

Разбор летучки

Лекция 6

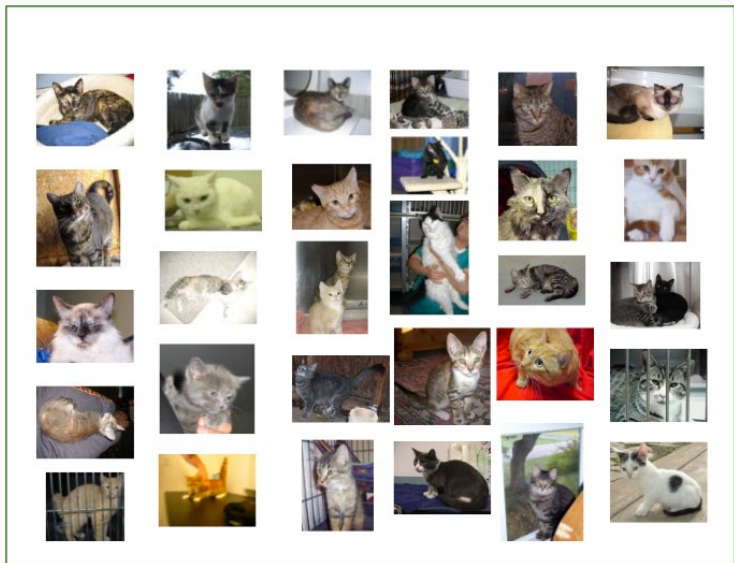
Линейные методы классификации

Екатерина Тузова

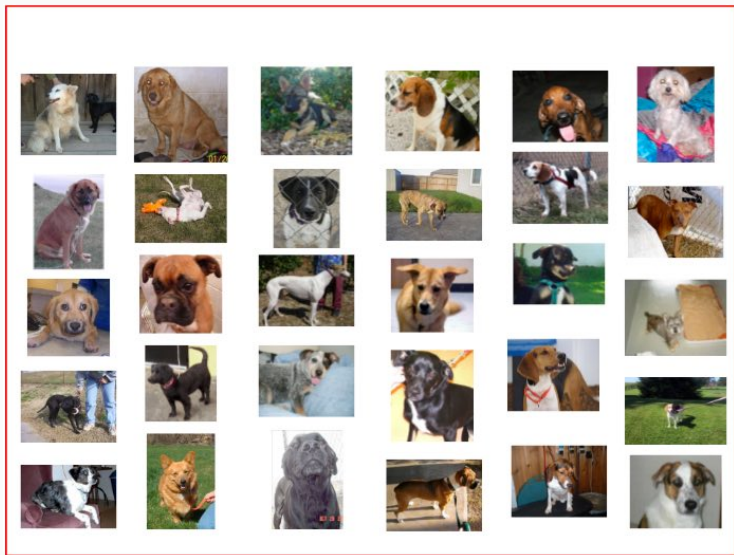
Мотивирующий пример



Пример



Пример



Постановка задачи

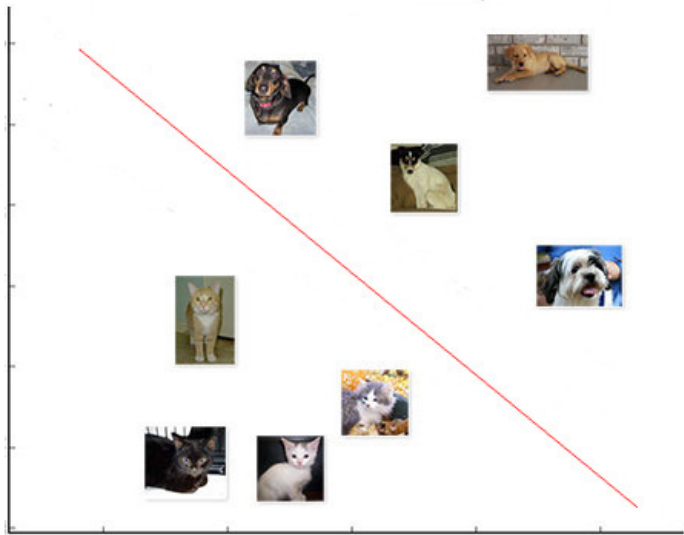
$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

$X^l = (x_i, y_i)_{i=1}^l$ – обучающая выборка

Задача: Построить алгоритм $a: X \rightarrow Y$, способный классифицировать произвольный объект $x \in X$.

