Курс “Анализ изображений”

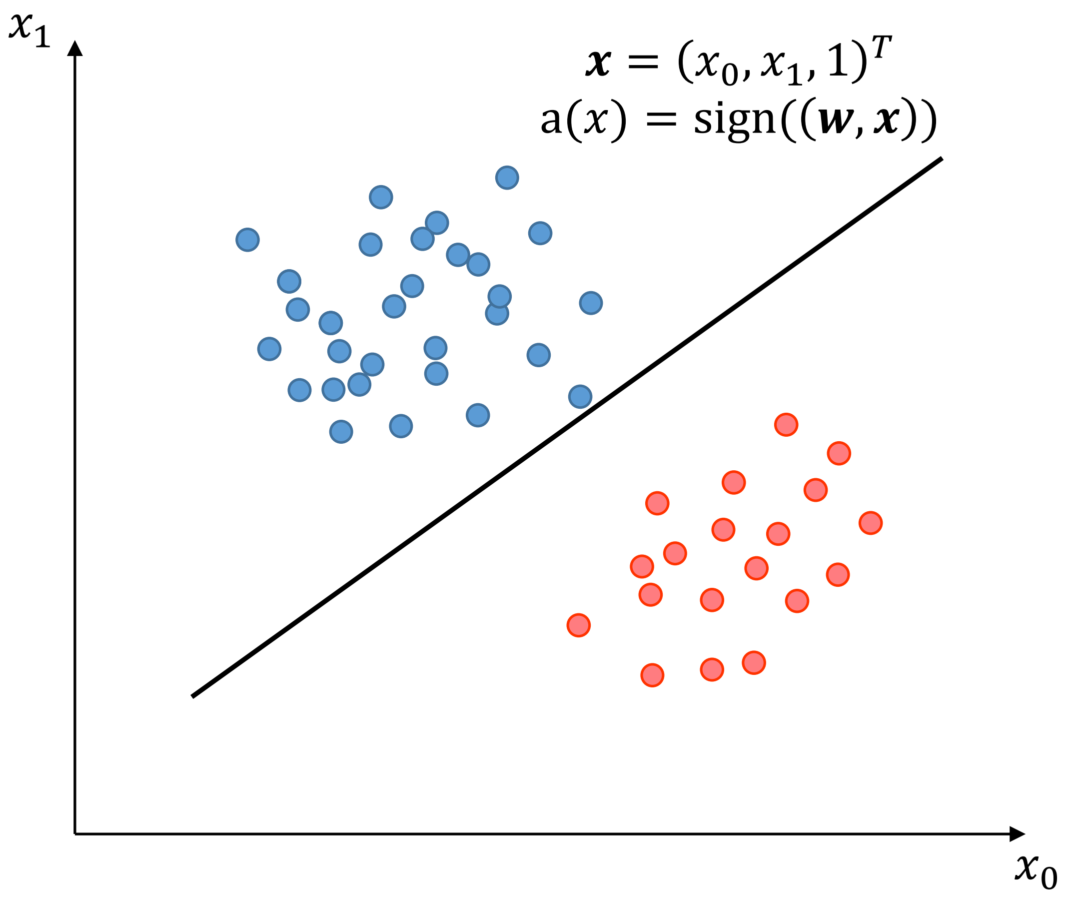
Лекция#8. Основы машинного обучения. Линейные классификаторы. Метод опорных векторов.

ФИВТ МФТИ

2017

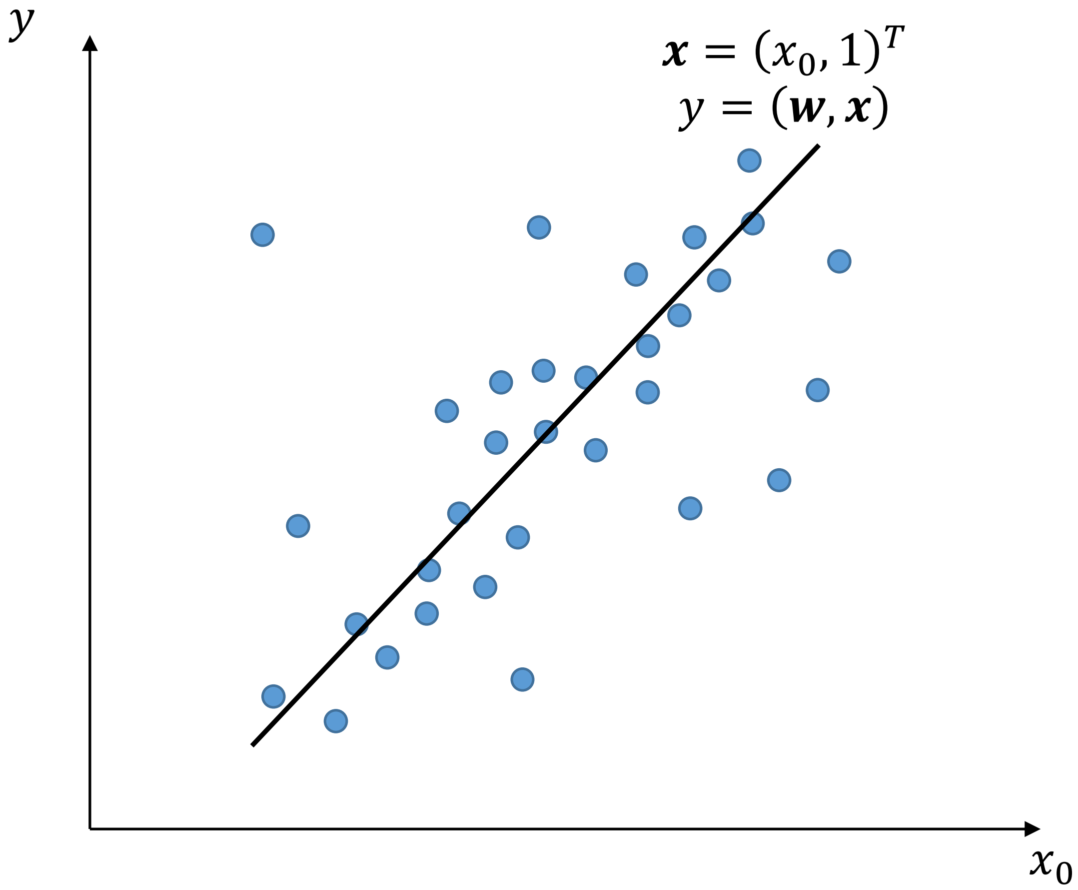
Задача классификации

- X — множество векторов
- $Y = \{y\}_{i=1}^{K}$ — множество классов
- Известно: $f: X \to Y$ на конечном (обучающем) наборе $\{x\}_{i=1}^{N}$
- Задача: построить $f: X \to Y$, определенную на всем множестве X
- Бинарная классификация:
$$Y = \{-1,1\}$$
- Детектирование: 1 класс против всего остального



Задача регрессии

- X — множество векторов
- $Y \equiv R^n$ — множество значений
- Известно: $f: X \to Y$ на конечном наборе (обучающем)
- Задача: построить $f: X \to Y$, определенную на всем множестве X

