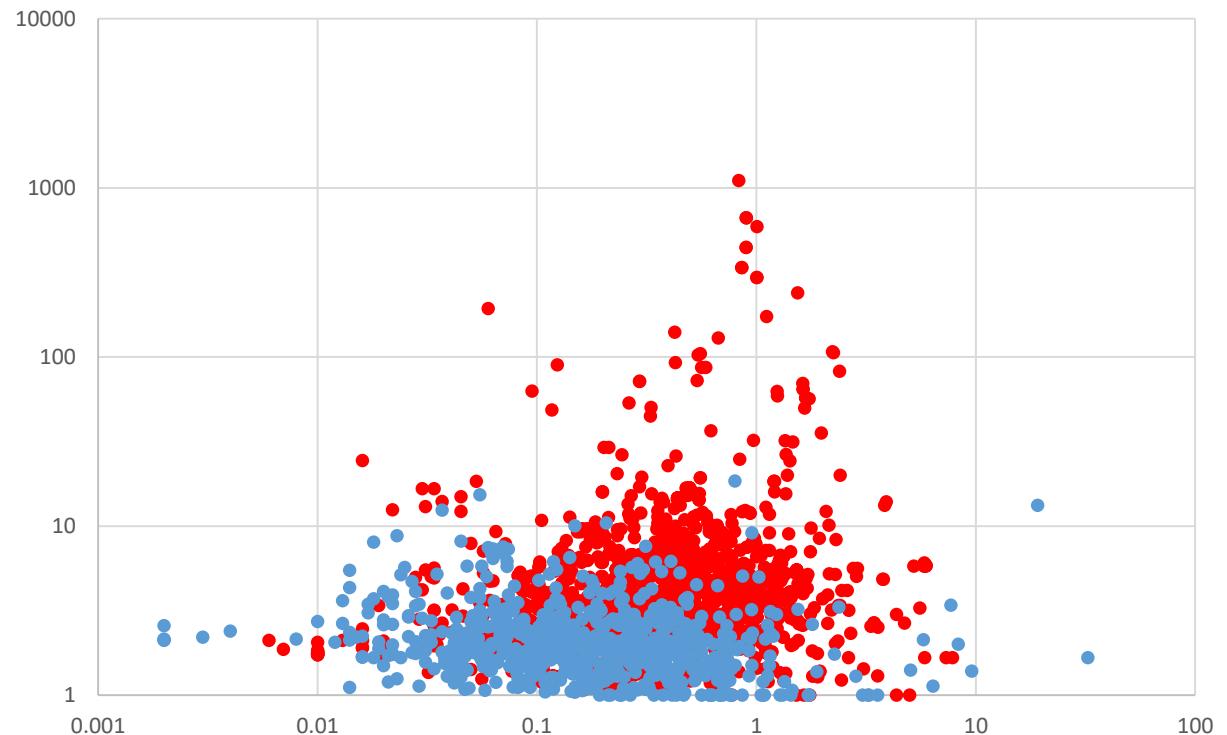


Задача классификации

- Ввод (Input): \mathbf{X}
- Вывод (Output): \mathbf{y}
- Целевая зависимость (Target function): $f: X \rightarrow Y$
- Данные (Data): $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$
- Гипотеза (Hypothesis): $h: X \rightarrow Y$

Y – множество классов



Гипотеза компактности

«Гипотеза компактности — в задачах классификации предположение о том, что схожие объекты гораздо чаще лежат в одном классе, чем в разных; или, другими словами, что классы образуют компактно локализованные подмножества в пространстве объектов.»

«В математическом анализе *компактными* называются ограниченные замкнутые множества. Гипотеза компактности не имеет ничего общего с этим понятием и должна пониматься в «более бытовом» смысле этого слова.»

Метрические классификаторы

Метрический классификатор Lazy learning

$$h(\mathbf{x}; D) = \arg \max_{y \in Y} \sum_{\mathbf{x}_i \in D} [y_i = y] w(\mathbf{x}_i, \mathbf{x})$$

$\Gamma_y(\mathbf{x})$

$w(\mathbf{x}_i, \mathbf{x})$ – вес соседа \mathbf{x}_i

$\Gamma_y(\mathbf{x})$ – близость \mathbf{x} к классу y

kNN – k ближайших соседей

$w(x_i, x) = 1$, если x_i – один из k ближайших соседей

$w(x_i, x) = 1$, если $\rho(x_i, x) < R$ (Radius Neighbors)

Оценка качества – “Leave One Out”:

$$\text{LOO}(k, D) = \sum_{x_i \in D} h_k(x_i; D \setminus x_i) \neq y_i$$

WkNN – k взвешенных ближайших соседей

Варианты w :

$$w_i = \frac{r - \rho(x, x_i)}{r}$$

$$w_i = q^{-\rho(x, x_i)}$$

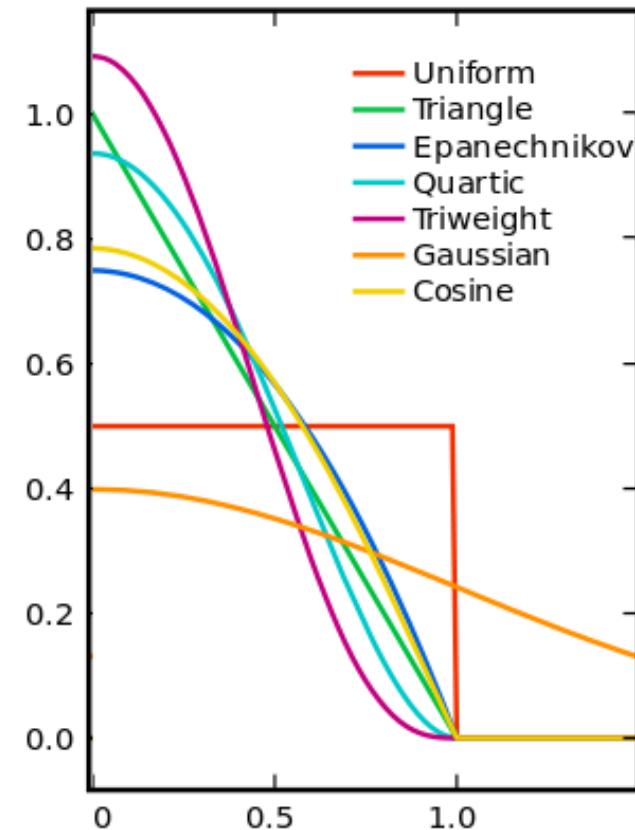
Метод окна Парзена (Parzen window):

$$w(x, x_i) = K \left(\frac{\rho(x, x_i)}{r} \right)$$

$$w(x, x_i) = K \left(\frac{\rho(x, x_i)}{\rho(x, x_j)} \right), \text{ где } x_j - (k+1)\text{-й сосед}$$

Фиксированная ширина

Переменная ширина



Метод потенциальных функций

$$h(\mathbf{x}, D) = \arg \max_{y \in Y} \sum_{\mathbf{x}_i \in D} [y_i = y] \gamma_i K\left(\frac{\rho(\mathbf{x}, \mathbf{x}_i)}{r_i}\right)$$

γ_i – веса объектов (заряд) Инициализация: $\gamma_i = 0$

r_i - радиус действия

Если $h(\mathbf{x}_i) \neq y_i \rightarrow \gamma_i = \gamma_i + 1$



В самом начале можно выбрать случайно или по наибольшему классу.

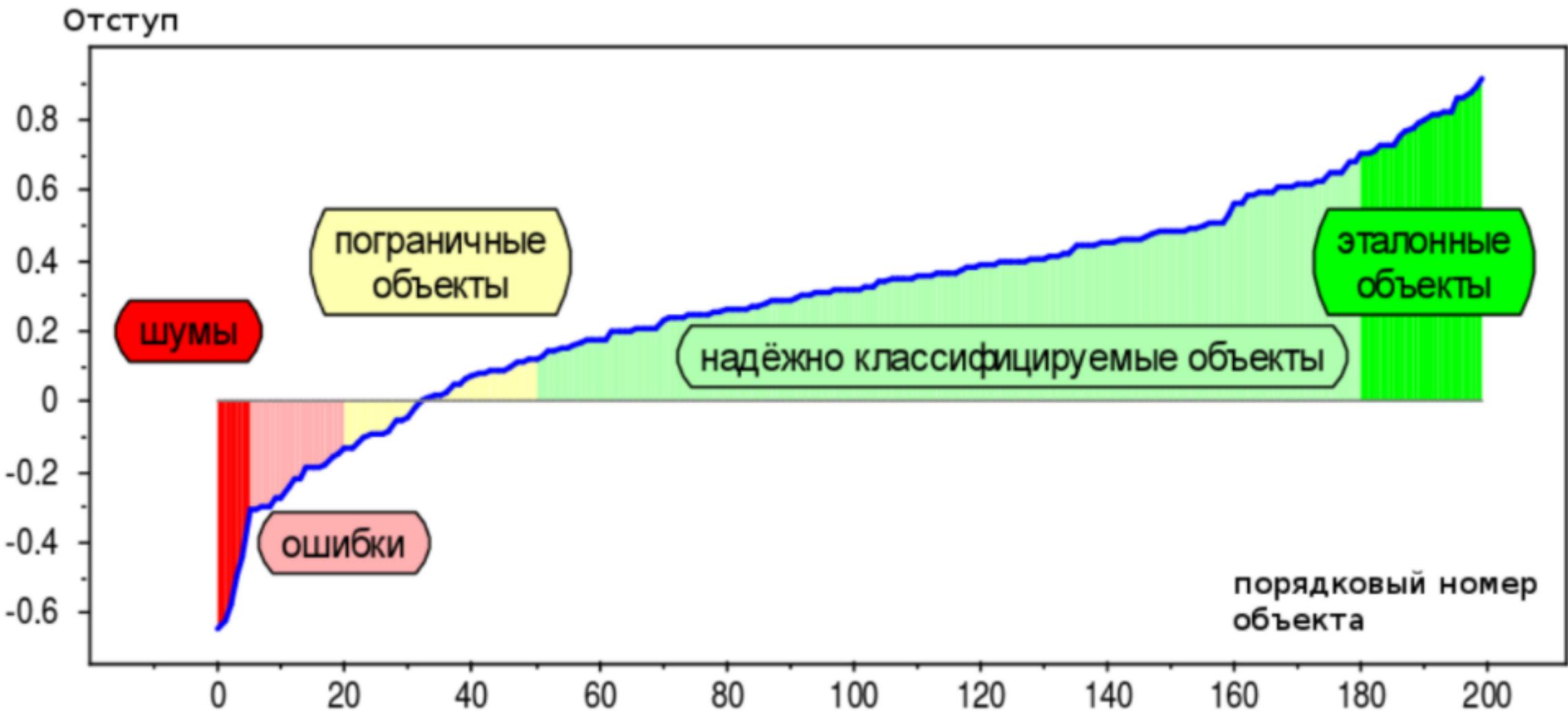
Prototype selection (Отбор эталонов)

$$f(\mathbf{x}): X \rightarrow Y$$

$$h(\mathbf{x}) = \arg \max_{y \in Y} \Gamma_y(\mathbf{x})$$

$$\text{Margin (отступ): } M(\mathbf{x}_i) = \Gamma_{y_i}(\mathbf{x}_i) - \max_{y \in Y \setminus y_i} \Gamma_y(\mathbf{x}_i)$$