Идея: $(n - 1)$ -мерную гиперплоскость, которая разделяет данные.

Пример



Модель McCulloch-Pitts

$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

Модель McCulloch-Pitts

$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

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

x^j — признаки объекта, $j = 1, \dots, n$

$w_j \in R$ — веса признаков

$\sigma(s)$ — функция активации (например, sign)

Модель McCulloch-Pitts

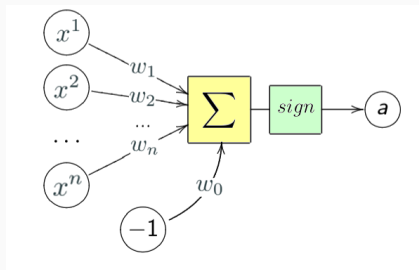
$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

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

x^j — признаки объекта, $j = 1, \dots, n$

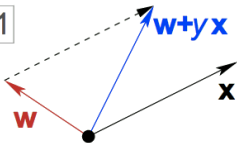
$w_j \in R$ — веса признаков

$\sigma(s)$ — функция активации (например, sign)

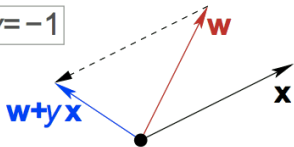


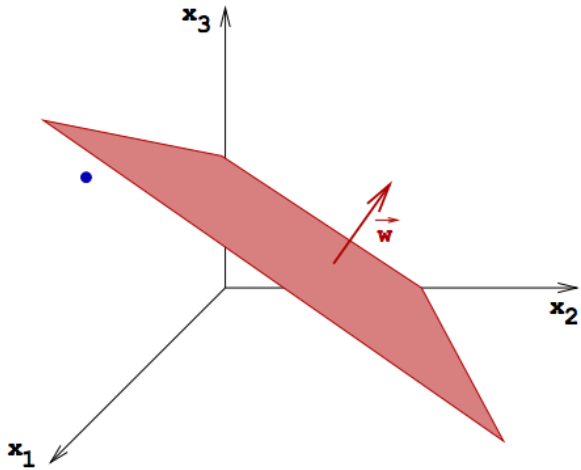
Как подобрать веса w ?

