Случай линейно разделимых классов

Задача

$$\begin{cases} \dfrac{2}{||\vec{w}||} \to max, \dfrac{1}{2}||\vec{w}|| \to min \\ y_i \left(\vec{w}^T \vec{x}_i - b \right) \geq 1 \end{cases}$$

Ответ

$$\begin{cases} \vec{w} = \sum\limits_i \lambda_i y_i \vec{x}_i \\ b = \vec{w}^T \vec{x}_i - y_i \end{cases}$$

$$f(\vec{x}) = sign \left(\sum_i \lambda_i y_i \vec{x}_i^T \vec{x} - b \right)$$

Решение

По теореме Куна-Таккера эта задача эквивалентна поиску седловой точки функции Лагранжа

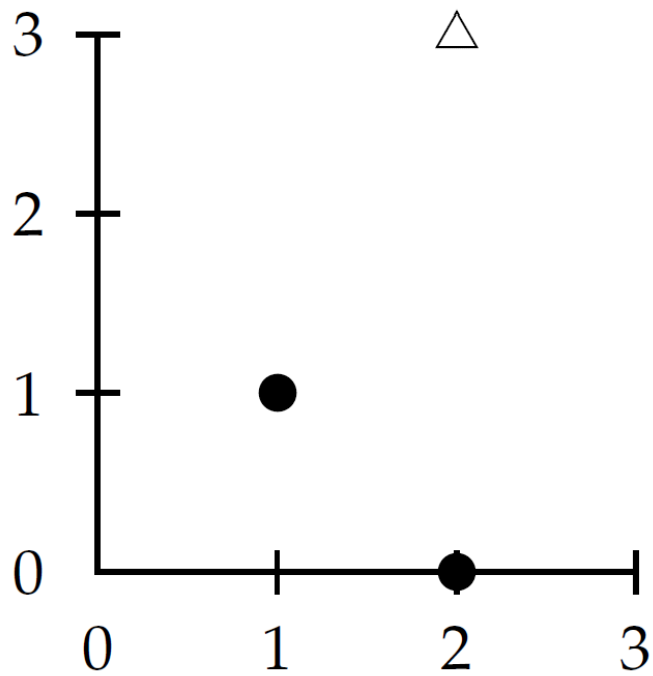
$$\begin{cases} \mathcal{L}(\vec{w}, b; \lambda) = \dfrac{1}{2}||\vec{w}||^2 - \sum\limits_i \lambda_i \left(y_i(\vec{w}^T \vec{x}_i) - b) - 1 \right) \to \min\limits_{\vec{w}, b} \max\limits_{\lambda} \\ \lambda_i \geq 1 \end{cases}$$

Случай линейно разделимых классов, пример

- $\vec{w} = (a, 2a)\ \forall a$
- $sign\left(y_i\left(\vec{w}^T\vec{x}_i + b\right)\right) = 1,$
  $i = 1, 2$

$$\begin{cases} (a, 2a)\begin{pmatrix}1\\1\end{pmatrix} + b = 1 \\ (a, 2a)\begin{pmatrix}2\\3\end{pmatrix} + b = -1 \end{cases}$$

