

# Введение в искусственный интеллект.

## Машинное обучение

### Тема: Линейные классификаторы

Бабин Д.Н., Иванов И.Е.

кафедра Математической Теории Интеллектуальных Систем



- 1 Понятие линейной классификации
- 2 Биологический нейрон и перцептрон
- 3 Функции активации
- 4 Эмпирические правила обучения и SGD
- 5 Теорема Новикова
- 6 Примеры линейных классификаторов: логистическая регрессия, оптимальный байесовский классификатор

# Напоминание: линейность байесовского классификатора

Из предыдущего материала известно, что оптимальный байесовский бинарный классификатор определяется как:

$$a(x) = \text{sign}(\lambda_+ p(y = +1|x) - \lambda_- p(y = -1|x)) = \text{sign} \left( \frac{p(y=+1|x)}{p(y=-1|x)} - \frac{\lambda_-}{\lambda_+} \right)$$

## Теорема о линейности байесовского классификатора

Если распределения  $p(x|y)$  экспонентны, параметры  $d()$ ,  $\delta$  не зависят от  $y$ , и среди признаков  $x_1, \dots, x_n$  есть константа, то байесовский классификатор линеен:

$$a(x) = \text{sign}(\langle w, x \rangle - w_0), w_0 = \ln \frac{\lambda_-}{\lambda_+};$$

при этом апостериорные вероятности классов  $p(y|x) = \sigma(\langle w, x \rangle y)$ , где  $\sigma(z) = \frac{1}{1+e^{-z}}$  – логистическая функция (сигмоид).



## Мотивация

- 1 Линейные модели возникают как оптимальные байесовские классификаторы для довольно широкого класса распределений
- 2 Линейные модели достаточно хорошо изучены
- 3 Линейные модели могут моделировать довольно сложные зависимости

# Понятие линейной классификации

## Мотивация

- 1 Линейные модели возникают как оптимальные байесовские классификаторы для довольно широкого класса распределений
- 2 Линейные модели достаточно хорошо изучены
- 3 Линейные модели могут моделировать довольно сложные зависимости

## Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

# Понятие линейной классификации

## Мотивация

- 1 Линейные модели возникают как оптимальные байесовские классификаторы для довольно широкого класса распределений
- 2 Линейные модели достаточно хорошо изучены
- 3 Линейные модели могут моделировать довольно сложные зависимости

## Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

- В случае **двух** классов разделяющей поверхностью является **гиперплоскость**, которая делит пространство признаков на два полупространства




# Понятие линейной классификации

## Мотивация

- 1 Линейные модели возникают как оптимальные байесовские классификаторы для довольно широкого класса распределений
- 2 Линейные модели достаточно хорошо изучены
- 3 Линейные модели могут моделировать довольно сложные зависимости

## Линейный классификатор

Это алгоритм классификации, основанный на построении **линейной** разделяющей поверхности

- В случае **двух** классов разделяющей поверхностью является **гиперплоскость**, которая делит пространство признаков на два полупространства
- В случае числа классов **больше двух** разделяющая поверхность **кусочно-линейна** 

- Рассмотрим задачу бинарной классификации:  $X \rightarrow Y$ ,  $Y = \{+1, -1\}$  на обучающей выборке  $X^m = (x_i, y_i)_{i=1}^m$



# Разделяющая поверхность: напоминание

- Рассмотрим задачу бинарной классификации:  $X \rightarrow Y$ ,  $Y = \{+1, -1\}$  на обучающей выборке  $X^m = (x_i, y_i)_{i=1}^m$
- Будем алгоритм искать в виде  $a(x, w) = \text{sign } g(x, w)$ , где  $g(x, w)$  - дискриминантная функция, а  $w$  - вектор параметров



# Разделяющая поверхность: напоминание

- Рассмотрим задачу бинарной классификации:  $X \rightarrow Y$ ,  $Y = \{+1, -1\}$  на обучающей выборке  $X^m = (x_i, y_i)_{i=1}^m$
- Будем алгоритм искать в виде  $a(x, w) = \text{sign } g(x, w)$ , где  $g(x, w)$  - дискриминантная функция, а  $w$  - вектор параметров
- $g(x, w) = 0$  - **разделяющая поверхность** (граница между классами); тогда ошибка классификации  $a(x_i, w) \neq y_i \Leftrightarrow y_i g(x_i, w) < 0$ .