Objects by margin



Prototype selection (Отбор эталонов)

$$h(\mathbf{x}; \Omega) = \arg \max_{y \in Y} \sum_{\mathbf{x}_i \in \Omega} [\mathbf{y}_i = y] w(\mathbf{x}_i, \mathbf{x})$$

Методы отбора эталонов

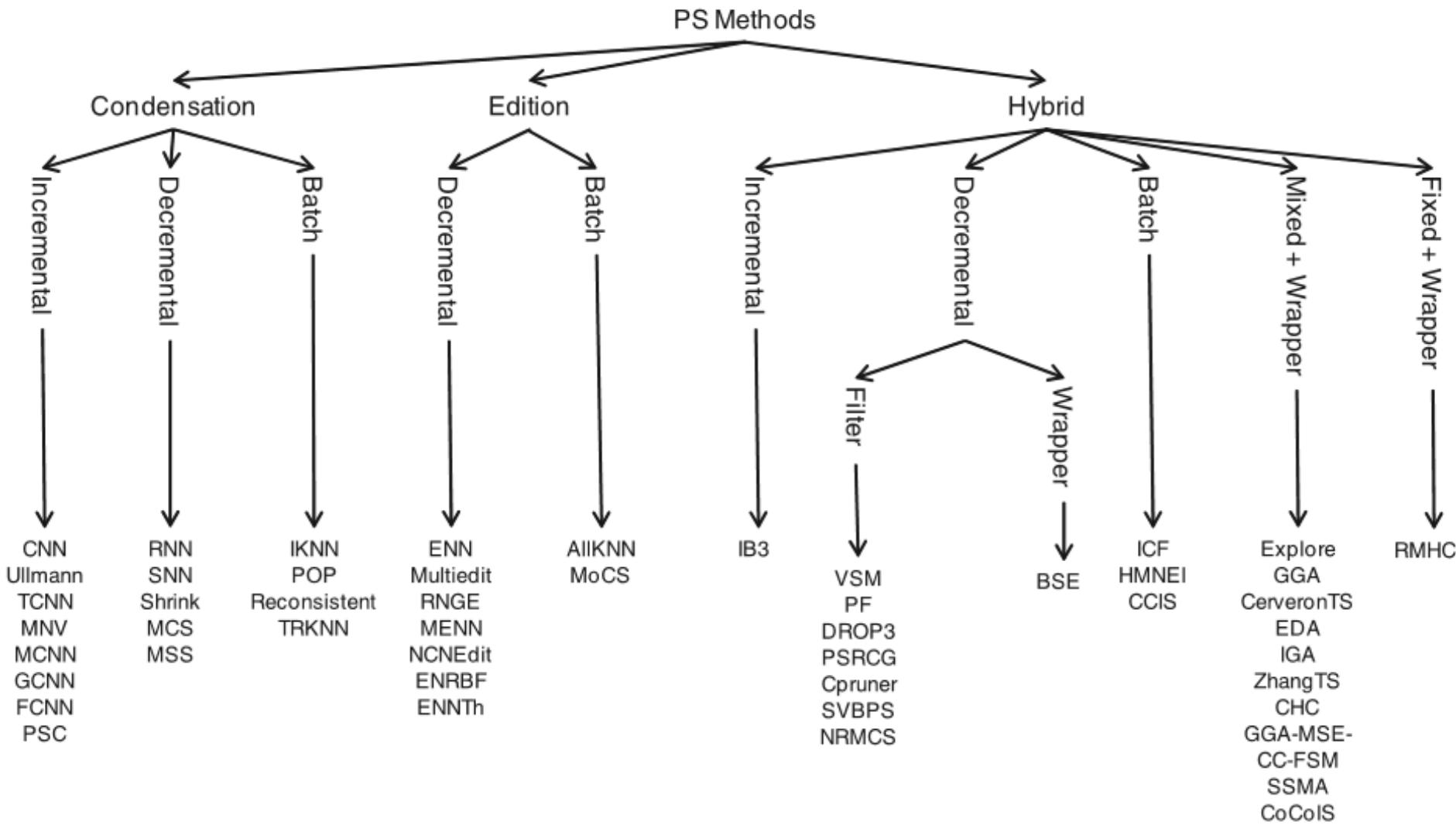
Направление поиска:

- Incremental
- Decremental
- Batch (Decremental)
- Mixed
- Fixed (Mixed)
- Replacement

Тип выбора:

- Condensation
- Edition
- Hybrid

Методы отбора эталонов



DROP5

(Decremental Reduction Optimization Procedure)

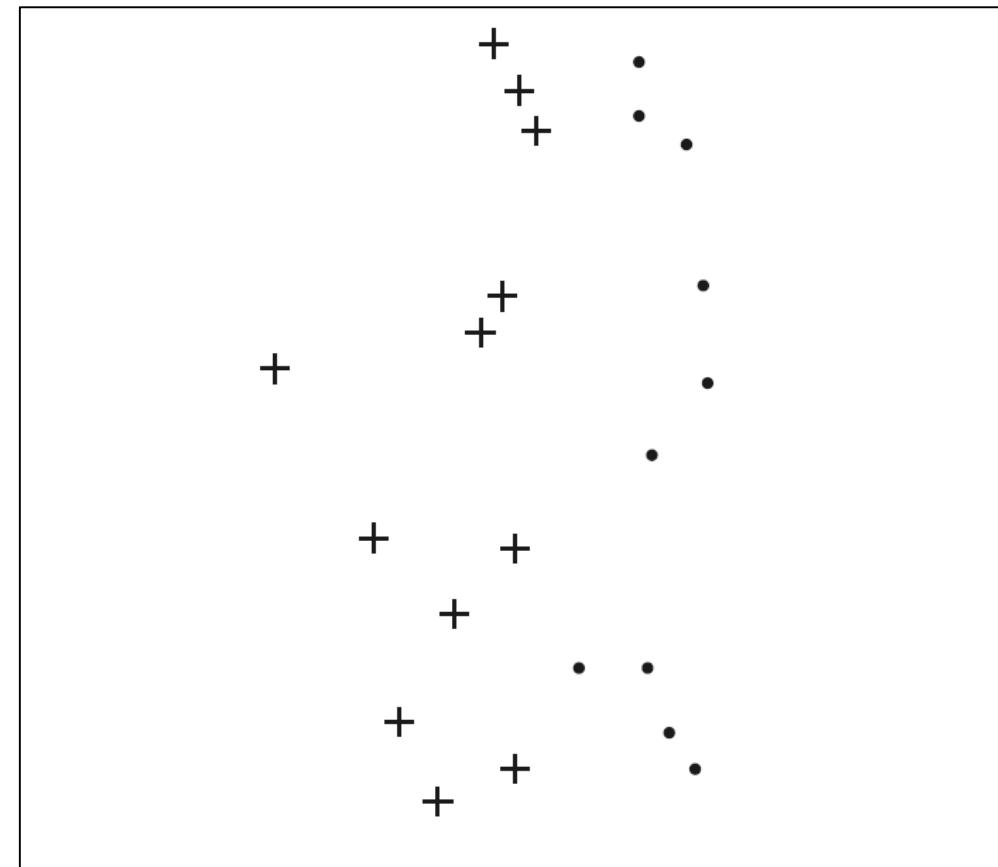
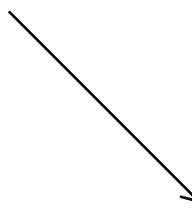
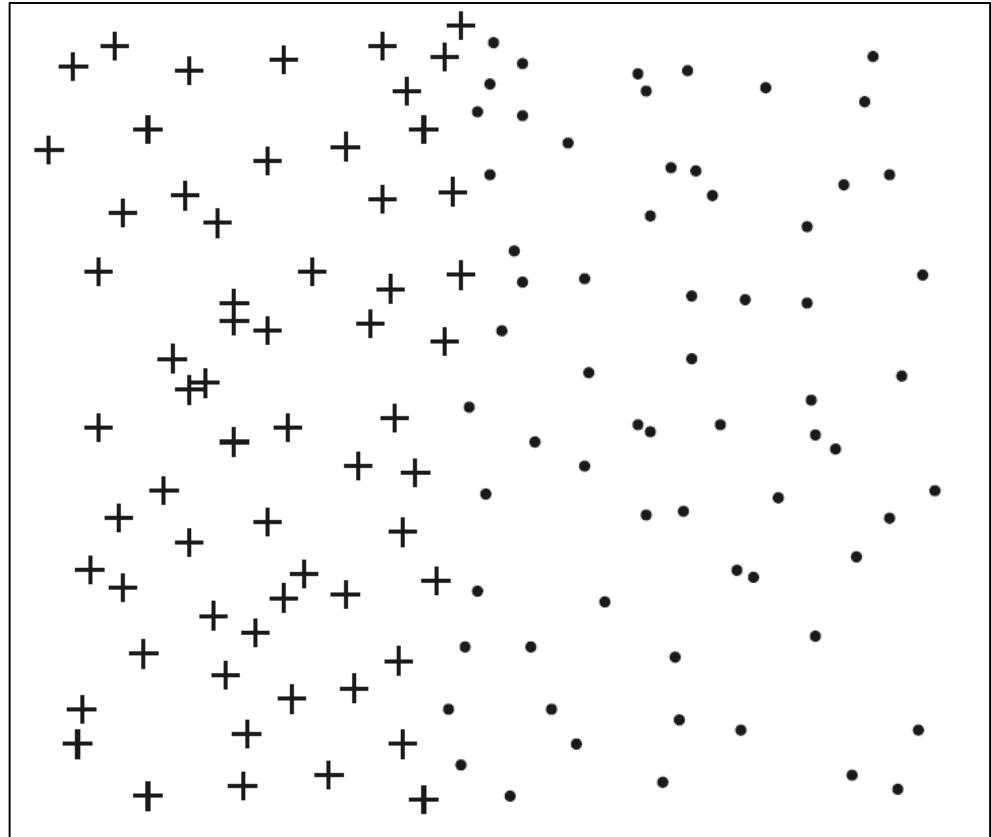
Начинаем с полного набора обучающих точек.

Отсортируем точки по расстоянию до ближайшей точки другого класса.

Идем от наименьшего расстояния к наибольшему (DROP5).

Удалить точку Р, если среди точек из полного набора, у которых Р была ближайшим соседом, правильно классифицированных останется столько же.

DROP5



Неравномерные признаки:

$$\rho(\mathbf{x}, \mathbf{x}_i) = \left(\sum_{j=1}^n w_j |x_j - x_{ij}|^p \right)^{\frac{1}{p}}$$

Метрика Минковского

Роль весов w :

1. Нормировка
2. Степень важности
3. Отбор

Добавление признаков

1. Выбираем один лучший признак k : $\rho_k(x_i, x_j) = |x_{jk} - x_{ik}|$
2. Есть расстояние ρ .
3. Добавляем признак k' :

$$\rho(x_i, x_j) = \rho(x_i, x_j) + w_{k'} \rho_{k'}(x_i, x_j)$$

4. Можно заменять признаки:

$$\rho(x_i, x_j) = \rho(x_i, x_j) - w_{k''} \rho_{k''}(x_i, x_j) + w_{k'} \rho_{k'}(x_i, x_j)$$

Будем добавлять, пока LOO уменьшается.