$$\begin{cases} a = 2/5, \vec{w} = (2/5, 4/5) \\ b = -11/5 \end{cases}$$

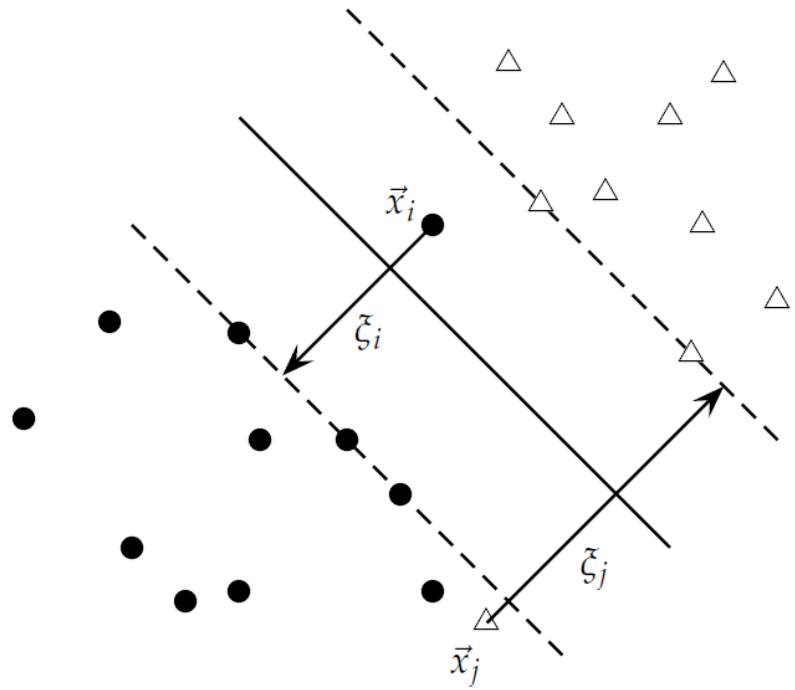


- Зазор $\frac{2}{||w||} = \frac{2}{\sqrt{\frac{4}{25}+\frac{16}{25}}} = \sqrt{5}$

SVM с мягким зазором

$$\begin{cases} \frac{1}{2}||\vec{w}||^2 + C\sum_i \xi_i \to \min\limits_{\vec{w},b,\xi_i} \\ y_i\left(\vec{w}^T\vec{x}_i - b\right) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases}$$

$$\begin{cases} \vec{w} = \sum_i \lambda_i y_i \vec{x}_i \\ b = \vec{w}^T\vec{x}_k - y_k(1-\xi_k), \\ k = argmax_k \lambda_k \end{cases}$$



SVM с несколькими классами

- Создаётся множество классификаторов, работающих по принципу «Один против всех»:
  Выбирается класс, на котором документ **максимально** удалён от разделяющей поверхности

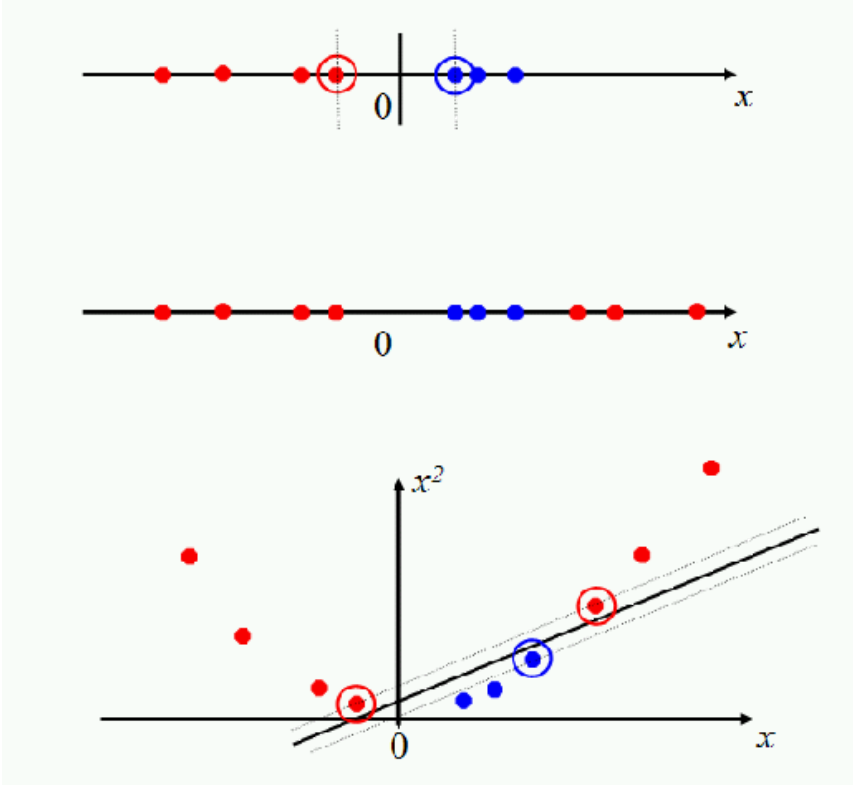
- Создаётся множество классификаторов, работающих по принципу «Один против одного»:
  Выбирается класс, предложенный **большинством** классификаторов