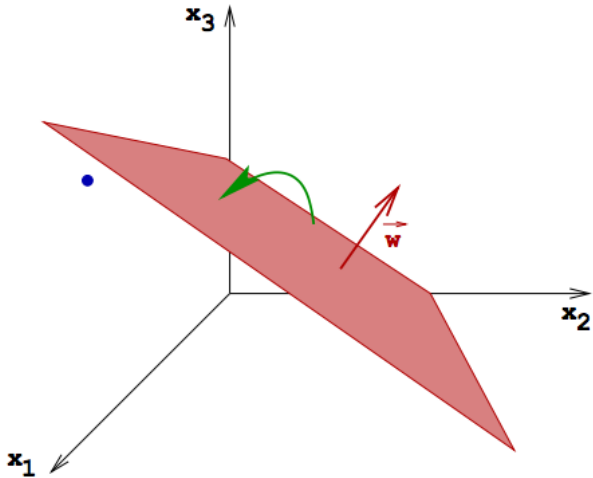
$$y=+1$$

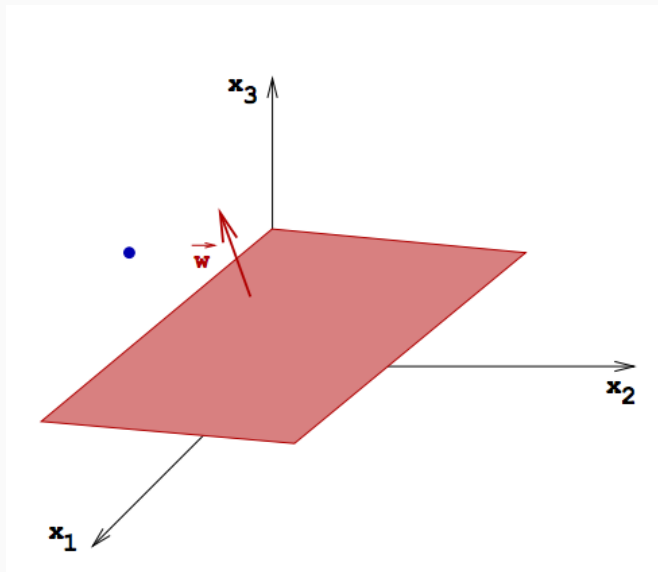


$$y=-1$$









Perceptron Learning Algorithm

```
1 function PLA( $X^l$ )
2   Инициализировать  $w_0, \dots, w_n$ 
3   repeat[пока  $\mathbf{w}$  изменяются]
4     for  $i = 1, \dots, l$  do
5       if  $a(\mathbf{x}_i) \neq y_i$  then
6          $\mathbf{w} = \mathbf{w} + y_i \mathbf{x}_i$ 
```

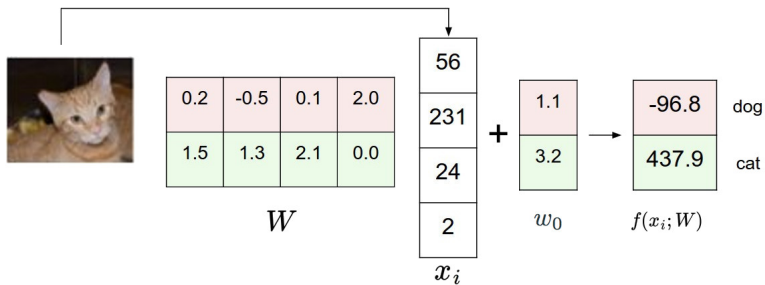
**Что делать в случае
нескольких классов?**

Идея: Для каждого класса заведём отдельный вектор w_y .

Идея: Для каждого класса заведём отдельный вектор \mathbf{w}_y .

$$a(x) = \arg \max_y \langle \mathbf{w}_y, \mathbf{x} \rangle$$

Пример



Multiclass PLA

```
1 function MULTICLASS_PLA( $X^l$ )
2   Инициализировать  $w_0, \dots, w_n$ 
3   repeat[пока  $w$  изменяются]
4     for  $i = 1, \dots, l$  do
5        $a(x_i) = \arg \max_y \langle w_y, x_i \rangle$ 
6       if  $a(x_i) \neq y_i$  then
7          $w_y = w_y - x_i$ 
8          $w_{y^*} = w_{y^*} + x_i$ 
```

Будем улучшать алгоритм

```
1 function MULTICLASS_PLA( $X^l, \alpha$ )
2   Инициализировать  $w_0, \dots, w_n$ 
3   repeat[пока  $w$  изменяются]
4     for  $i = 1, \dots, l$  do
5        $a(x_i) = \arg \max_y \langle w_y, x_i \rangle$ 
6       if  $a(x_i) \neq y_i$  then
7          $w_y = w_y - \alpha x_i$ 
8          $w_{y^*} = w_{y^*} + \alpha x_i$ 
```

- + Алгоритм гарантированно сходится, если обучающая выборка линейно разделима

- + Алгоритм гарантированно сходится, если обучающая выборка линейно разделима
- Не работает в случае неразделимой выборки

- + Алгоритм гарантированно сходится, если обучающая выборка линейно разделима
- Не работает в случае неразделимой выборки
- "Слишком правильные" предсказания добавляют штраф

Что делать, если данные не
линейно делимы?