Быстрый поиск соседей k-d tree

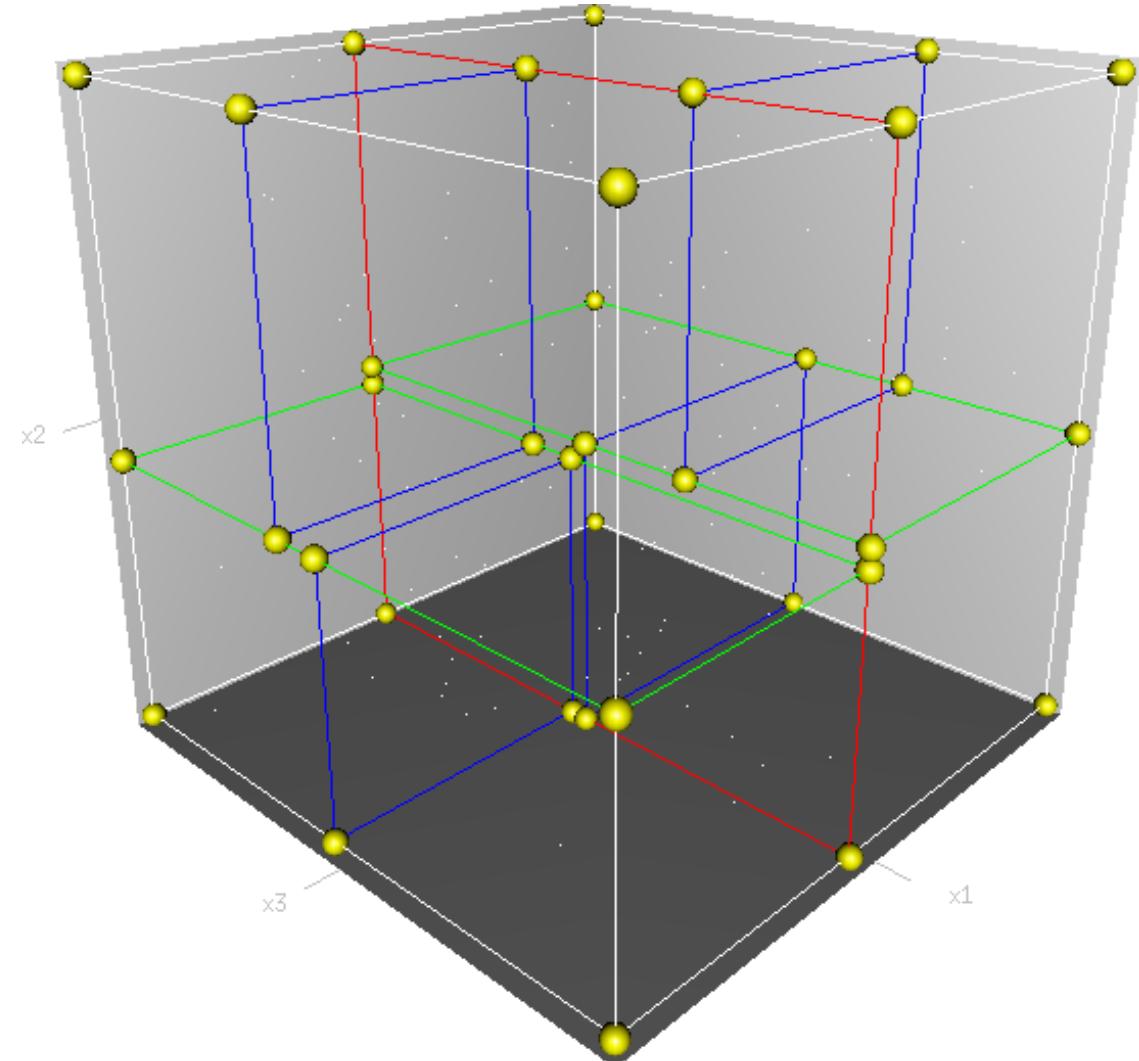
Разбиваем пространство гиперплоскостями, ортогональными одной из координатных осей, последовательно по медианным точкам.

Получаем участки с близким количеством точек в них (листы дерева).

Поиск соседей начинаем с точек листа, в котором находится точка, если соседей недостаточно – поднимаемся на узел выше.

Возможны ошибки, но их можно избежать, если следить, насколько близко точка лежит к границе листа.

На картинке слева – дерево глубины 3, делящее пространство на 8 листов (сначала по красной гиперплоскости, потом по зеленым гиперплоскостям, потом по синим).



Домашнее задание

Мягкий дедлайн (на полный балл) – в следующий четверг,

Жесткий дедлайн (на половинный балл) – еще через неделю.

Задачи можно решать на любом языке в любом окружении, но ОЧЕНЬ РЕКОМЕНДУЕТСЯ использовать Python и Jupyter Notebook.

На паре после лекции можно будет сдавать домашнее задание и задавать вопросы, если что-то не получается.

Ближайший мягкий дедлайн – 15 марта, жесткий – 22 марта.

Также есть дополнительные задачи на дополнительные баллы и более поздний дедлайн:

- Для того, чтобы сдавать дополнительные задания, нужно сделать все основные
- Больше 120-ти баллов набрать нельзя
- Для получения зачета нужно набрать половину баллов основных задач каждой домашки (можно за счет дополнительных :))

Кластеризация

Цели кластеризации

- Выделение закономерностей
- Построение иерархии множества объектов
- Упрощение дальнейшей обработки данных
- Сокращение объема данных
- Выделение нетипичных объектов
- Получение новых признаков

Кластеризация – обучение без учителя

- Нет точной постановки задачи
- Число кластеров?
- Критерий качества?

K-Means

Количество кластеров – задается.

μ_i – центр кластера C_i

Задача – минимизация:

$$\sum_{x_j} \min_{\mu_i} \|x_j - \mu_i\|_2^2$$



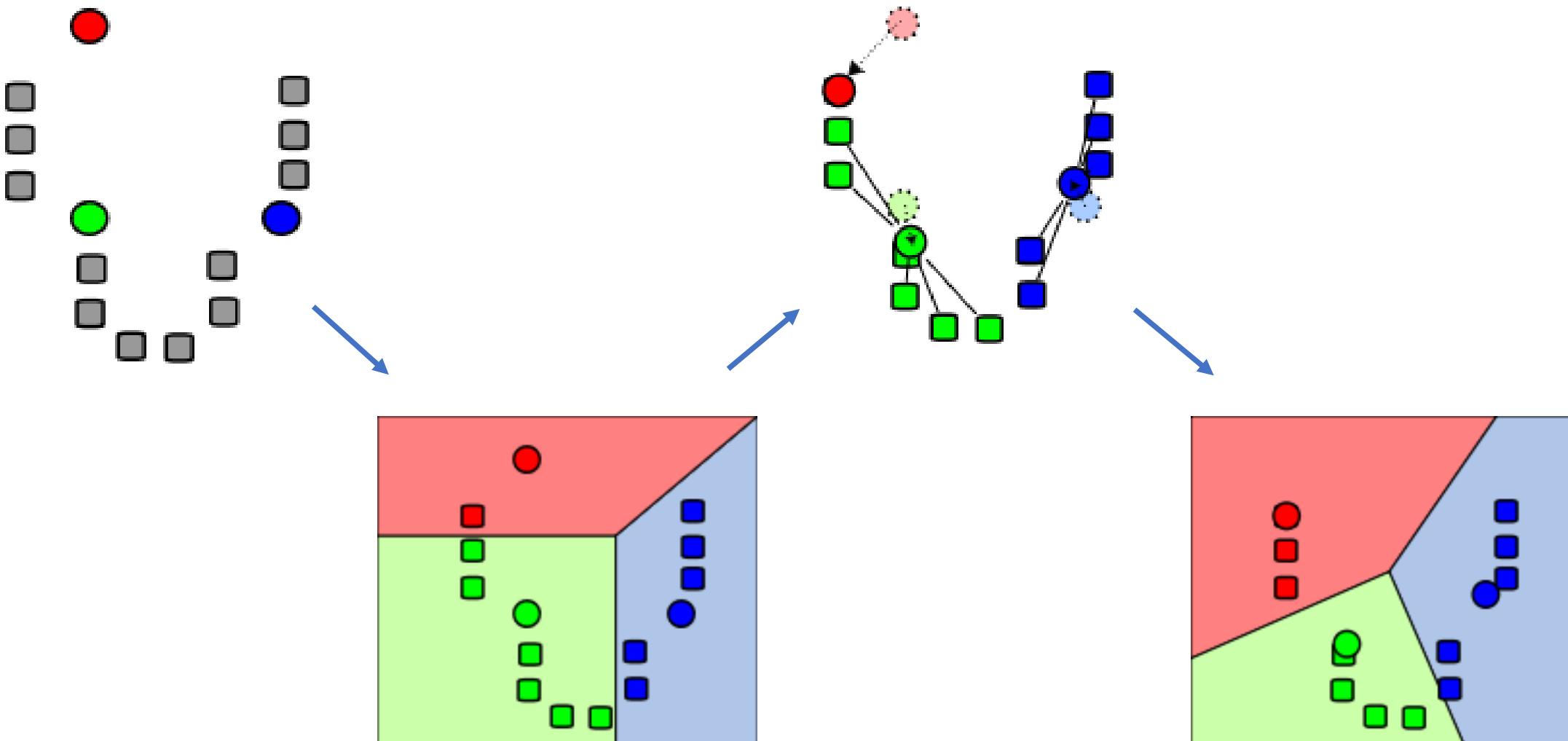
Алгоритм K-Means

1. Инициализируем центры кластеров (случайно или более хитрым образом).
2. Припишем каждую точку к ближайшему центру.
3. Переместим центры кластеров в «центр масс» кластеров:

$$\mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j$$

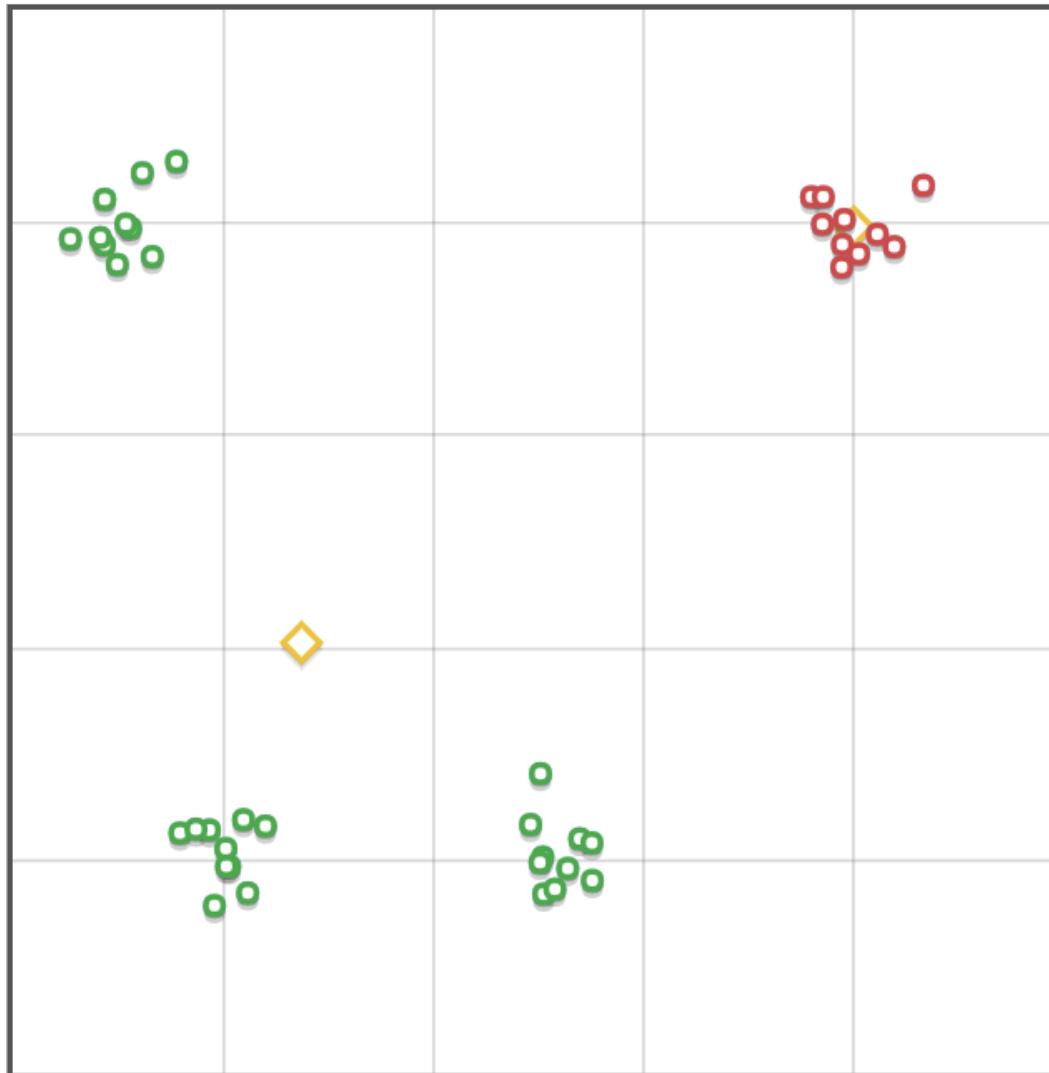
4. Повторяем шаги 2-3 до сходения.

K-Means



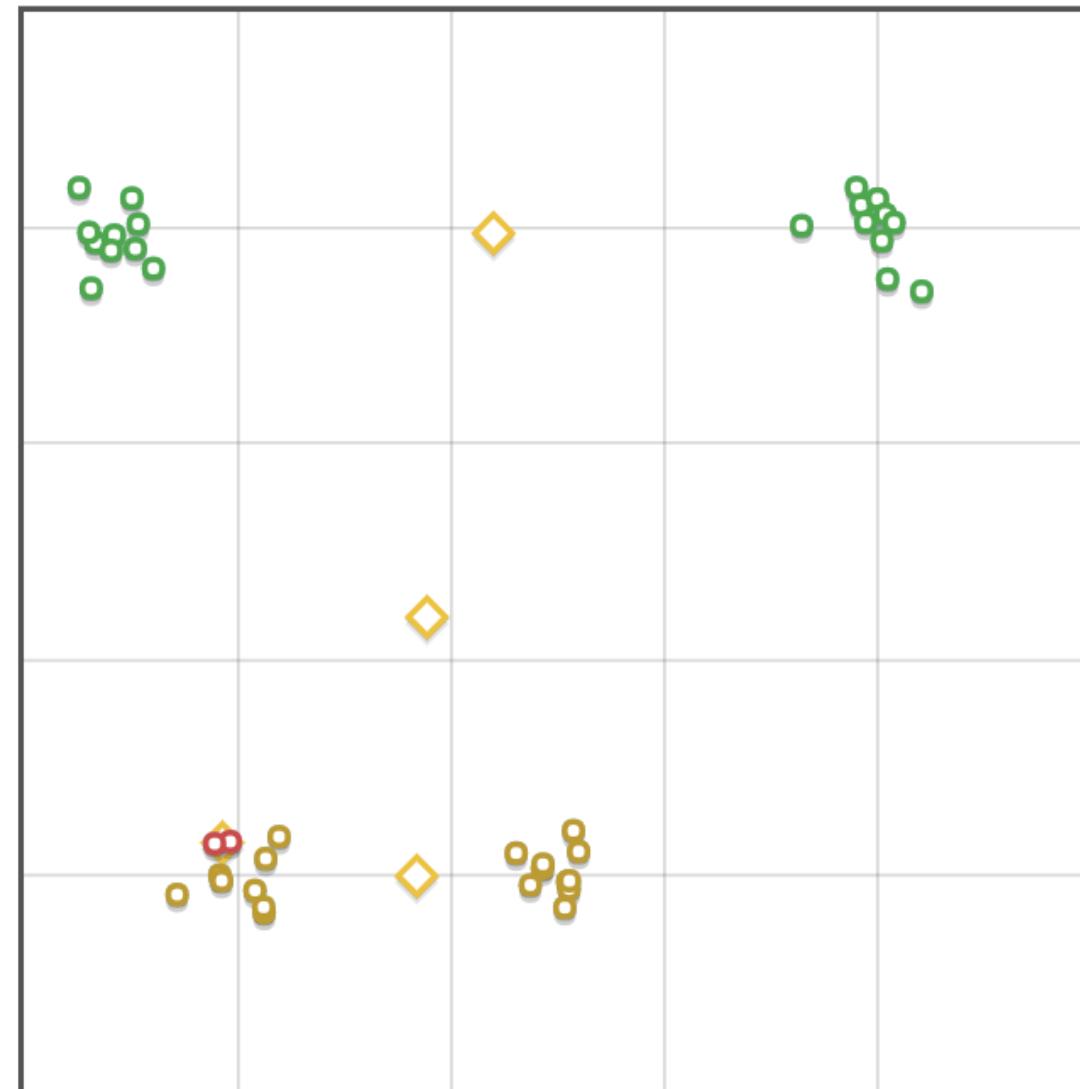
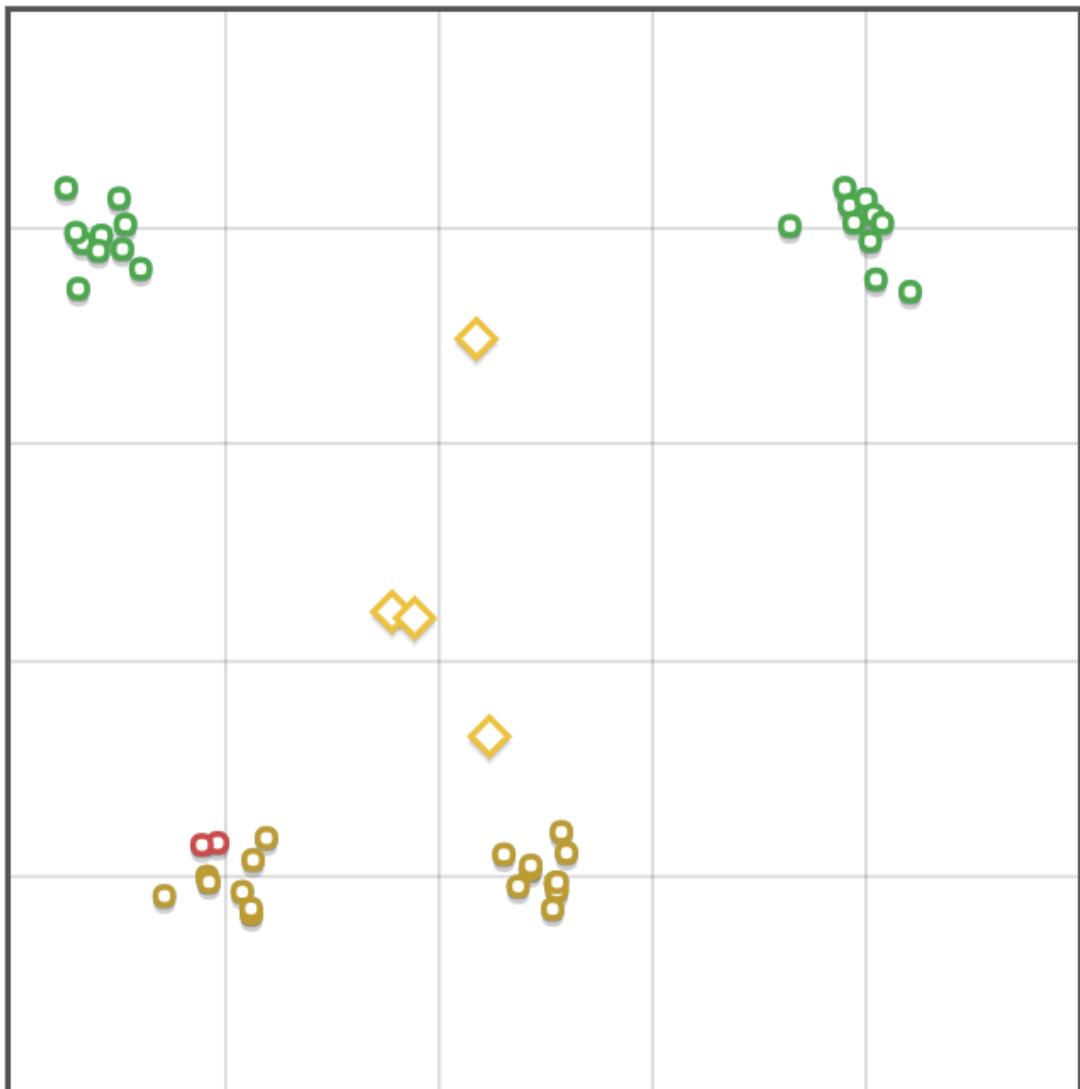
Проблемы K-Means