Честно построенный многоклассовый классификатор

- Строим бинарный классификатор по вектору признаков $\Phi(\vec{x}, y)$
- $y = argmax_{y'} \Phi(\vec{x}, y')$
- $\forall i \; \forall y \neq y_i \;\; \vec{w}^T \Phi(\vec{x}_i, y_i) - \vec{w}^T \Phi(\vec{x}_i, y) \geq 1 - \xi_i$

Нелинейный SVM

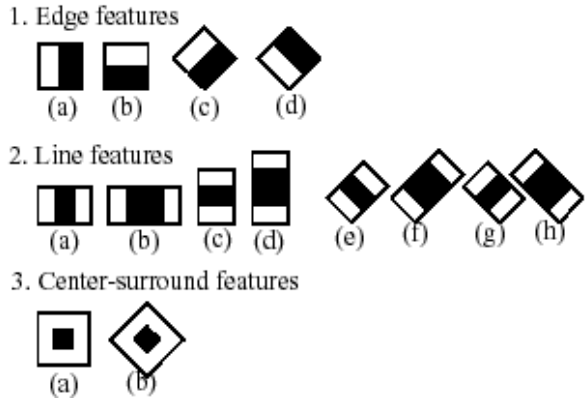
- Отображаем данные в пространство более высокой размерности, где примененяем линейный классификатор
- $f(\vec{x}) = sign\left(\sum_i \lambda_i y_i K(\vec{x}_i \vec{x}) - b\right)$
- Полиномиальное ядро $K(\vec{x}, \vec{\zeta}) = \left(1 + \vec{x}^T \vec{\zeta}\right)^d$
- Радиальная базисная функция Гаусса $K(\vec{x}, \vec{\zeta}) = e^{\frac{(\vec{x} - \vec{\zeta})^2}{2\sigma^2}}$



Признаки Хаара

Схожи с вейвлетами Хаара

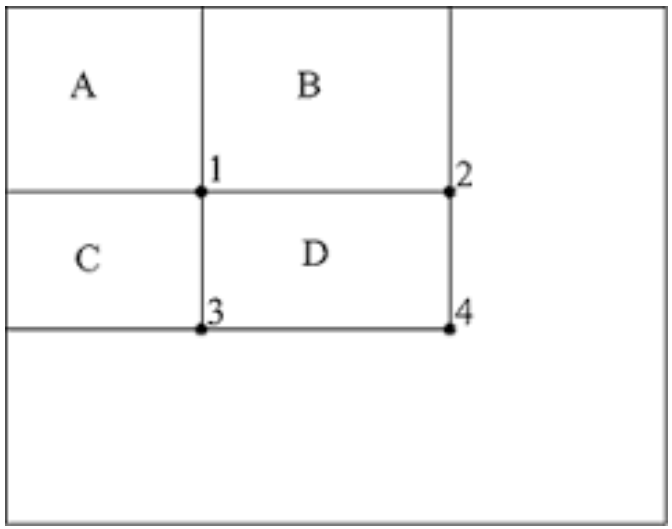
$$\psi(t) = \begin{cases} 1, 0 \leq t < \frac{1}{2}, \\ -1, \frac{1}{2} \leq t \leq 1, \\ 0, \text{ иначе} \end{cases}$$