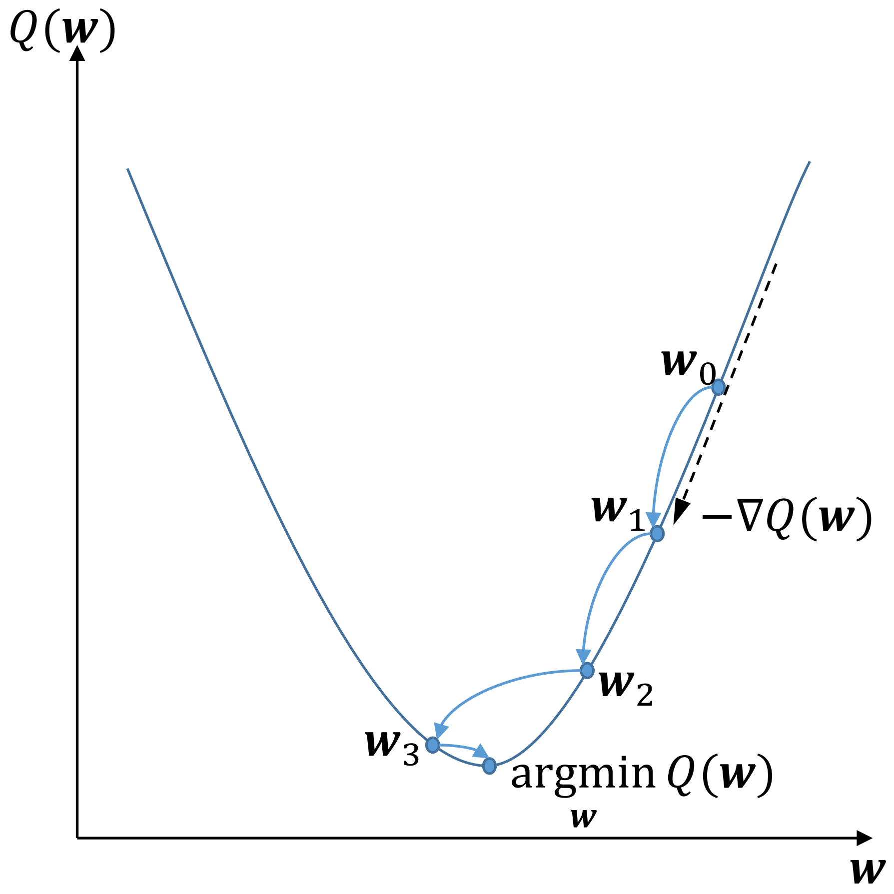


Минимизация эмпирического риска

- Функция потерь $L(y, y')$, описывающая отклонения ответа алгоритма $y' = f(x)$ от истинного ответа y
- Классификация — пороговая:
$$L(y, y') = I[y \neq y']$$
- Регрессия — L2:
$$L(y, y') = (y - y')^2$$
- Эмпирический риск:
$$Q(\boldsymbol{y}, \boldsymbol{x}, f) = E\big[L\big(y, f(x)\big)\big]$$
$$\boldsymbol{x} = \{x\}_{i=1}^{n}; \boldsymbol{y} = \{y\}_{i=1}^{n} \text{ — обучающий набор}$$
- Минимизация эмпирического риска:
$$f = \operatorname*{argmin}_{f \in F} Q(\boldsymbol{y}, \boldsymbol{x}, f)$$

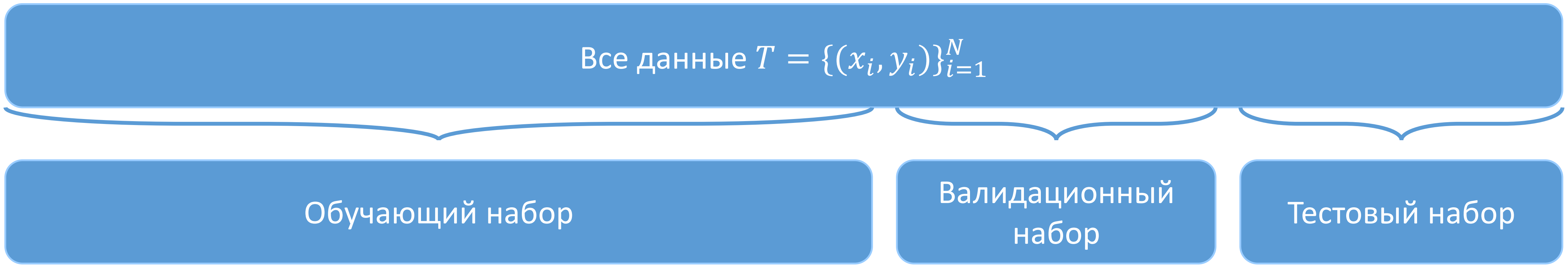
Градиентный спуск

- Итеративная оптимизация
- Правило обновления параметров алгоритма:
$$\boldsymbol{w_{i+1}} = \boldsymbol{w_i} - \eta \nabla Q(\boldsymbol{w_i})$$
- η — скорость обучения (learning rate)
- $\boldsymbol{w_0}$— начальное приближение
- Стохастический градиентный спуск (SGD) — параметры обновляются по **одному** примеру обучающей выборки (x_i, y_i)



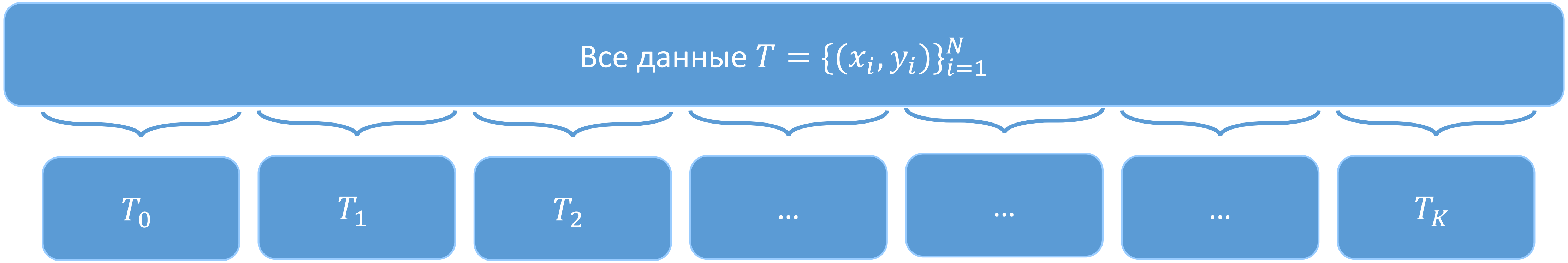
Контроль качества. Разбиение выборки.