Идея: На этапе предобработки скомбинировать признаки в полиномиальные признаки.

Обобщённый линейный классификатор

$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

$X^l = (x_i, y_i)_{i=1}^l$ – обучающая выборка

Задача:

$(n - 1)$ -мерную гиперплоскость, которая разделяет данные **как можно лучше**.

$$X = \mathbb{R}^n, Y = \{-1, +1\}$$

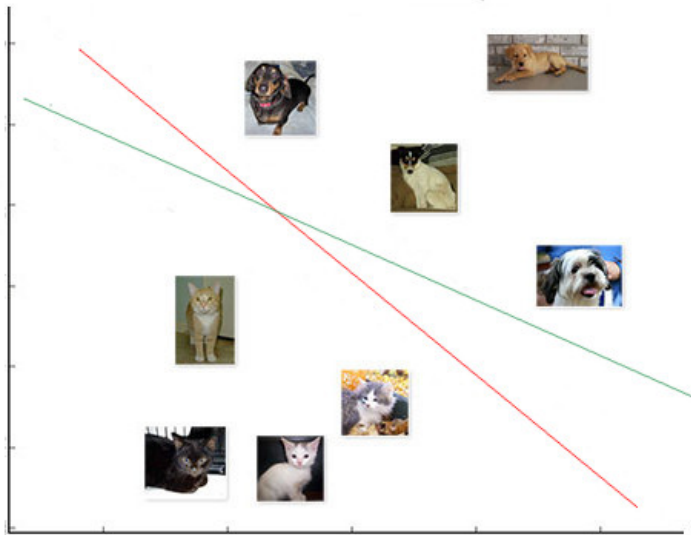
$X^l = (x_i, y_i)_{i=1}^l$ – обучающая выборка

Задача:

$(n - 1)$ -мерную гиперплоскость, которая разделяет данные **как можно лучше**.

Как можно лучше – это как?

Пример



$g = \langle \mathbf{w}, \mathbf{x} \rangle = 0$ – разделяющая поверхность

$M_i(\mathbf{w}) = \langle \mathbf{w}, \mathbf{x}_i \rangle y_i$ – отступ объекта x_i

$M_i(\mathbf{w}) < 0 \Rightarrow$ алгоритм $a(\mathbf{x}, \mathbf{w})$ ошибается на x_i

Число ошибок на обучающей выборке:

$$Q(\mathbf{w}) = \sum_{i=1}^l [M_i(\mathbf{w}) < 0] \rightarrow \min_{\mathbf{w}}$$

Число ошибок на обучающей выборке:

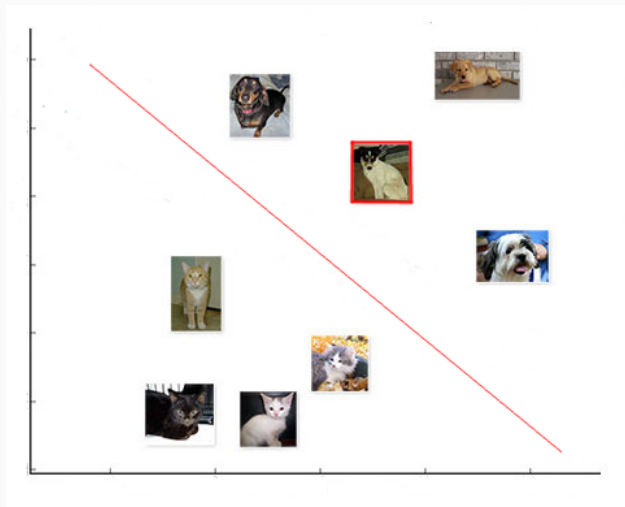
$$Q(\mathbf{w}) = \sum_{i=1}^l [M_i(\mathbf{w}) < 0] \rightarrow \min_{\mathbf{w}}$$