Выбор количества кластеров



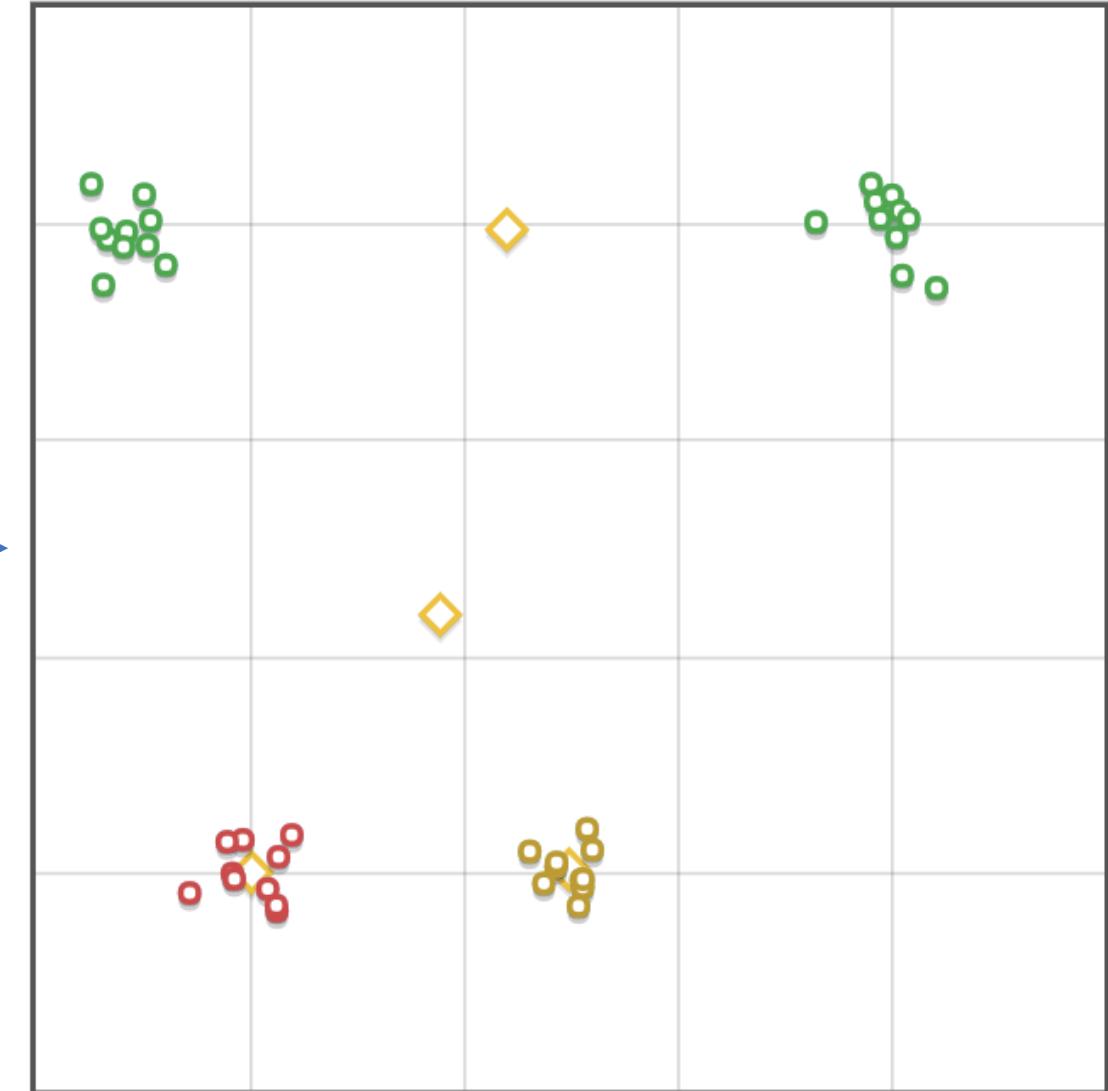
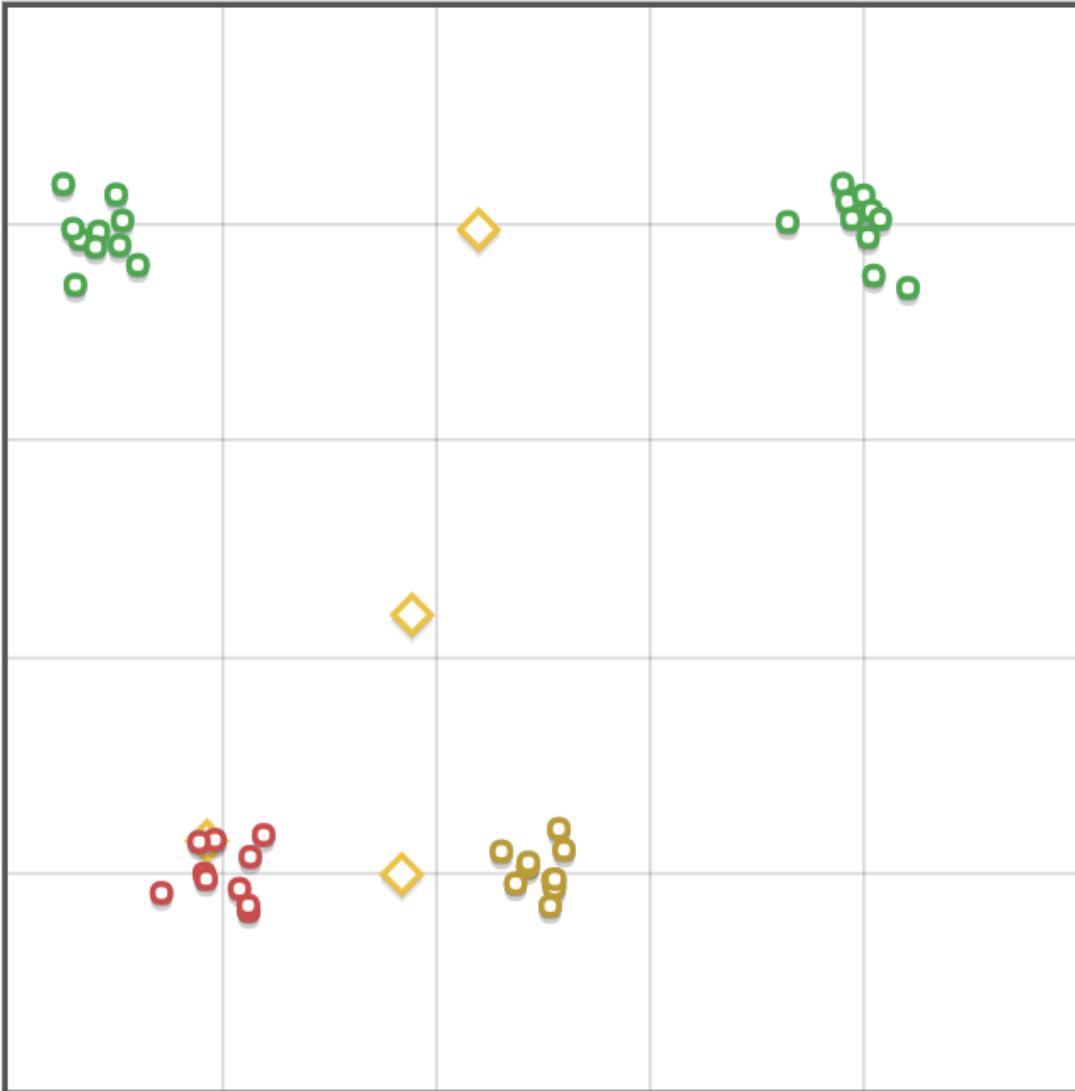
Проблемы K-Means

Выбор начального положения центров



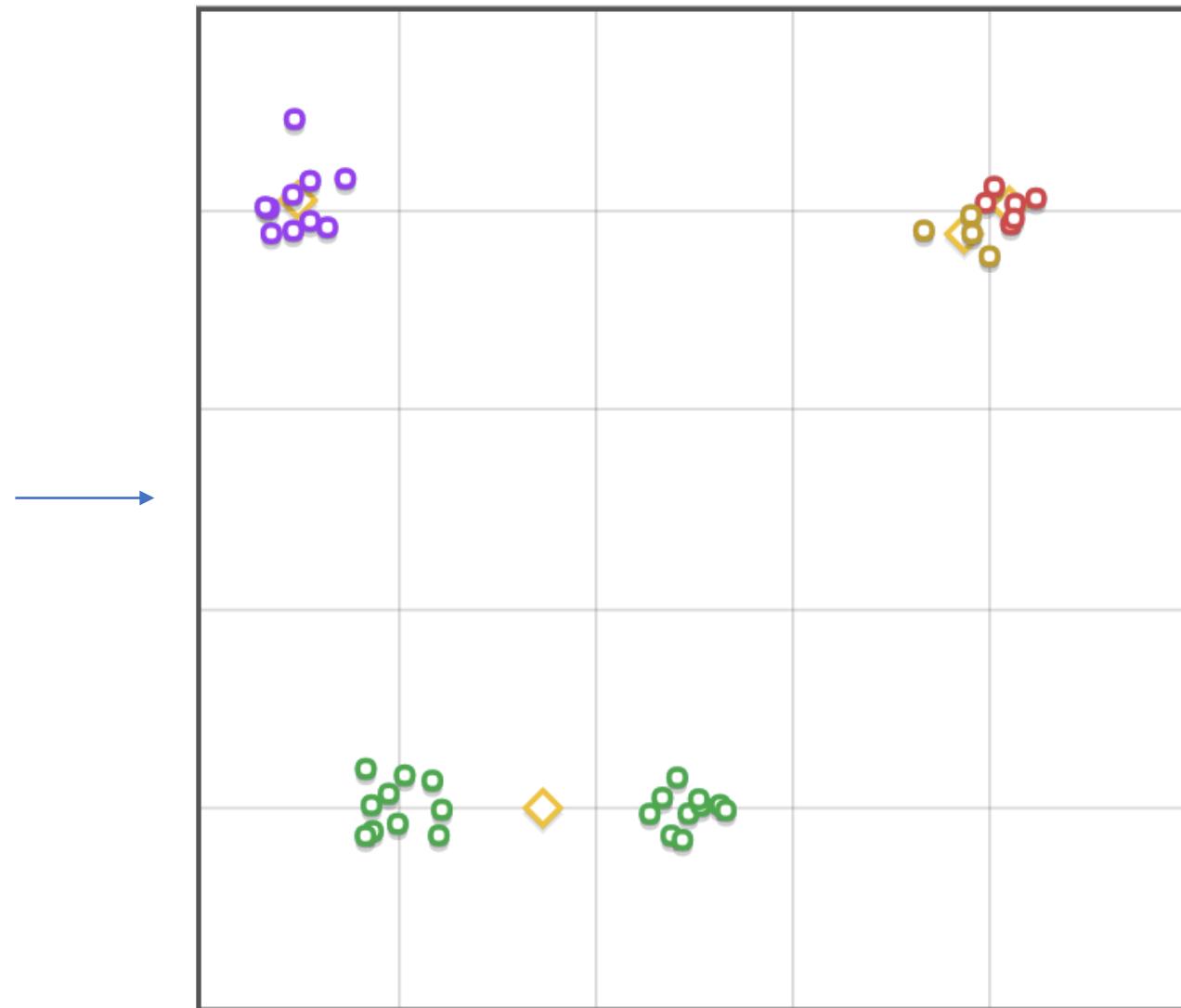
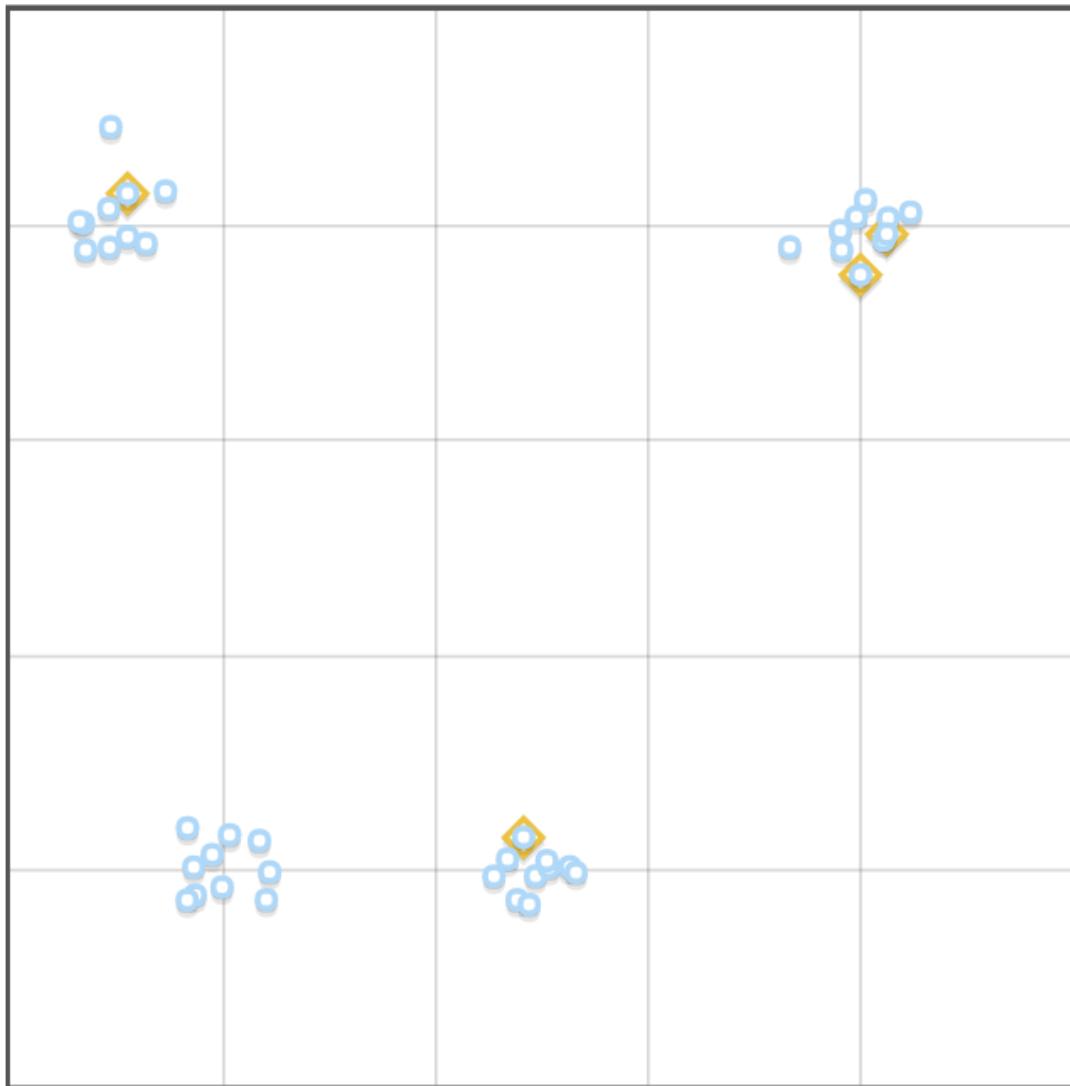
Проблемы K-Means