- Обучающий набор — для подбора параметоров модели
- Валидационный набор — для подбора гиперпараметров модели, выбора из семейства моделей
- Тестовый набор — для финального замера качества модели

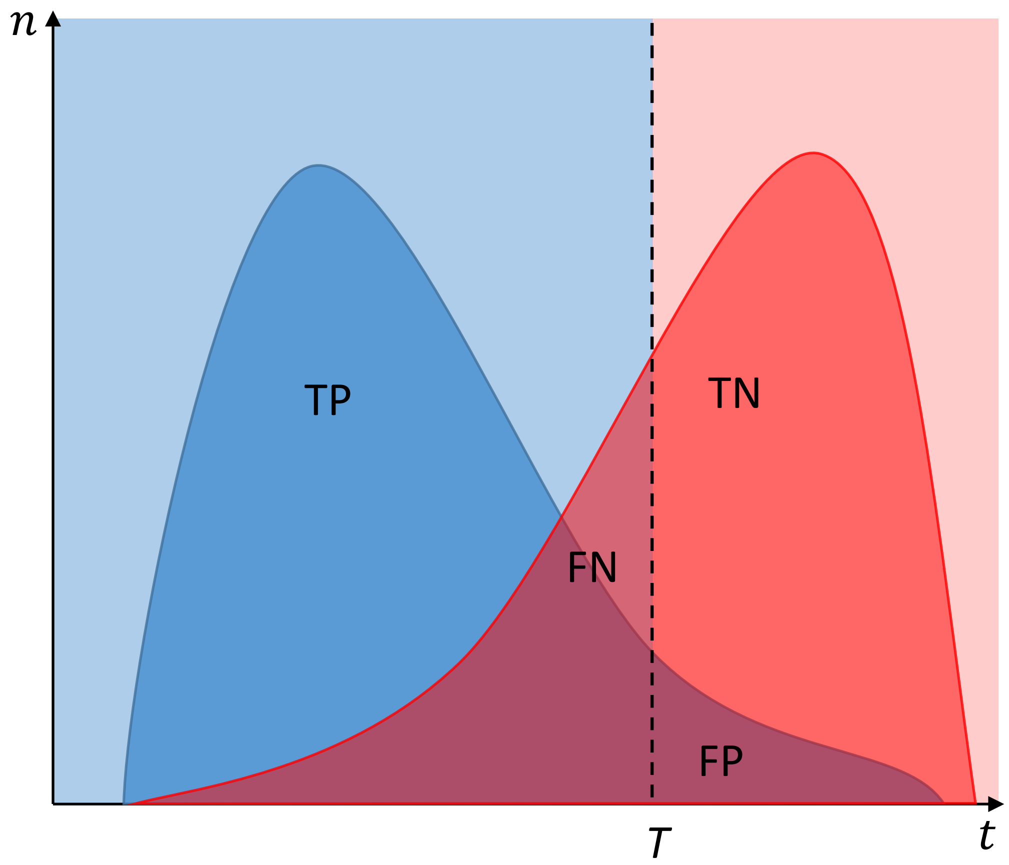


Кросс-валидация

- Обучаем K алгоритмов, выкидывая одну из частей выборки, на которой проверяем
- Финальное качество — объединение результатов по всем валидационным выборкам
- Финальный алгоритм — либо переучен на всей выборке, либо как ансамбль из всех K алгоритмов



Классификация. Ошибки 1-го и 2-го рода.



		Правильный класс (y_i)	
		Positive	Negative
Предсказанный класс ($\widehat{y_i}$)	Positive	True Positive	False Positive **(ошибка 1 рода)**
	Negative	False Negative **(ошибка 2 рода)**	True Negative

Многоклассовая классификация. Confusion matrix.

Правильный класс

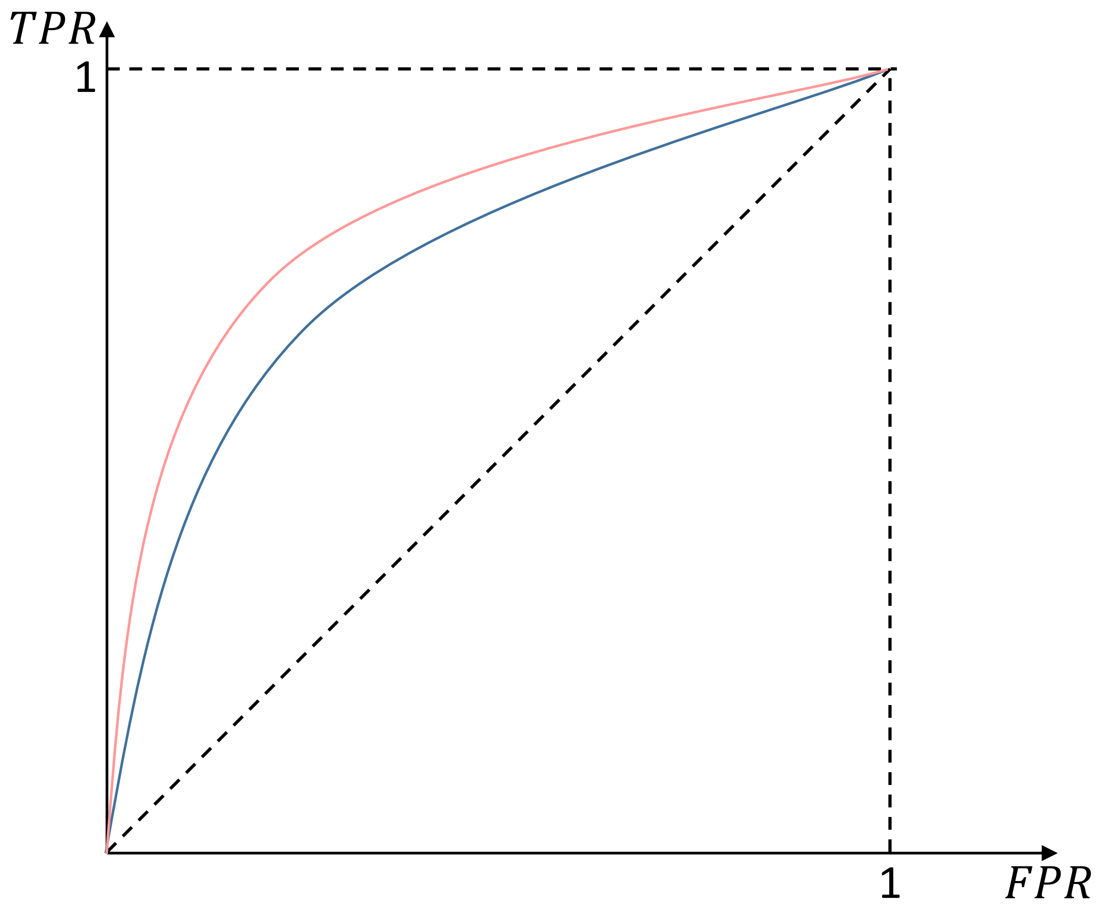
Предсказанный класс	0	1	2	3
0	8	1	5	2
1	0	3	2	4
2	7	1	6	0
3	6	4	1	9