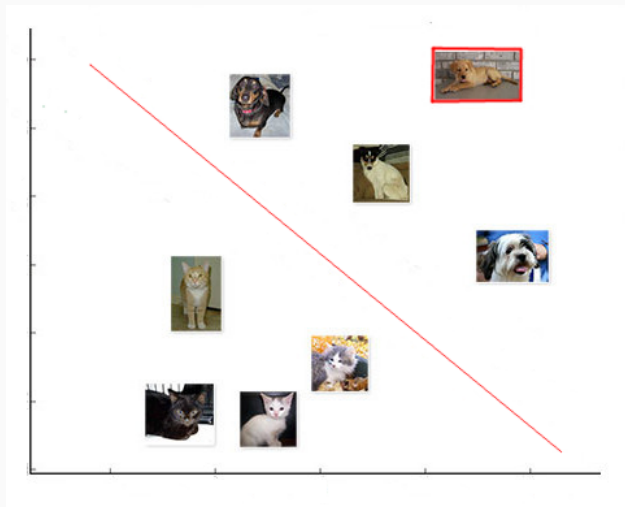
– Индикаторную функцию сложно оптимизировать

Число ошибок на обучающей выборке:

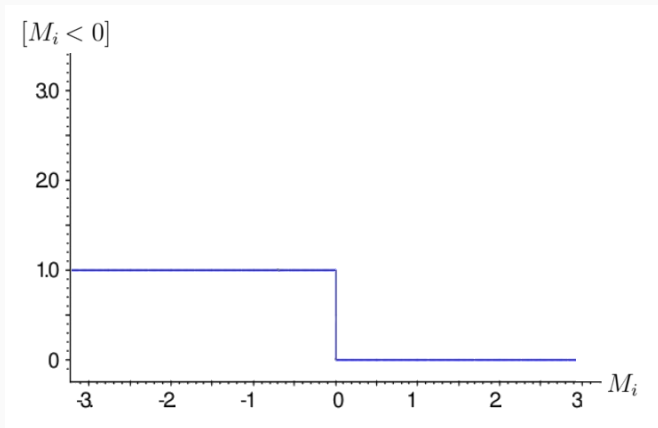
$$Q(\mathbf{w}) = \sum_{i=1}^l [M_i(\mathbf{w}) < 0] \rightarrow \min_{\mathbf{w}}$$

- Индикаторную функцию сложно оптимизировать
- Теряем информацию насколько i -й объект был надежен





Функция $[M < 0]$

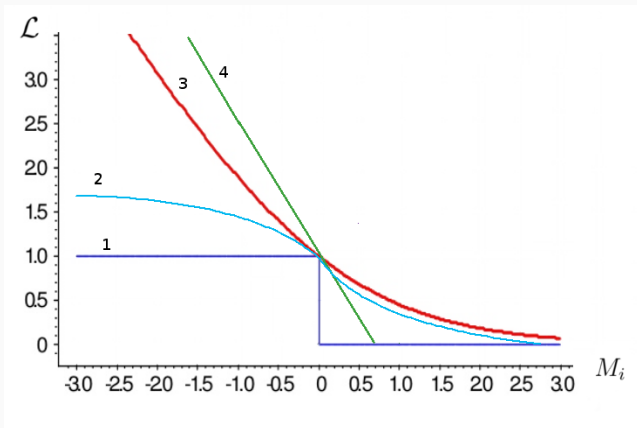


$$Q(\mathbf{w}) = \sum_{i=1}^l [M_i(\mathbf{w}) < 0] \leq \sum_{i=1}^l \mathcal{L}(M_i(\mathbf{w})) \rightarrow \min_{\mathbf{w}}$$

\mathcal{L} – функция потерь, невозрастающая, неотрицательная.