## Два класса

Дискриминантная функция:

$g(x, w) = \sum_{j=1}^n w_j f_j - w_0$ , где  
 $f_j : X \rightarrow \mathbb{R}$  – числовые признаки.

Алгоритм классификации

$a(x, w) = \text{sign}(\sum_{j=1}^n w_j f_j - w_0)$ .

Если ввести константный признак  $f_0 \equiv -1$ , то

$x = (f_0(x), \dots, f_n(x))$ ,

и алгоритм в векторной записи:

$a(x, w) = \text{sign}(\langle w, x \rangle)$ .

$y_i g(x_i, w) = \langle w, x_i \rangle y_i$



# Линейная классификация: определения

## Два класса

Дискриминантная функция:

$g(x, w) = \sum_{j=1}^n w_j f_j - w_0$ , где  
 $f_j : X \rightarrow \mathbb{R}$  – числовые признаки.

Алгоритм классификации

$a(x, w) = \text{sign}(\sum_{j=1}^n w_j f_j - w_0)$ .

Если ввести константный признак  $f_0 \equiv -1$ , то

$x = (f_0(x), \dots, f_n(x))$ ,

и алгоритм в векторной записи:

$a(x, w) = \text{sign}(\langle w, x \rangle)$ .

$y_i g(x_i, w) = \langle w, x_i \rangle y_i$

## Произвольное число классов

У каждого класса  $c \in Y$  свой вектор весов:  $w^c = (w_0^c, \dots, w_n^c)$ .

Линейный классификатор:

$a(x, w) = \arg \max_{c \in Y} \sum_{j=0}^n w_j^c f_j(x) =$   
 $\arg \max_{c \in Y} \langle w^c, x \rangle$ .

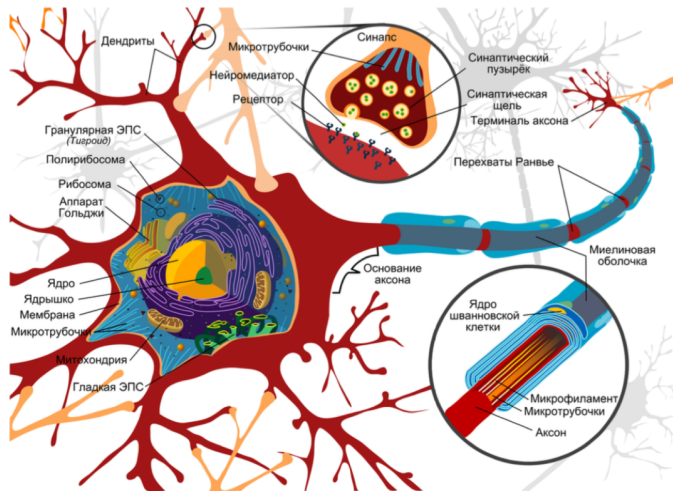
$y_i g(x_i, w) =$   
 $\langle x_i, w^{y_i} \rangle - \max_{c \in Y, c \neq y_i} \langle x_i, w^c \rangle$

**Замечание.** Обратите внимание на разницу со случаем двух классов!



# Биологический нейрон

- Кора головного мозга содержит  $10^{11}$  нейронов
- Каждый нейрон связан синапсами с  $10^3 - 10^4$  другими нейронами
- Скорость распространения импульсов 100 м/с
- Входы (много) - дендриты
- Выход (один) - аксон

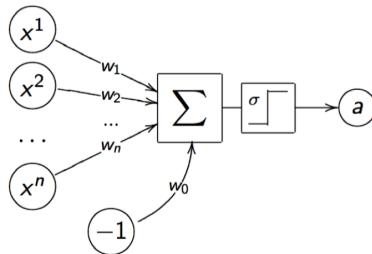


# Математическая модель нейрона

Предложена МакКаломом и Питтсом в 1943 году<sup>1</sup>.

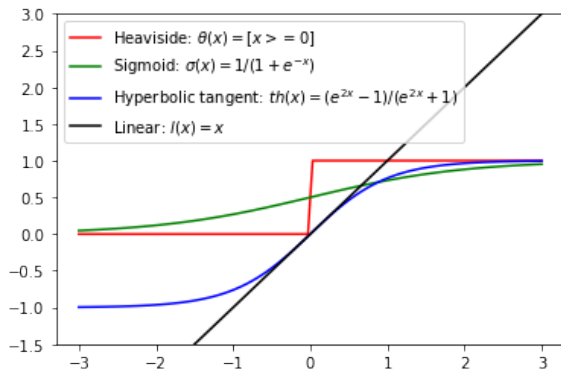
$$a(x, w) = \sigma(\langle w, x \rangle) = \sigma\left(\sum_{j=1}^n w_j f_j - w_0\right)$$

где  $\sigma(x)$  - некоторая функция активации (например, sign).