Характеристики качества бинарного классификатора

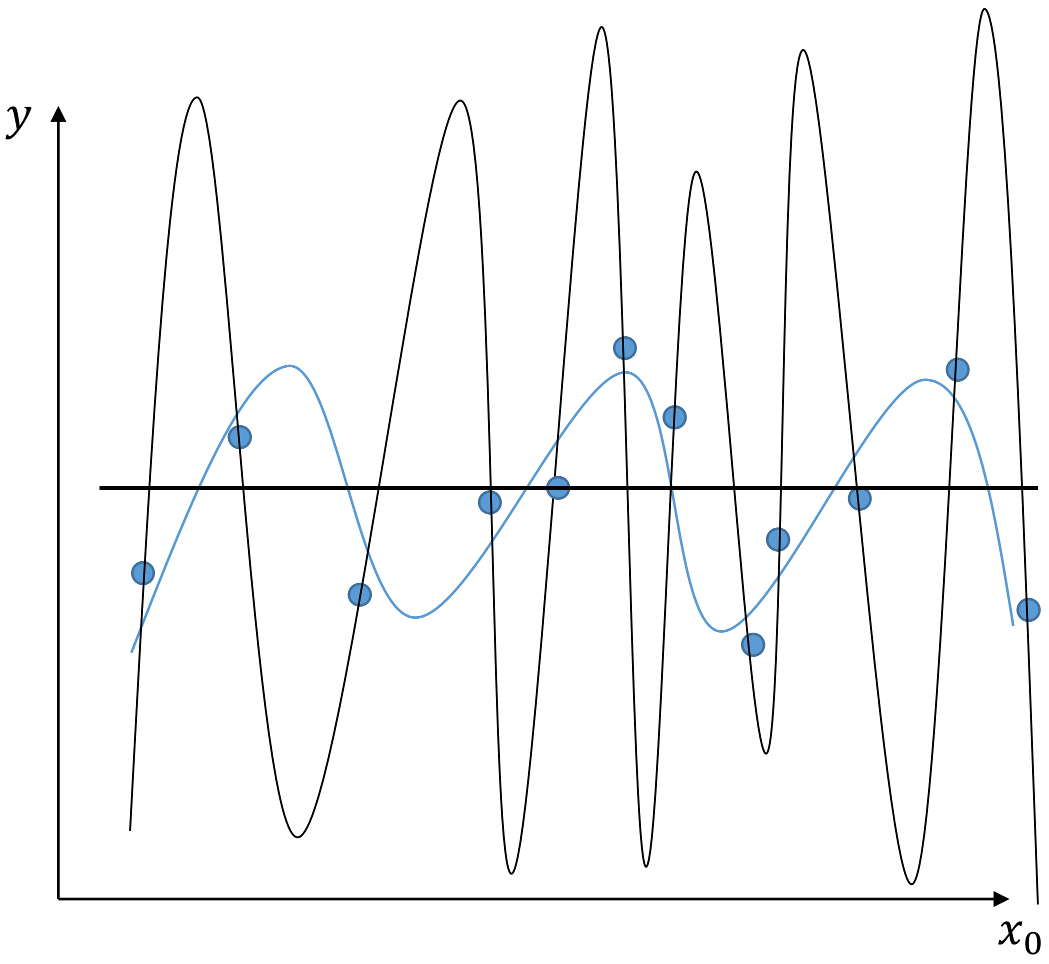
- False positive rate: $FPR = \frac{FP}{N} = \frac{FP}{FP+TN}$
- True positive rate: $TPR = \frac{TP}{P} = \frac{TP}{TP+FN} = Recall$
- False negative rate: $FNR = \frac{FN}{P} = \frac{FN}{TP+FN}$
- True negative rate: $TNR = \frac{TN}{N} = \frac{TN}{FP+TN} = Specificity$
- Positive Predictive Value: $PPV = \frac{TP}{TP+NP} = Precision$

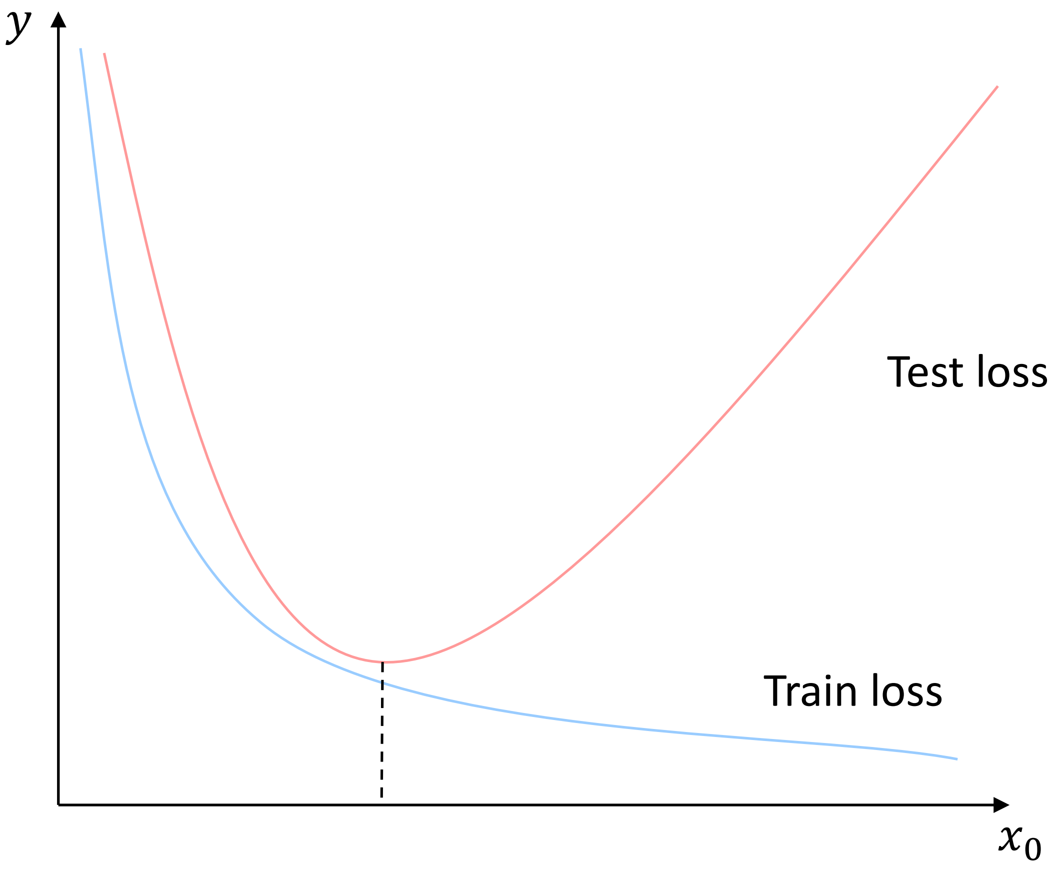
ROC-кривая

- Receiver Operating Characteristic
- График, характеризующий качество бинарного классификатора, в зависимости от порога T алгоритма
- Площадь под кривой (AUC) — часто используется для измерения качества классификаторов (questionable)