\mathcal{L} должна мажорировать $[M_i(\mathbf{w}) < 0]$

Примеры \mathcal{L}



1. $[M_i(\mathbf{w}) < 0]$
2. $S(M) = 2(1 + e^M)^{-1}$ – сигмоидная
3. $L(M) = \log_2(1 + e^{-M})$ – логарифмическая
4. $V(M) = (1 - M)_+$ – кусочно-линейная

Что такое градиент?

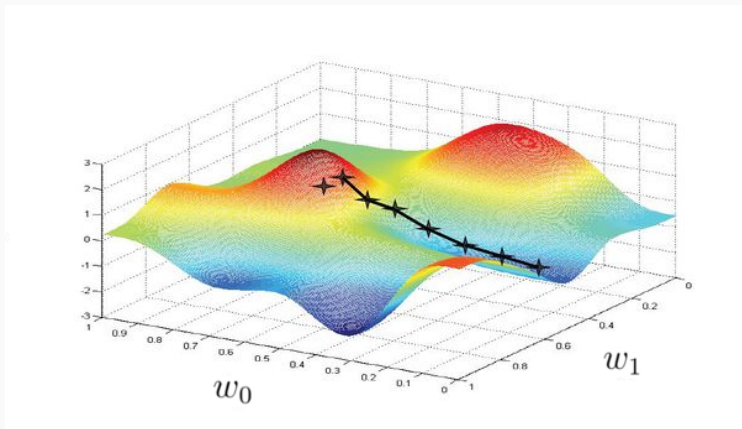


Метод градиентного спуска

```
1 function GRADIENT_DESCENT( $X^l, \alpha$ )
2   Инициализировать  $\mathbf{w}$ 
3   repeat[пока  $\mathbf{w}$  изменяются]
4      $\mathbf{w} = \mathbf{w} - \alpha \nabla Q(\mathbf{w})$ 
```

$$\nabla Q(\mathbf{w}) = \left(\frac{\partial Q(\mathbf{w})}{\partial w_j} \right)_{j=0}^n = \sum_{i=1}^l \mathcal{L}'(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i) \mathbf{x}_i y_i$$

Градиентный спуск

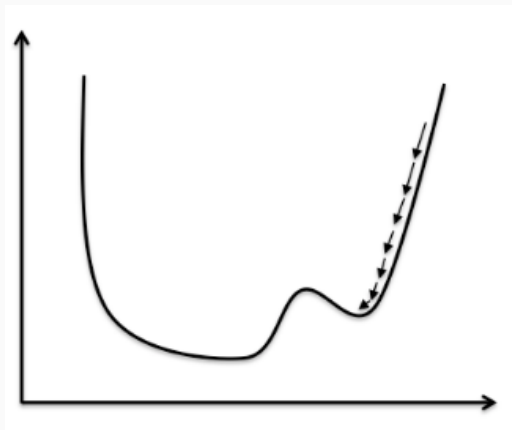


Случай двух признаков.

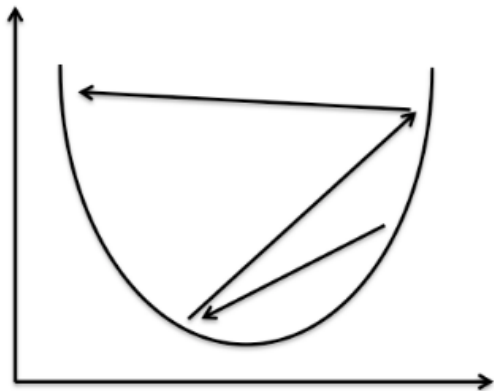
```
1 function GRADIENT_DESCENT( $X^l, \alpha$ )
2   Инициализировать  $w_0, w_1$ 
3   repeat[пока  $w_0$  и  $w_1$  не стабилизируются]
4      $tmp_0 = w_0 - \alpha \frac{\partial Q(w)}{\partial w_0}$ 
5      $tmp_1 = w_1 - \alpha \frac{\partial Q(w)}{\partial w_1}$ 
6      $w_0 = tmp_0$ 
7      $w_1 = tmp_1$ 
```


Почему важно обновить w_0 и w_1 одновременно?