<sup>1</sup>McCulloch, W. S. and Pitts, W. (1943). "A logical calculus of the ideas immanent in nervous activity"

# Примеры функций активаций





## Правило Хэбба, 1949<sup>2</sup>

В задаче бинарной ( $Y = \{-1, +1\}$ ) классификации линейный классификатор:

$$a(x, w) = \text{sign}(\langle w, x \rangle)$$

Функция потерь:  $L(a(x_i, w), y_i) = [a(x_i, w) \neq y_i]$ .

Шаг обновления: если  $a(x_i, w^{(t)}) \neq y_i \Leftrightarrow a(x_i, w^{(t)})y_i < 0$ , то  $w^{(t+1)} = w^{(t)} + \eta x_i y_i$

<sup>2</sup>Hebb, D. O. (1949). "The organization of behavior: a neuropsychological theory."

<sup>3</sup>Rosenblatt, F. (1957). "The perceptron, a perceiving and recognizing automaton"

# Историческая справка: правила Хэбба и Розенблатта

## Правило Хэбба, 1949<sup>2</sup>

В задаче бинарной ( $Y = \{-1, +1\}$ ) классификации линейный классификатор:

$$a(x, w) = \text{sign}(\langle w, x \rangle)$$

Функция потерь:  $L(a(x_i, w), y_i) = [a(x_i, w) \neq y_i]$ .

Шаг обновления: если  $a(x_i, w^{(t)}) \neq y_i \Leftrightarrow a(x_i, w^{(t)})y_i < 0$ , то  $w^{(t+1)} = w^{(t)} + \eta x_i y_i$

## Правило перцептрона Розенблатта, 1957<sup>3</sup>

Пусть  $X = \{0, 1\}^n$ ,  $Y = \{0, +1\}$ , линейный классификатор — это функция Хевисайда

$$a(x, w) = \theta(\langle w, x \rangle) = [\langle w, x \rangle > 0]. \text{ Тогда:}$$

если  $a(x_i, w^{(t)}) \neq y_i$ :  $w^{(t+1)} = w^{(t)} + \eta x_i$ , если  $y_i = 1$ , и  $w^{(t+1)} = w^{(t)} - \eta x_i$ , если  $y_i = 0$

<sup>2</sup>Hebb, D. O. (1949). "The organization of behavior: a neuropsychological theory."

<sup>3</sup>Rosenblatt, F. (1957). "The perceptron, a perceiving and recognizing automaton"

# SGD для линейной регрессии: ADALINE

В задаче регрессии функция потерь:

$$L(a(x_i, w), y_i) = \frac{1}{2}(a(x_i, w) - y_i)^2$$

Эмпирическое правило обновления весов — т.н. **дельта-правило**:

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

Адаптивный линейный нейрон (ADaptive Linear NEuron) ADALINE предложен Уидроу и Хоффом в 1960<sup>4</sup>:  $a(x, w) = \langle w, x \rangle$

**Дельта-правило** в случае ADALINE совпадает с градиентным шагом стохастического градиентного спуска:

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

---

<sup>4</sup>Widrow, B. and Hoff, M. E. (1960). "Adaptive switching circuits"

**Дельта-правило** (эмпирическое правило обновления весов):

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

Т.о., правило Хэбба и правило Розенблатта — суть одно и то же (а именно, дельта-правило), и совпадают с правилом ADALINE (которое является градиентным шагом стохастического градиентного спуска) с заменой  $\langle w^{(t)}, x_i \rangle$  на:

- $a(x, w) = \text{sign}(\langle w, x \rangle)$  в случае правила Хэбба,
- $a(x, w) = \theta(\langle w, x \rangle)$  в случае правила Розенблатта.



# Теорема Новикова<sup>5</sup>

Задача бинарной классификации  $X = \mathbb{R}^{n+1}$ ,  $Y = \{-1, +1\}$ .