Выбор начального положения центров



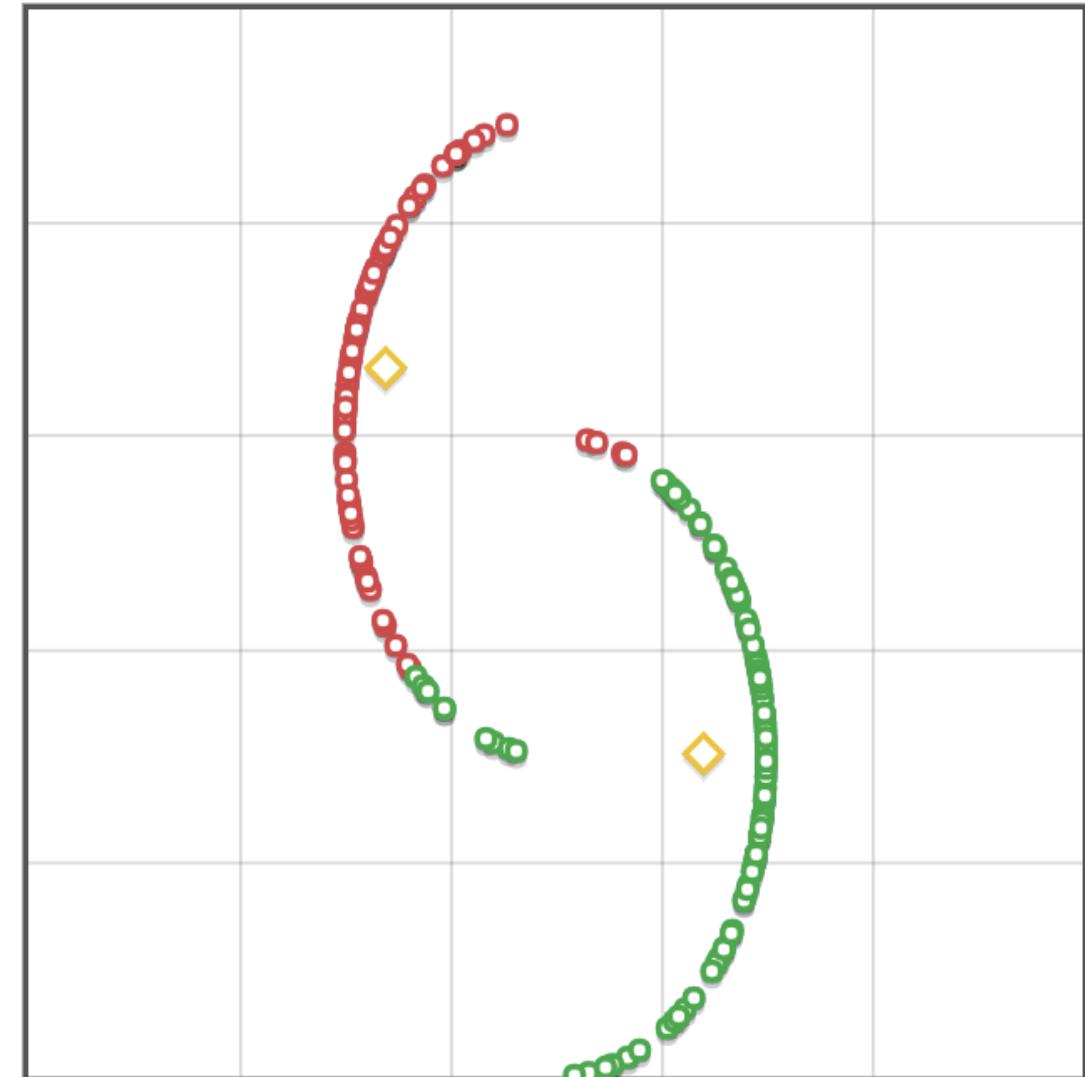
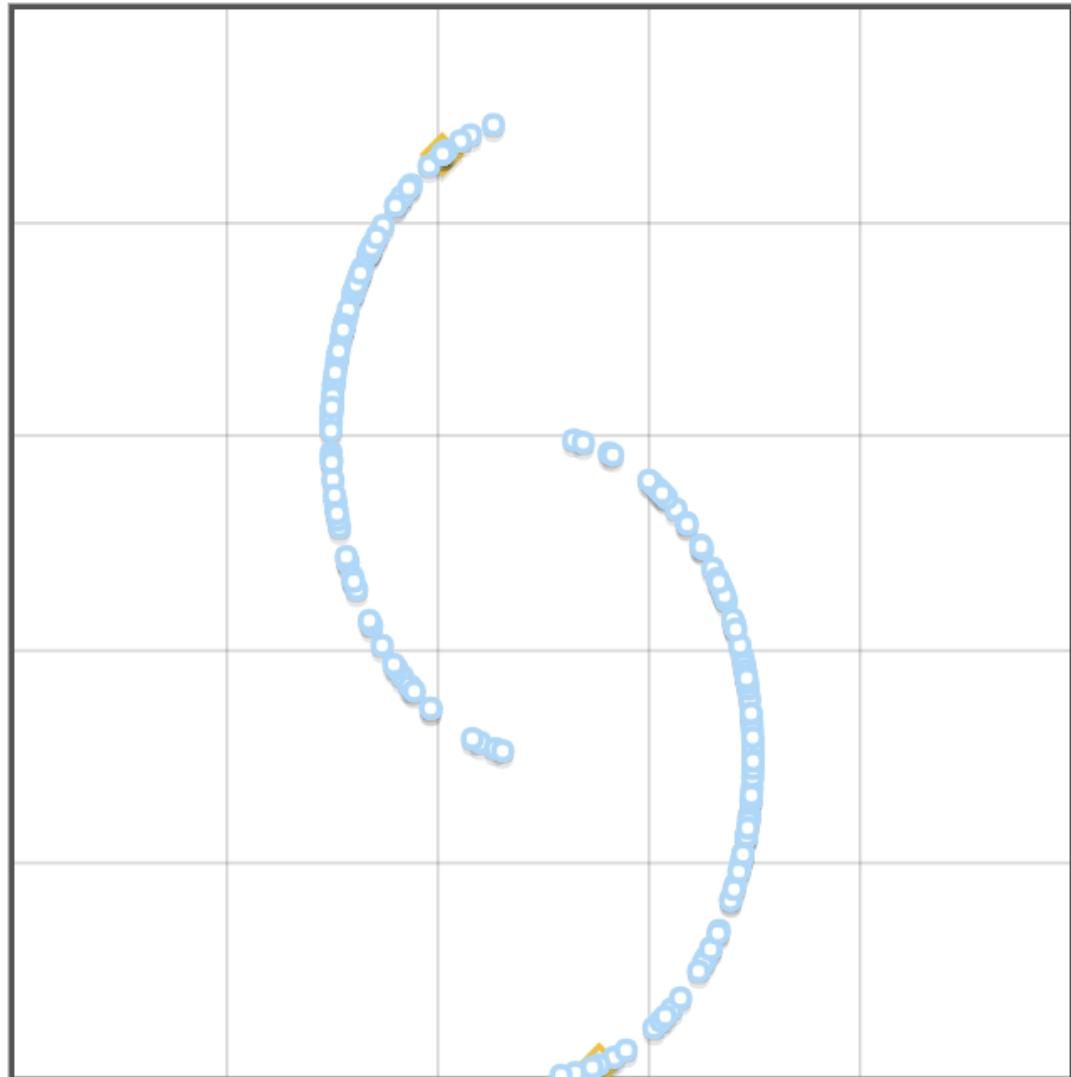
Проблемы K-Means