Маленький градиентный шаг



Большой градиентный шаг



- $\alpha_t \rightarrow 0$
- Метод скорейшего градиентного спуска:
 $Q(w - \alpha \nabla Q(w)) \rightarrow \min_{\alpha}$
- Пробные случайные шаги

В чем проблема?

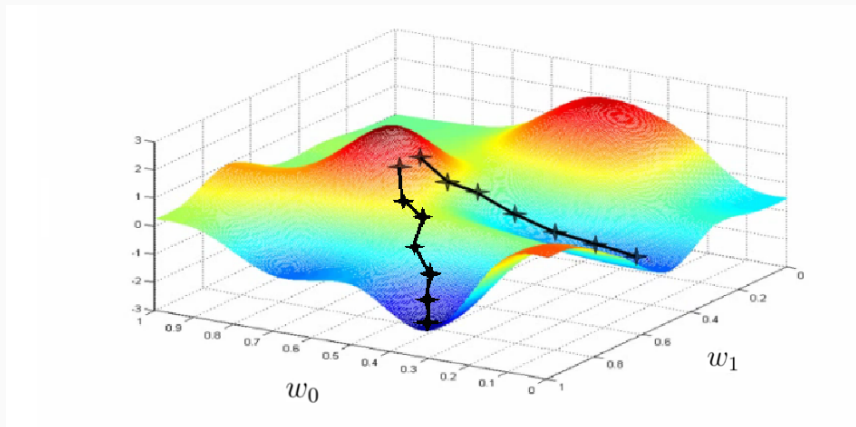
$$\mathbf{w} = \mathbf{w} - \alpha \sum_{i=1}^l \mathcal{L}'(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i) \mathbf{x}_i y_i$$

$$\mathbf{w} = \mathbf{w} - \alpha \sum_{i=1}^l \mathcal{L}'(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i) \mathbf{x}_i y_i$$

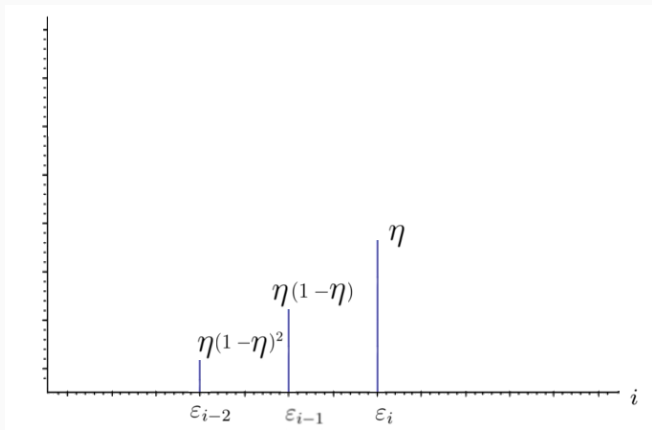
Идея: Давайте брать (x_i, y_i) по одному и сразу обновлять вектор весов

```
1 function STOCHASTIC_GRADIENT( $X^l, \alpha, \eta$ )
2   Перемешать данные в  $X^l$ 
3   Инициализировать  $\mathbf{w}$ 
4    $Q(\mathbf{w}) = \sum_{i=1}^l \mathcal{L}(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i)$ 
5   repeat[пока  $Q$  и/или  $w$  не стабилизируются]
6     Взять  $\mathbf{x}_i$  из  $X^l$ 
7     Потеря:  $\varepsilon_i = \mathcal{L}(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i)$ 
8     Градиентный шаг:  $\mathbf{w} = \mathbf{w} - \alpha \mathcal{L}'(\langle \mathbf{w}, \mathbf{x}_i \rangle y_i) \mathbf{x}_i y_i$ 
9     Оценить  $Q = (1 - \eta)Q + \eta \varepsilon_i$ 
```