Интегральное представление изображения

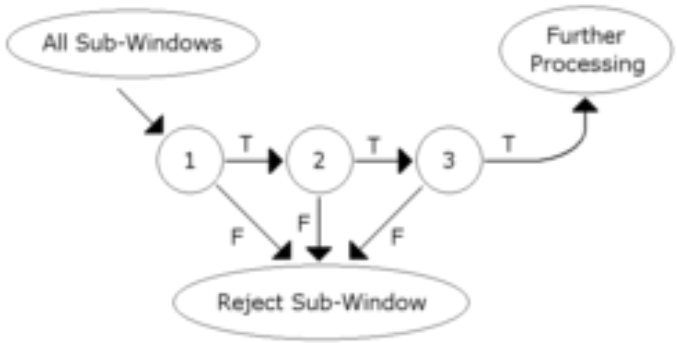
- 1 – это сумма пикселей в прямоугольнике A
- 2 – $A + B$
- 3 – $A + C$
- 4 – $A + B + C + D$
- Тогда $4 + 1 - (2 + 3)$ – это сумма пикселей в прямоугольнике D
- $ii(D) = ii(ABCD) + ii(A) - (ii(B) + ii(C))$

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$



Метод Виолы-Джонса, 2001 г.

- Первый детектор лиц, работающий в режиме реального времени
- Работает по принципу “сканирующего окна”, когда окно фиксированного размера проходит с неким шагом по входному изображению, представленном в интегральном виде, и вычисляет признаки Хаара
- Признаки Хаара собираются в каскадный классификатор



AdaBoost (Adaptive Boosting)

В процессе обучения новый каскад конструируется по (**адаптируется к**) объектам, неправильно классифицированным предыдущими каскадами
По множеству слабых классификаторов строит один сильный классификатор

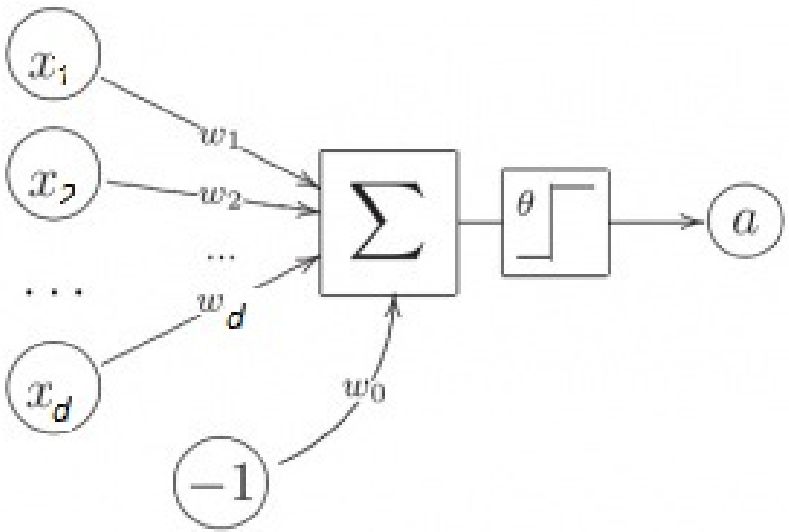
- Слабый классификатор – кл-р, кот. допускает много ошибок (ненамного лучше случайного угадывания)
- Сильный классификатор – кл-р, кот. допускает мало ошибок, и пригоден для классификации