Выбор начального положения центров



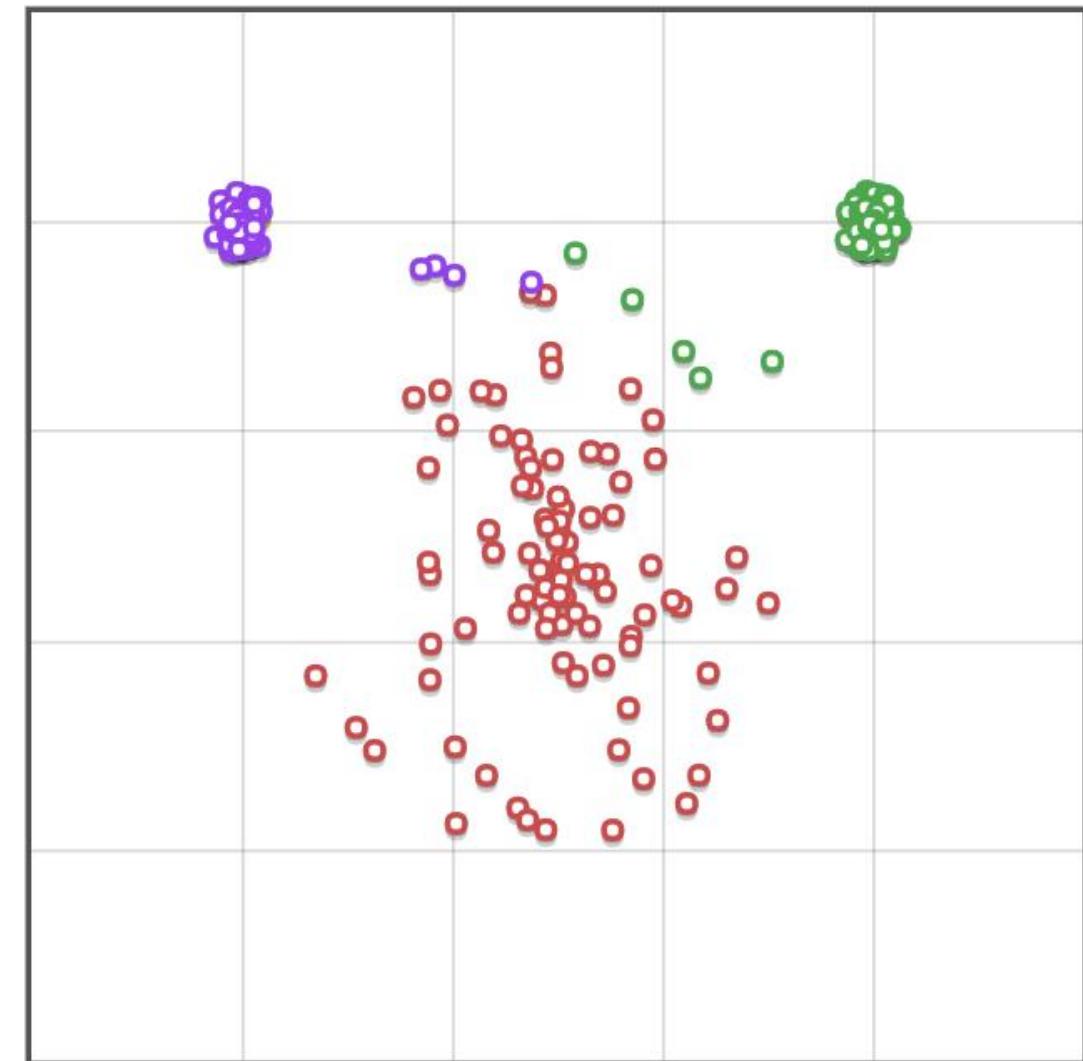
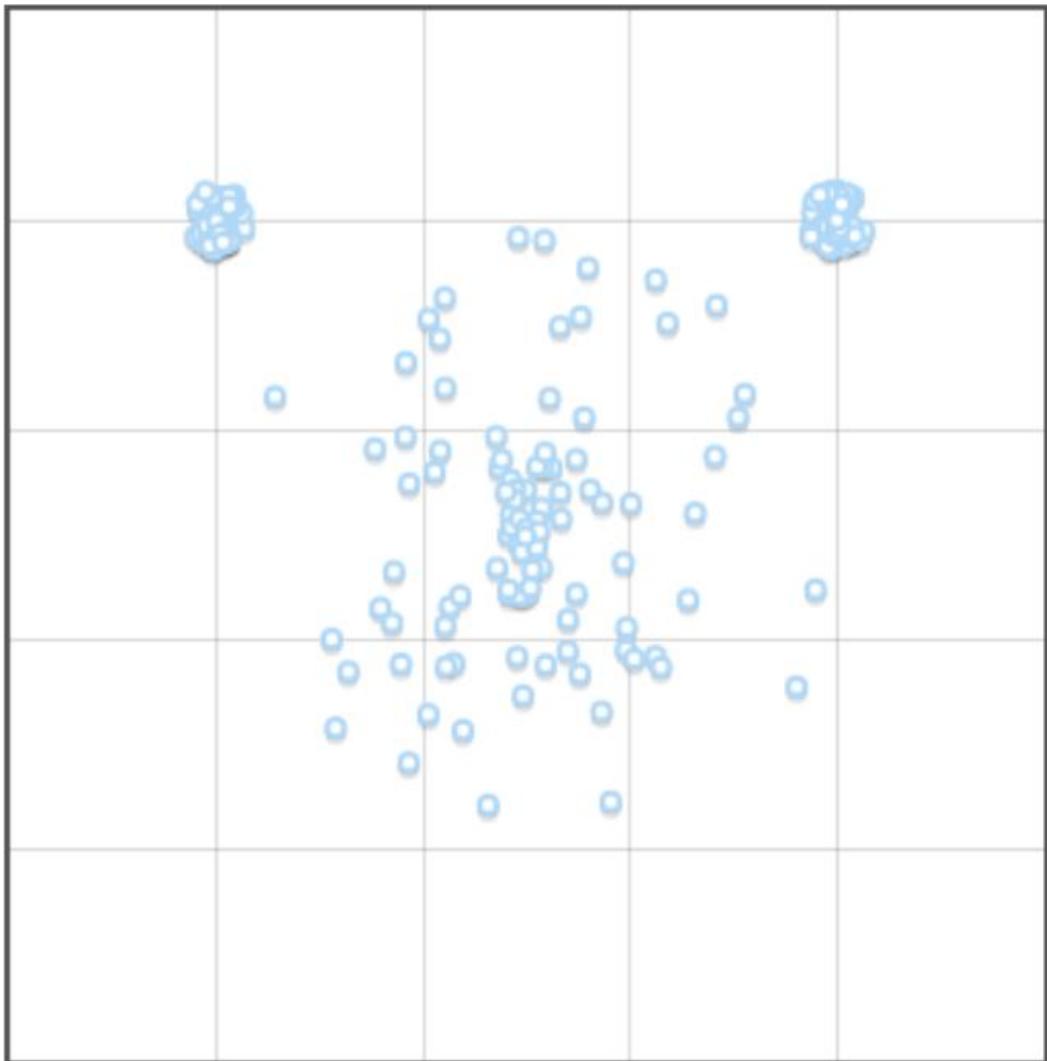
Проблемы K-Means

Несферические кластеры



Проблемы K-Means

Разноразмерные кластеры



Mean Shift

Задается не количество кластеров, а максимальный размер кластера.

Ищем плотные области, сдвигая центры.

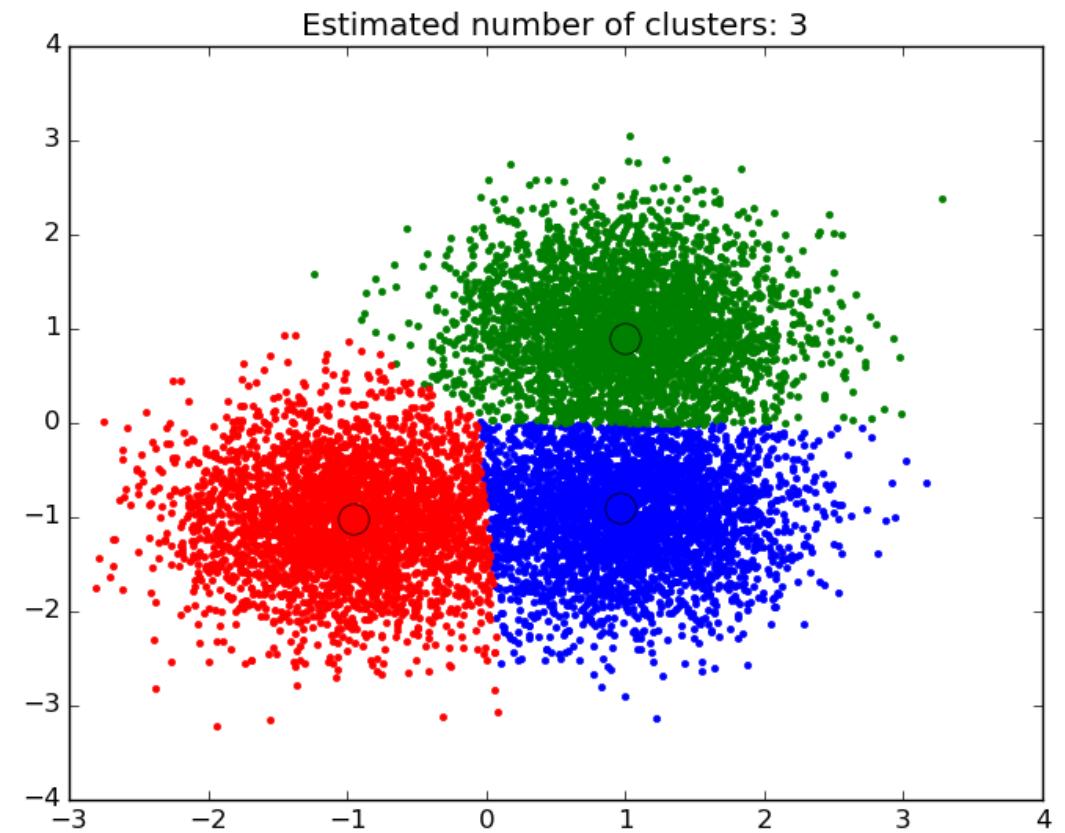
$$\mu_i(t + 1) = m(\mu_i(t))$$

$$m(\mu_i) = \frac{\sum_{x_j \in N(\mu_i)} K(x_j - \mu_i) x_j}{\sum_{x_j \in N(\mu_i)} K(x_j - \mu_i)},$$

где $N(\mu_i)$ – окрестность точки μ_i , K – RBF (Radial Basis Function):

$$K(x_j - \mu_i) = e^{-c \|x_j - \mu_i\|_2^2}$$

Последний шаг – фильтруем центры, убираем дубликаты.