Градиентный спуск

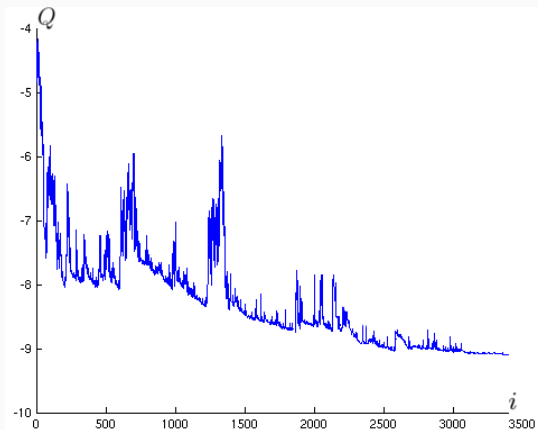


Учет ошибки ε_i в алгоритме

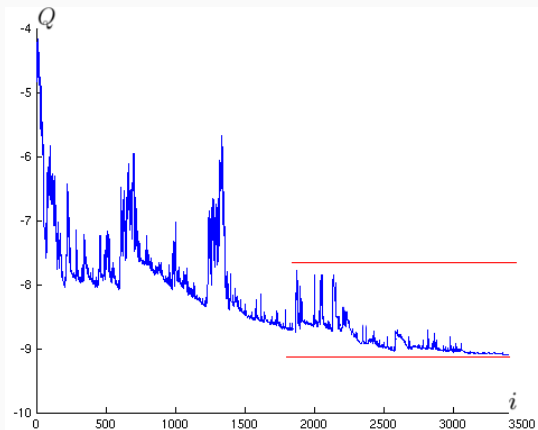


Что значит – "пока Q и/или w не стабилизируются"?

Зависимость Q от номера итерации



Зависимость Q от номера итерации



- Инициализация весов

- Инициализация весов
- Порядок предъявления объектов

Инициализация весов

- $w_j = 0, j = 1, \dots, n$
- $w_j = \text{random}(-\frac{1}{2n}, \frac{1}{2n})$ – небольшие случайные значения
- Обучение по небольшой случайной подвыборке объектов
- Многократный запуск из разных случайных начальных приближений

Порядок предъявления объектов

Порядок предъявления объектов

- Попеременно брать объекты из разных классов
- Чаще брать те объекты, на которых была допущена большая ошибка
- Вообще не брать "хорошие" объекты с $M_i > \mu_+$
- Вообще не брать выбросы с $M_i < \mu_-$

+ Легко реализовать

- + Легко реализовать
- + Легко обобщить на разные g, \mathcal{L}

- + Легко реализовать
- + Легко обобщить на разные g, \mathcal{L}
- + Не обязательно брать все элементы выборки для обучения

- + Легко реализовать
- + Легко обобщить на разные g, \mathcal{L}
- + Не обязательно брать все элементы выборки для обучения
- Возможна медленная сходимость

- + Легко реализовать
- + Легко обобщить на разные g, \mathcal{L}
- + Не обязательно брать все элементы выборки для обучения
- Возможна медленная сходимость
- Застревание в локальных минимумах

- + Легко реализовать
- + Легко обобщить на разные g, \mathcal{L}
- + Не обязательно брать все элементы выборки для обучения

- Возможна медленная сходимость
- Застревание в локальных минимумах
- Подбор параметров

- + Легко реализовать
- + Легко обобщить на разные g , \mathcal{L}
- + Не обязательно брать все элементы выборки для обучения

- Возможна медленная сходимость
- Застревание в локальных минимумах
- Подбор параметров
- Проблема переобучения

Вопросы?

Что почитать по этой лекции

- Professor Yaser Abu-Mostafa MOOC
- Tom Mitchell "Machine Learning" Chapter 4

На следующей лекции

- Функционалы качества
- Неравенство Хефдинга
- Близость гипотез
- Неравенство Вапника-Червоненкиса
- Генерация модельных данных