## Теорема Новикова, 1962

Пусть выборка  $X^m$  линейно разделима, т.е.  $\exists \tilde{w}, \|\tilde{w}\| = 1, \exists \delta > 0 : \langle \tilde{w}, x_i \rangle y_i > \delta$  для всех  $i = 1, \dots, m$ . Пусть начальный вектор весов  $w^0 = 0$ . Также в процедуре обучения каждый объект обучающей выборки появляется повторно через некоторый конечный интервал времени.

Тогда алгоритм SGD с правилом Хэбба находит вектор весов  $w$ :

- разделяющий выборку без ошибок,
- при любом шаге градиентного спуска  $\eta$ ,
- независимо от порядка предъявления  $x_i$ ,
- за конечное число исправлений вектора  $w$ :  $t_{max} \leq \frac{1}{\delta^2} \max_i \|x_i\|^2$

<sup>5</sup>Novikoff, A. (1962). "On Convergence Proofs on Perceptrons"



# Теорема Новикова: доказательство

С одной стороны,

$$\langle \tilde{w}, w^t \rangle = \langle \tilde{w}, w^{t-1} \rangle + \eta \langle \tilde{w}, x_i \rangle y_i > \langle \tilde{w}, w^{t-1} \rangle + \eta \delta > \dots > \langle \tilde{w}, w^0 \rangle + t\eta\delta = t\eta\delta.$$

С другой стороны, поскольку выборка конечна,  $\exists D > 0 : \|x_i\| < D$  для всех  $i$ . В силу этого  $\|w^t\|^2 = \|w^{t-1}\|^2 + \eta^2 \|x_i\|^2 + 2\eta \langle w^{t-1}, x_i \rangle y_i$ . Так как для применения правила Хэбба должно быть  $\langle w^{t-1}, x_i \rangle y_i < 0$ , то

$$\|w^t\|^2 < \|w^{t-1}\|^2 + \eta^2 D^2 < \dots < \|w^0\|^2 + t\eta^2 D^2 = t\eta^2 D^2.$$

По неравенству Коши-Буняковского  $\langle \tilde{w}, w^t \rangle \leq \|\tilde{w}\| \cdot \|w^t\|$ .

Объединяя эти неравенства, получаем  $\eta\delta t \leq \langle \tilde{w}, w^t \rangle \leq \eta D \sqrt{t} \cdot \|\tilde{w}\|$ , или  $\sqrt{t} \leq \frac{D}{\delta}$ .

Т.о. при  $t > \frac{D^2}{\delta^2}$  не найдётся ни одного  $x_i$ , т.ч.  $\langle w^t, x_i \rangle y_i < 0$ , т.е. вся выборка будет правильно классифицирована. Ч.т.д.





# Линейность байесовского классификатора

Из предыдущего материала известно, что оптимальный байесовский бинарный классификатор определяется как:

$$a(x) = \text{sign}(\lambda_+ p(y = +1|x) - \lambda_- p(y = -1|x)) = \text{sign} \left( \frac{p(y=+1|x)}{p(y=-1|x)} - \frac{\lambda_-}{\lambda_+} \right)$$

## Теорема о линейности байесовского классификатора

Если распределения  $p(x|y)$  экспонентны, параметры  $d()$ ,  $\delta$  не зависят от  $y$ , и среди признаков  $x_1, \dots, x_n$  есть константа, то байесовский классификатор линеен:

$$a(x) = \text{sign}(\langle w, x \rangle - w_0), w_0 = \ln \frac{\lambda_-}{\lambda_+};$$

при этом апостериорные вероятности классов  $p(y|x) = \sigma(\langle w, x \rangle y)$ , где  $\sigma(z) = \frac{1}{1+e^{-z}}$  – логистическая функция (сигмоид).





# Линейность байесовского классификатора

Из предыдущего материала известно, что оптимальный байесовский бинарный классификатор определяется как:


$$a(x) = \text{sign}(\lambda_+ p(y = +1|x) - \lambda_- p(y = -1|x)) = \text{sign} \left( \frac{p(y=+1|x)}{p(y=-1|x)} - \frac{\lambda_-}{\lambda_+} \right)$$

## Теорема о линейности байесовского классификатора

Если распределения  $p(x|y)$  экспонентны, параметры  $d()$ ,  $\delta$  не зависят от  $y$ , и среди признаков  $x_1, \dots, x_n$  есть константа, то байесовский классификатор линеен:

$$a(x) = \text{sign}(\langle w, x \rangle - w_0), w_0 = \ln \frac{\lambda_-}{\lambda_+};$$

при этом апостериорные вероятности классов  $p(y|x) = \sigma(\langle w, x \rangle y)$ , где  $\sigma(z) = \frac{1}{1+e^{-z}}$  – логистическая функция (сигмоид).