Алгоритм AdaBoost'a

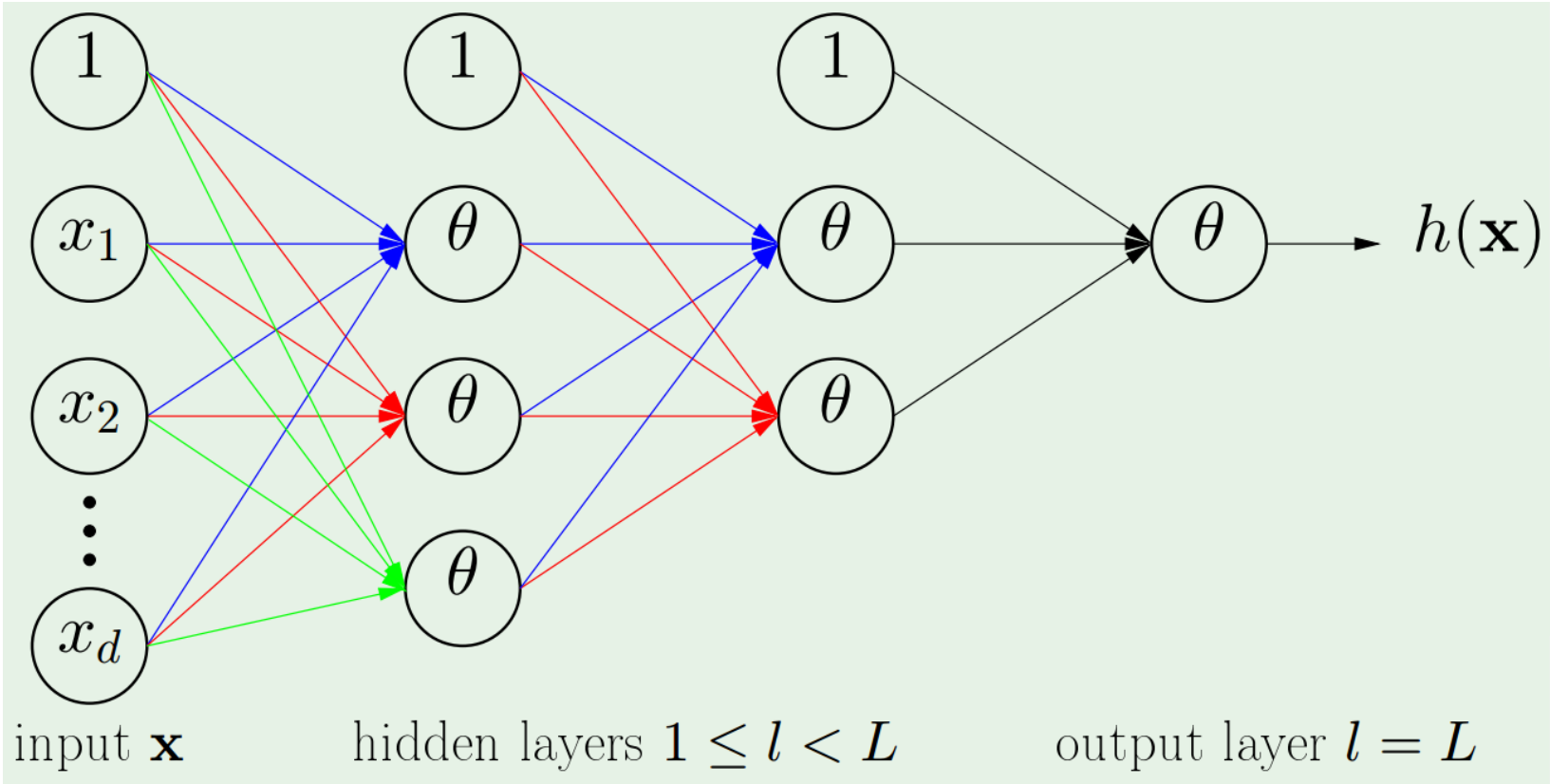
- Имеем экземпляры изображений $(x_1, y_1), \dots (x_n, y_n)$, $y_i \in [0, 1]$
- Инициализируем веса $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$, где m и l – это объём позитивной и негативной выборок
- Для $t = 1, \dots, T$
  - Нормализуем веса: $w_{t,i} = \frac{w_{t,i}}{\sum_{j=0}^{n} w_{t,j}}$
  - Для каждого признака j, обученного классификатора h_j $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$, находим $h_t = argmin_j \epsilon_j$
  - Обновляем веса: $w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$, где $e_i = 0$, если образец x_i классифицирован правильно, $e_i = 1$, иначе, и $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
- Сильный классификатор: $h(x) = 1$, если $\sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t$, и $h(x) = 0$, иначе, где $\alpha_t = \log \frac{1}{\beta_t}$

Модель нейрона МакКалока-Питтса (1943 г.)