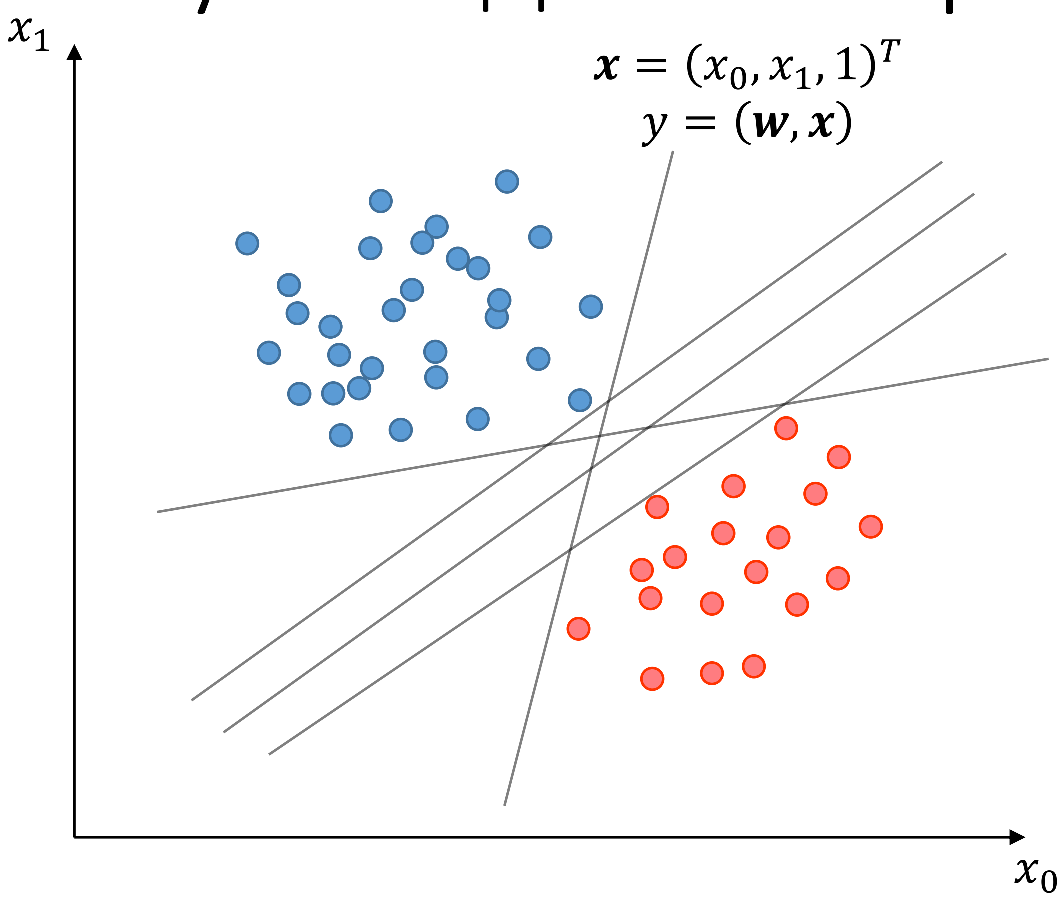


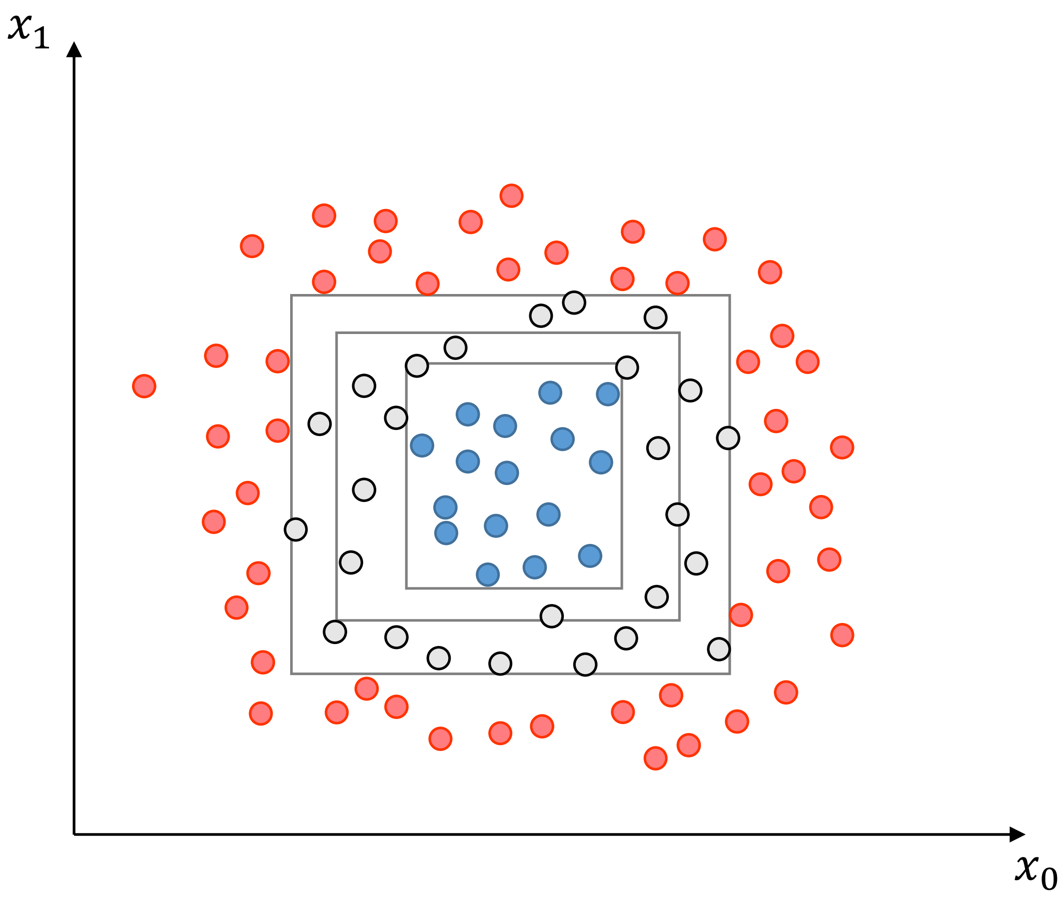
Переобучение





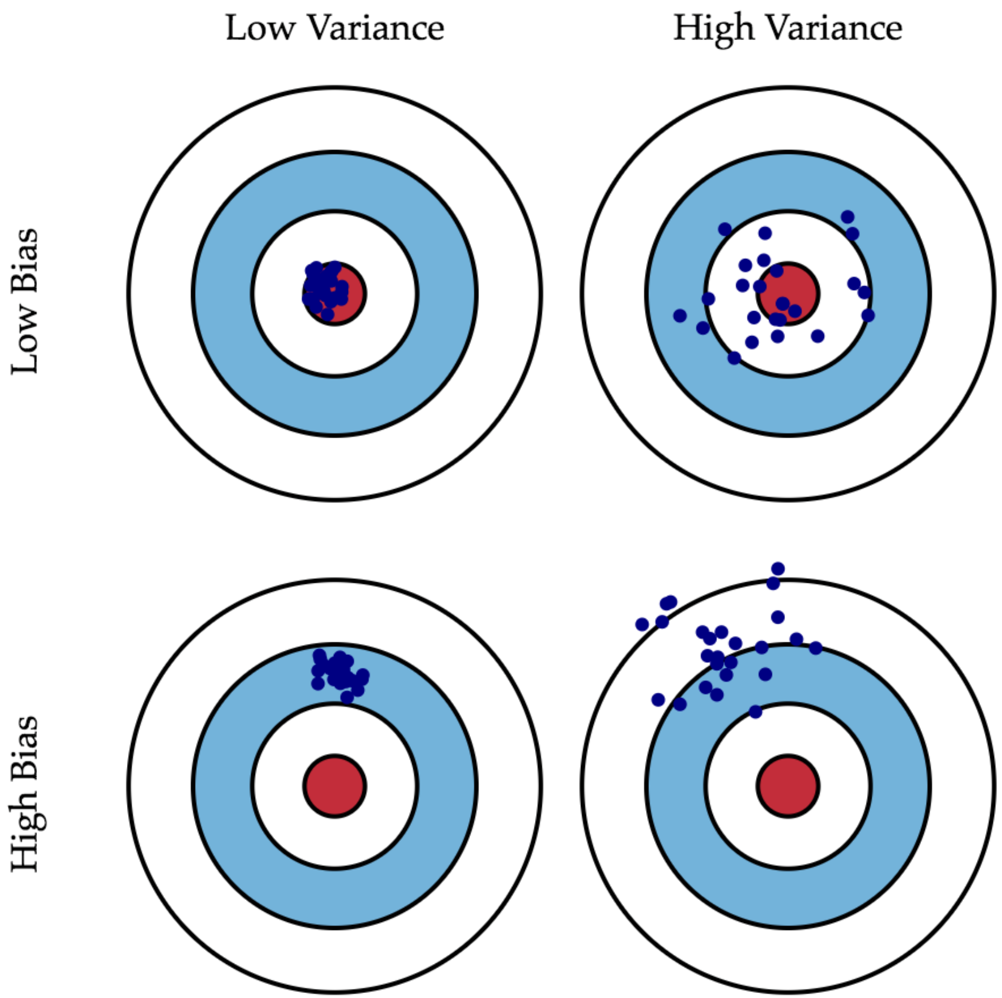
Какую модель выбрать?