Т.о., от решающего правила типа  $a(x, w) = \theta(\langle w, x \rangle)$  перешли к правилу  $a(x, w) = [\sigma(\langle w, x \rangle) > \frac{1}{2}]$ , но все так же от линейной функции по входу; однако при этом дополнительно приобрели возможность оценивать вероятность принадлежности к классу 

# Логарифмическая функция потерь и логистическая регрессия

## Определение логистической регрессии

Классификационная бинарная модель, в которой вероятность принадлежности к положительному классу задаётся **сигмоидом** от **линейной функции** по входу.

# Логарифмическая функция потерь и логистическая регрессия

## Определение логистической регрессии

Классификационная бинарная модель, в которой вероятность принадлежности к положительному классу задаётся **сигмоидом** от **линейной функции** по входу.

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^m p(x_i, y_i) \rightarrow \max_w$



## Определение логистической регрессии

Классификационная бинарная модель, в которой вероятность принадлежности к положительному классу задаётся **сигмоидом** от **линейной функции** по входу.

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^m p(x_i, y_i) \rightarrow \max_w$

Подставим в формулу выражение для логистической регрессии

$$p(x, y) = p(y|x) \cdot p(x) = \sigma(\langle w, x \rangle) \cdot \text{const}(w):$$

- $L(w, X^m) = \sum_{i=1}^m \log \sigma(\langle w, x_i \rangle y_i) + \text{const}(w) \rightarrow \max_w$



## Определение логистической регрессии

Классификационная бинарная модель, в которой вероятность принадлежности к положительному классу задаётся **сигмоидом** от **линейной функции** по входу.

Напоминание: максимизация логарифма правдоподобия:

- $L(w, X^m) = \log \prod_{i=1}^m p(x_i, y_i) \rightarrow \max_w$

Подставим в формулу выражение для логистической регрессии

$$p(x, y) = p(y|x) \cdot p(x) = \sigma(\langle w, x \rangle) \cdot \text{const}(w):$$

- $L(w, X^m) = \sum_{i=1}^m \log \sigma(\langle w, x_i \rangle y_i) + \text{const}(w) \rightarrow \max_w$

Максимизация  $L$  эквивалентна минимизации аппроксимированного Э.Р. с логарифмической функцией потерь  $R$ :

$$R(w, X^m) = \sum_{i=1}^m \log(1 + \exp(-\langle w, x_i \rangle y_i)) \rightarrow \min_w$$



# Многоклассовая логистическая регрессия

Рассмотрим случай произвольного количества классов  $|Y| > 2$ . Тогда линейный классификатор (напоминание):

$$a(x) = \arg \max_{c \in Y} \langle w^c, x \rangle \quad x, w^c \in \mathbb{R}^n$$



# Многоклассовая логистическая регрессия

Рассмотрим случай произвольного количества классов  $|Y| > 2$ . Тогда линейный классификатор (напоминание):

$$a(x) = \arg \max_{c \in Y} \langle w^c, x \rangle \quad x, w^c \in \mathbb{R}^n$$

Вероятность принадлежности объекта  $x$  к классу  $c$  определяется т.н. функцией SoftMax:

$$\text{SoftMax}(\langle w^c, x \rangle) = P(y = c | x, w) = \frac{\exp(\langle w^c, x \rangle)}{\sum_{z \in Y} \exp(\langle w^z, x \rangle)}$$

Т.о. функция  $\text{SoftMax} : \mathbb{R}^{|Y|} \rightarrow \mathbb{R}^{|Y|}$  преобразует любой вещественнозначный вектор в вектор дискретного распределения.



- 1 Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)



- 1 Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)
- 2 При линейно разделимых множествах процедура построения разделяющей поверхности конечна

- 1 Линейный классификатор – предельный простой случай (тем не менее, работающий на практике!)
- 2 При линейно разделимых множествах процедура построения разделяющей поверхности конечна
- 3 Все эмпирические правила классификации и регрессии — это частные случаи SGD