- $x_1, \ldots, x_d$ – сигналы на входах нейрона
- $w_1, \ldots, w_d$ – веса входов
- $S = \sum_{i=1}^{d} w_i x_i + w_0 x_0$, где $w_0 x_0$ – порог чувствительности нейрона
- $\theta = \theta(S)$ – это модель нейрона (функция активации, передаточная функция)



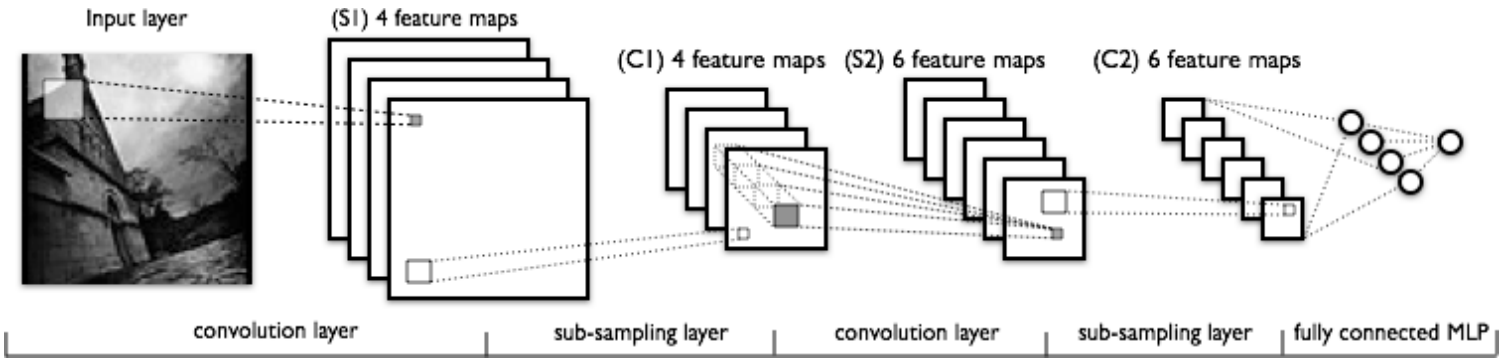
Многослойная нейросеть



Свёрточная нейросеть

Свёрточная нейросеть строится по чередующимся слоям свёртки и подвыборки

1. Слой свёртки формирует карты признаков
2. Слой подвыборки (объединения) уменьшает размерность сформированных карт признаков