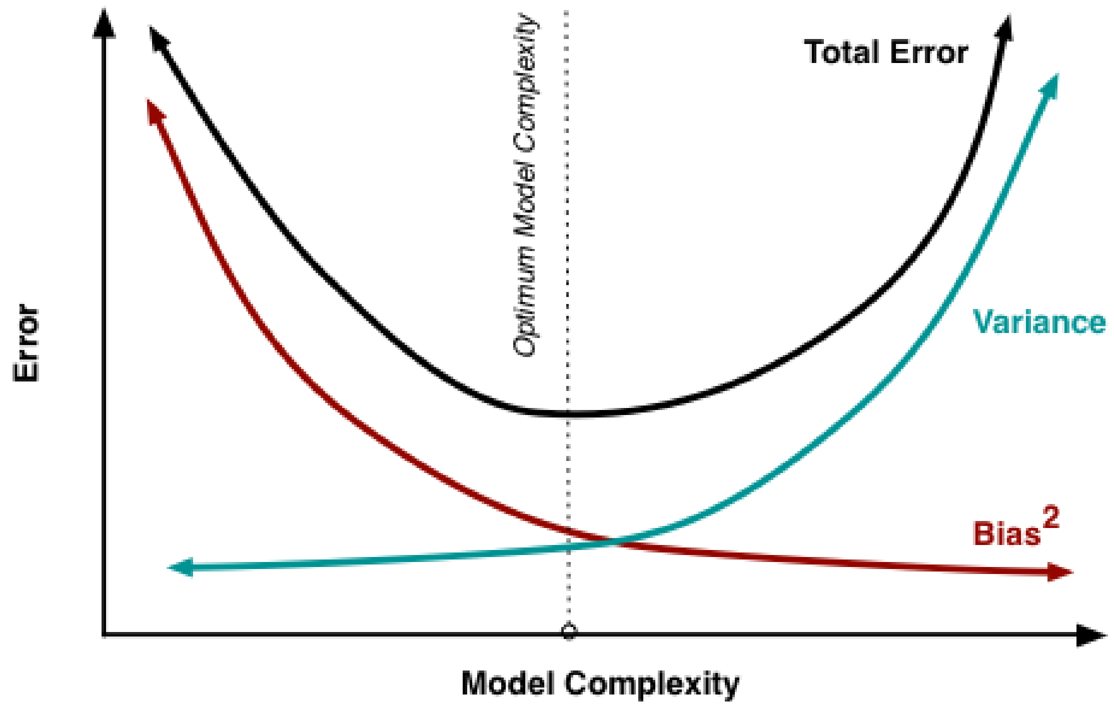




Общее правило – выбирать наиболее простую модель, обеспечивающую приемлемую ошибку

Разложение ошибки





Линейная регрессия

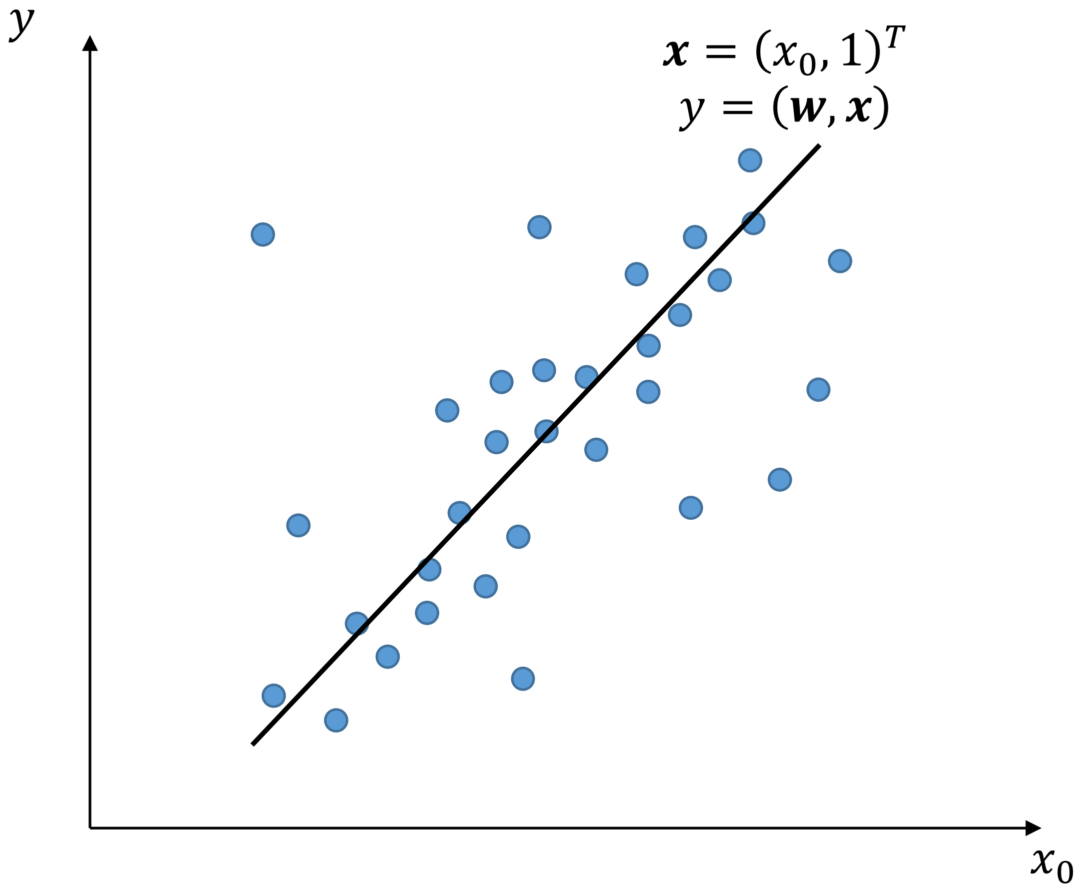
$$a(x) = (w, x)$$

- Выпуклая задача — единственное решение, может быть записано в явном виде:

$$X = (x_0, x_1, \dots, x_N)$$
$$\boldsymbol{y} = (y_0, y_1, \dots, y_N)$$
$$\boldsymbol{w} = (X^T X)^{-1} X^T y = X^+ \boldsymbol{y}$$

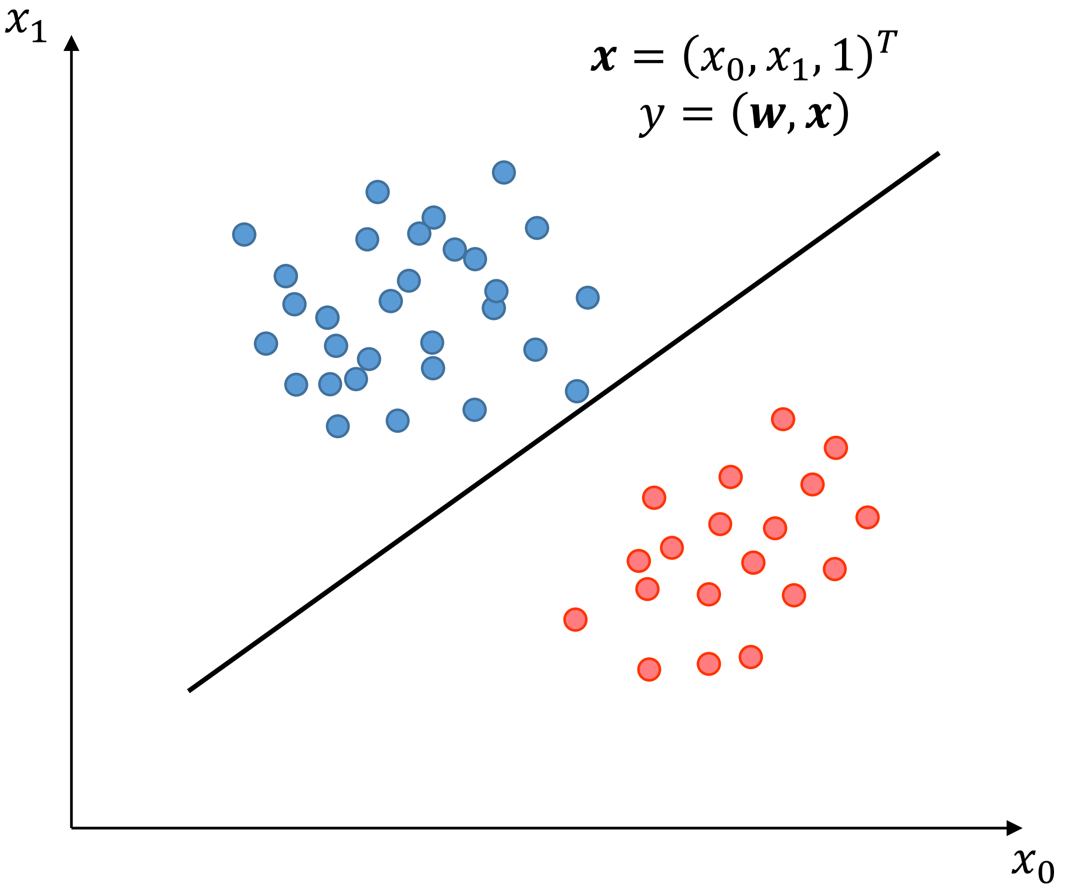
- X^+ — псевдообратная матрица
- Классификатор:

$$a(x) = \mathrm{sign}(w, x)$$



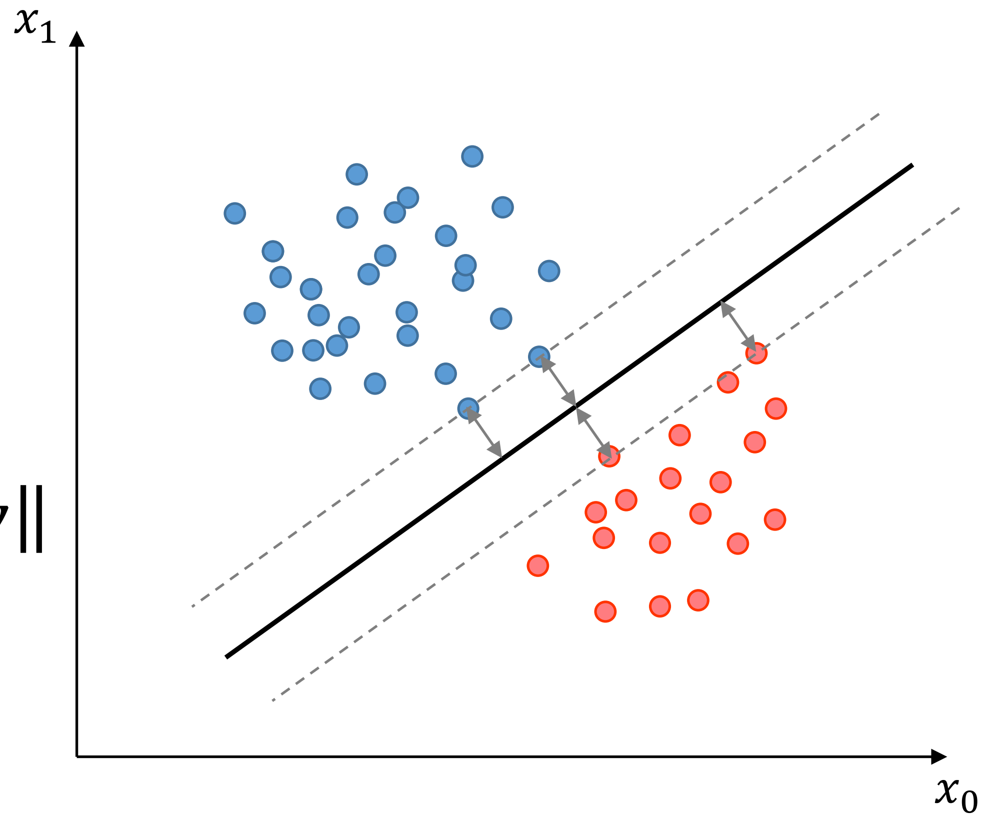
Логистическая регрессия

- Называется регрессией, но на самом деле —классификатор
- Оценивает вероятность:
$$P(y = 1|x) = \sigma((w, x)) = \frac{1}{1 + e^{-(w,x)}}$$
- σ — логистическая функция
- $\sigma'(z) = \sigma(z)\big(1 - \sigma(z)\big)$
- Функция потерь:
$$L(y, x, w) = -\log(\sigma\big((w, x)y\big))$$



Метод опорных векторов (Support Vector Machine)

- Максимизирует отступ между классами
- Расстояние до разделяющей гиперплоскости:
$$\frac{|(w, x) + b|}{\|w\|}$$
- Ширина разделяющей полосы: $2/\|w\|$



Линейно разделимый случай

- Максимизируем ширину, для этого решаем задачу

$$\begin{cases} \frac{1}{2} w^T w \to \min \\ y_i((w, x_i) + b) \geq 1 \end{cases}$$

- Задача квадратичного программирования, единственное решение, методом множителей Лагранжа

- Решение в виде:

$$w = \sum_i \alpha_i y_i x_i$$

- $\alpha_i = 0$ для всех не-опорных векторов

- Решающее правило:

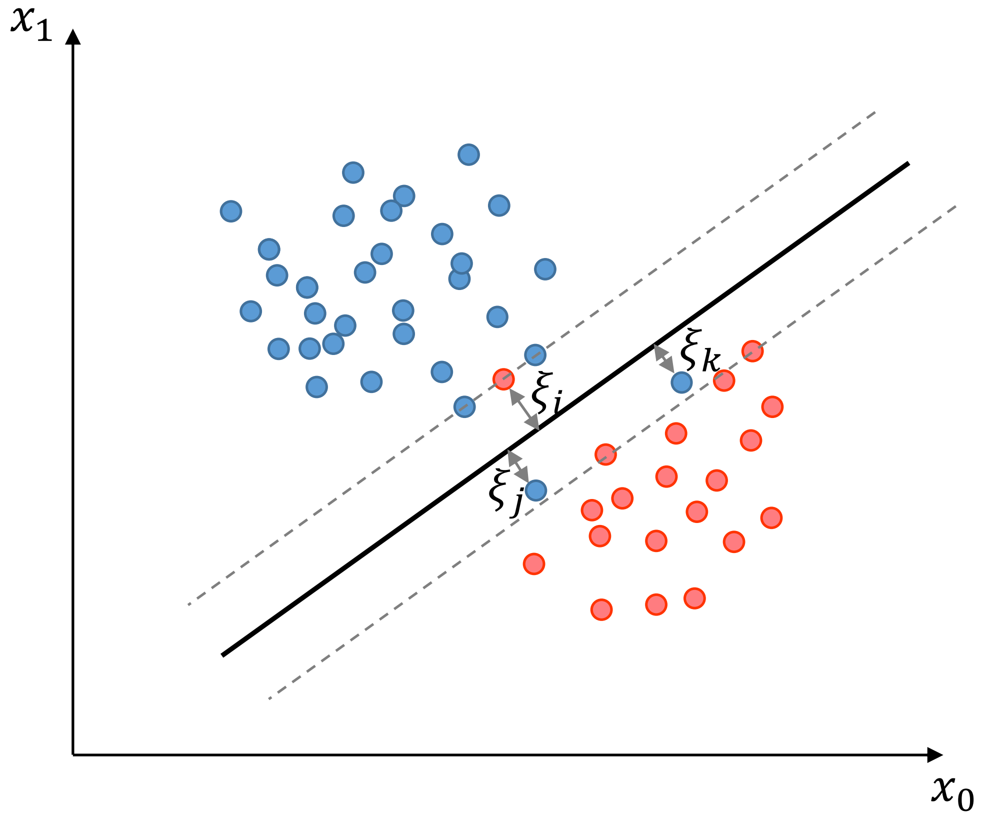
$$w \cdot x + b = \sum_i \alpha_i y_i x_i \cdot x + b$$

Линейно неразделимый случай

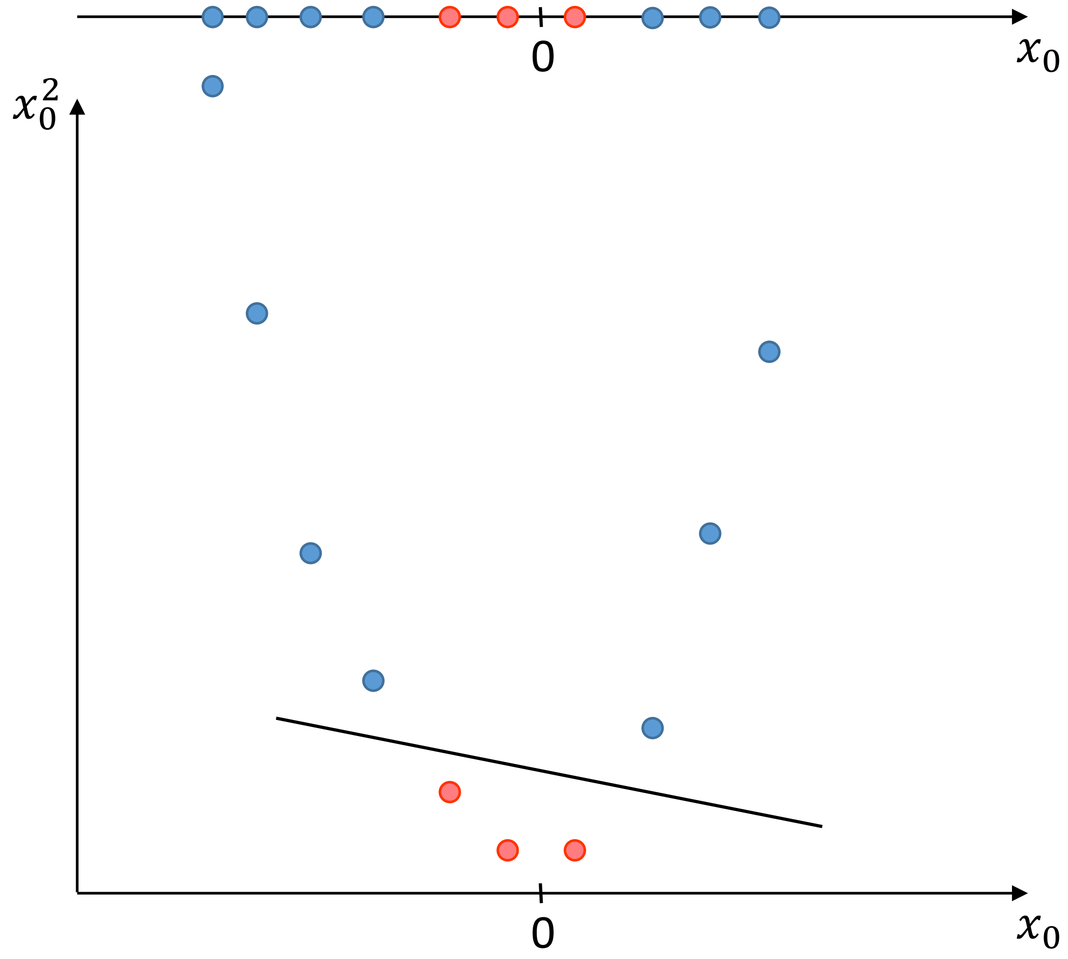
- Добавляем дополнительные переменные $\xi_i \geq 0$
- Модифицируем задачу:

$$\begin{cases} \frac{1}{2}w^T w + C\sum_i \xi_i \to \min \\ y_i\big((w, x_i) + b\big) \geq 1 - \xi_i \end{cases}$$

- C — параметр регуляризации



Kernel trick



- Ядро:

$$K(x_i, x_j) = (\varphi(x_i), \varphi(x_j))$$

- Условие Мерсера: матрица $K(x_i, x_j)$ должна быть симметричной и положительно определенной
- Решающее правило:

$$\sum_i \alpha_i y_i K(x_i, x) + b$$

- Пример — полиномиальное ядро:

$$K(x, y) = ((x, y) + c)^d$$

Если $d = 2, \mathrm{x} = (x_0, x_1)$:

$$\varphi(x) = (x_0^2, x_1^2, \sqrt{2}x_1 x_2, \sqrt{2c}x_1, \sqrt{2c}x_2, c)$$