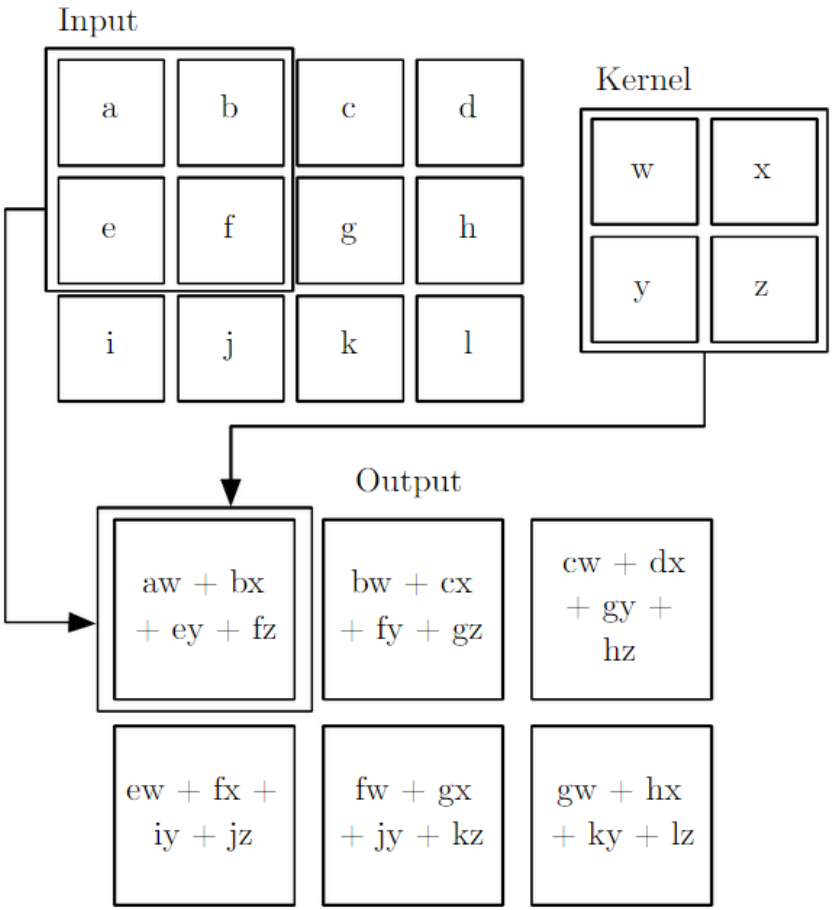


Операция свёртки

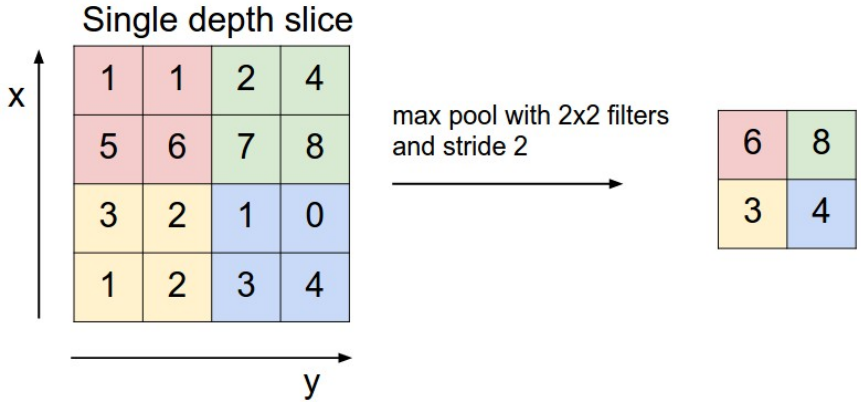
- Пусть $x, w : [0, \infty) \to R$. Свёртка этих функций – это операция вида $(x * w)(t) = \int\limits_0^t x(\tau)w(t-\tau)d\tau$, x называют входом, w – ядром
- Если предположить, что $x, w : Z \to Z$, то получим формулу дискретной свёртки $(x * w)(t) = \sum\limits_{\tau=0}^{t} x(\tau)w(t-\tau)$
- Допустим, что I – входное изображение, K – ядро. Тогда $(I * K)[i, j] = \sum\limits_m \sum\limits_n I[m, n]K[i-m, j-n]$

Свёрточная нейросеть

Операция свёртки



Операция объединения



Для начала

Устанавливаем

1. Python
2. NumPy
3. OpenCV
4. PyCharm

Задача бинарной классификации цифр

- По базе рукописных цифр ▸ MNIST решить задачу бинарной классификации методом градиентного спуска сигмоидальной функции $h(s) = \frac{1}{1+e^{-s}}$, где $s = y \cdot w^T \cdot x$
- Допустим, первый клас будет помечен -1, второй $+1$
- Посчитать ошибки первого и второго рода, т.е. *FP* и *FN*, а вместе с ними TP и TN, по которым правильно проинтерпретировать точность (*precision*) и полноту (*recall*) классификации
- Решение отправить по адресу dkosaty@gmail.com

- Christopher M. Bishop – “Pattern Recognition and Machine Learning”
- Курс Y. Abu-Mostafa ▸ Learning from data
- С 26.01.2016 на Coursera от Yandex доступен курс ▸ Введение в машинное обучение
- ▸ GoogLeNet