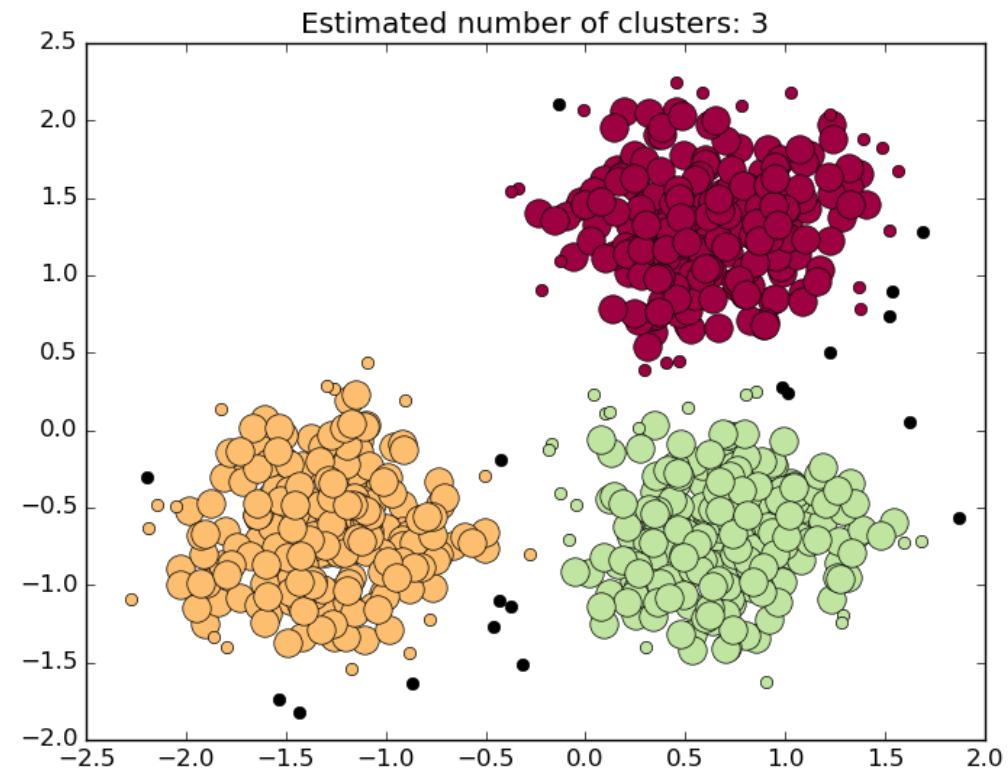


DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Количество кластеров – не задается.

Ищем объекты плотности (core samples) – такие объекты, в ϵ окрестности которых есть хотя бы m других объектов.

Объединяем объекты плотности на расстоянии ϵ и их окружение в кластеры.



Иерархическая кластеризация

Agglomerative Clustering

Иерархическая кластеризация снизу-вверх.

Начинаем с того, что каждая точка – отдельный кластер.

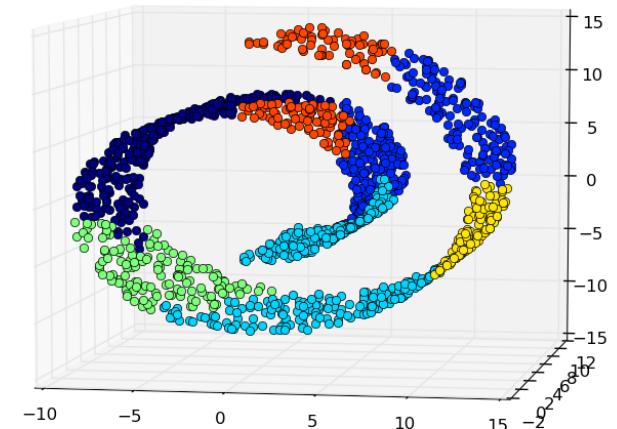
Три стратегии порядка слияния (linkage),
минимизирующие соответствующие величины:

- ward – дисперсия соединяемых кластеров
- average – среднее расстояние между объектами кластеров
- maximum (complete) – максимальное расстояние
между объектами кластеров

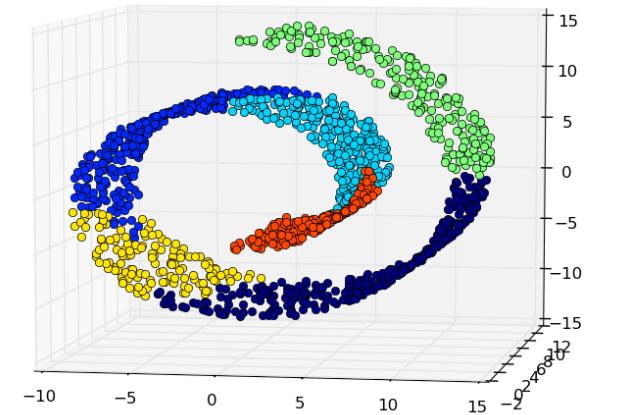
Сливаем пока не останется один кластер.

Можно добавить требование близости соединяемых кластеров (connectivity).

Without connectivity constraints (time 0.11s)



With connectivity constraints (time 0.16s)



Affinity Propagation

Affinity Propagation

Данные – близости объектов x_i и x_k - $s(i, k)$.

Распространяем до сходжения 2 величины –
ответственность $r(i, k)$ и доступность $a(i, k)$ (в начале - 0).

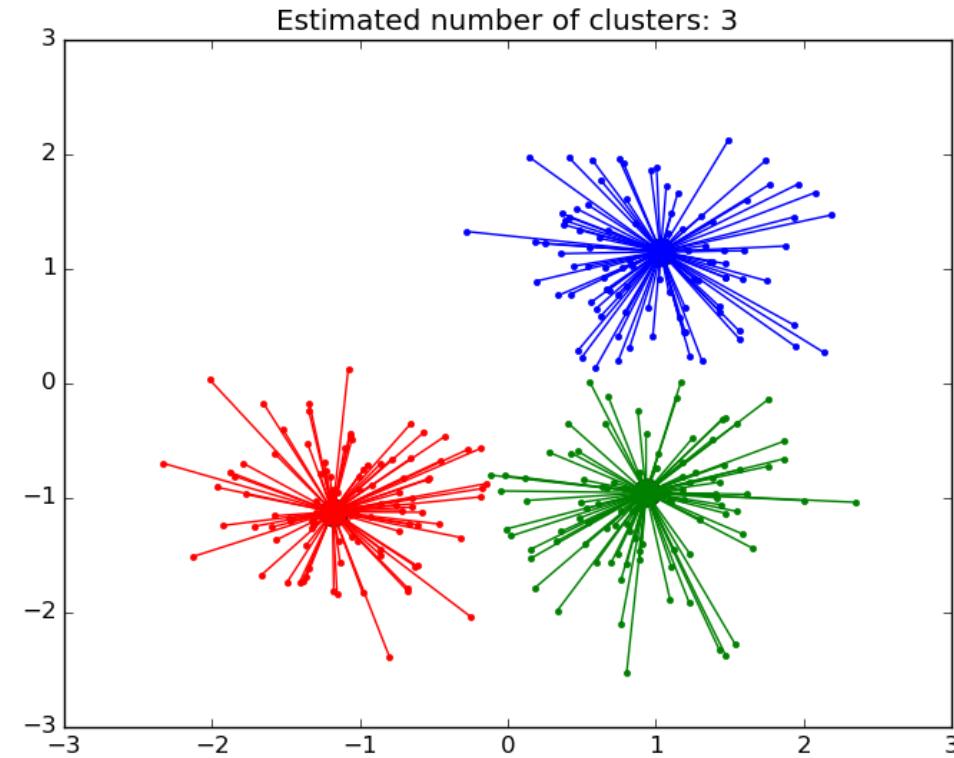
$$r(i, k) \leftarrow s(i, k) - \max_{m \neq k} (a(i, m) + s(i, m))$$

$$a(i, k) \leftarrow \begin{cases} \sum_{j \neq i} \max(0, r(j, k)), & \text{если } k = i \\ \min \left(0, r(k, k) + \sum_{j \neq i, k} \max(0, r(j, k)) \right), & \text{если } k \neq i \end{cases}$$

Выбираем самые «ответственные» и «доступные точки» точки:

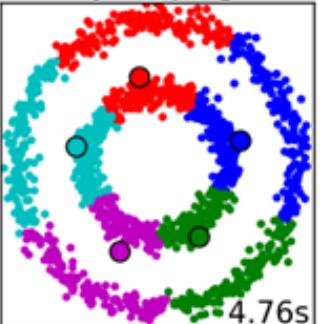
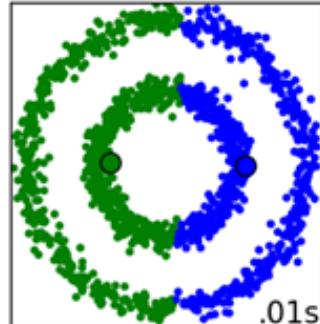
$$\mu_i = \operatorname{argmax}_k (a(i, k) + r(i, k))$$

$s(i, i)$ - параметр, регулирующий количество кластеров

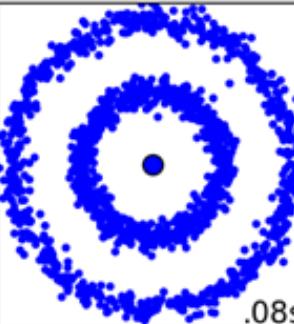


AgglomerativeClustering

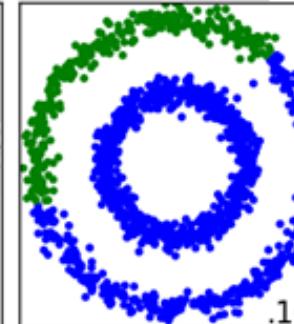
MiniBatchKMeans AffinityPropagation



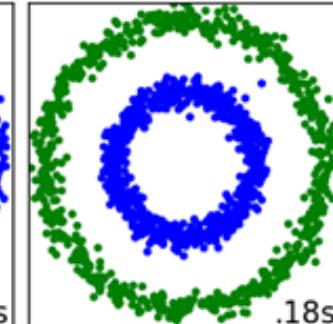
MeanShift



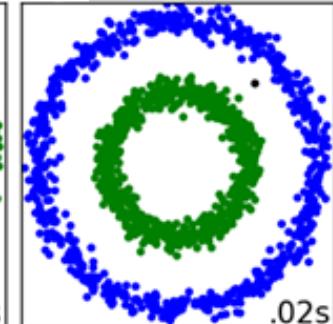
Ward



Average



DBSCAN



.02s

5.02s

.06s

.26s

.21s

.02s

.02s

4.19s

.06s

.18s

.21s

.03s

.02s

4.43s

.09s

.15s

.18s

.02s

Метрики кластеризации

Внутренние (internal):

- Davies-Bouldin index:

$$DB = \frac{1}{n} \sum_{i=1}^n \max_{j \neq i} \left(\frac{\rho(\mu_i, x^i) + \rho(\mu_j, x^j)}{\rho(\mu_i, \mu_j)} \right)$$

- Dunn index:

$$D = \frac{\min_{i \neq j} \rho(\mu_i, \mu_j)}{\max_{x_i, x_j \in \mu} (x_i, x_j)}$$

Внешние (external):

- Purity:

Доля точек максимального класса в кластерах

- *метрики классификации

Active learning

Semi-supervised learning

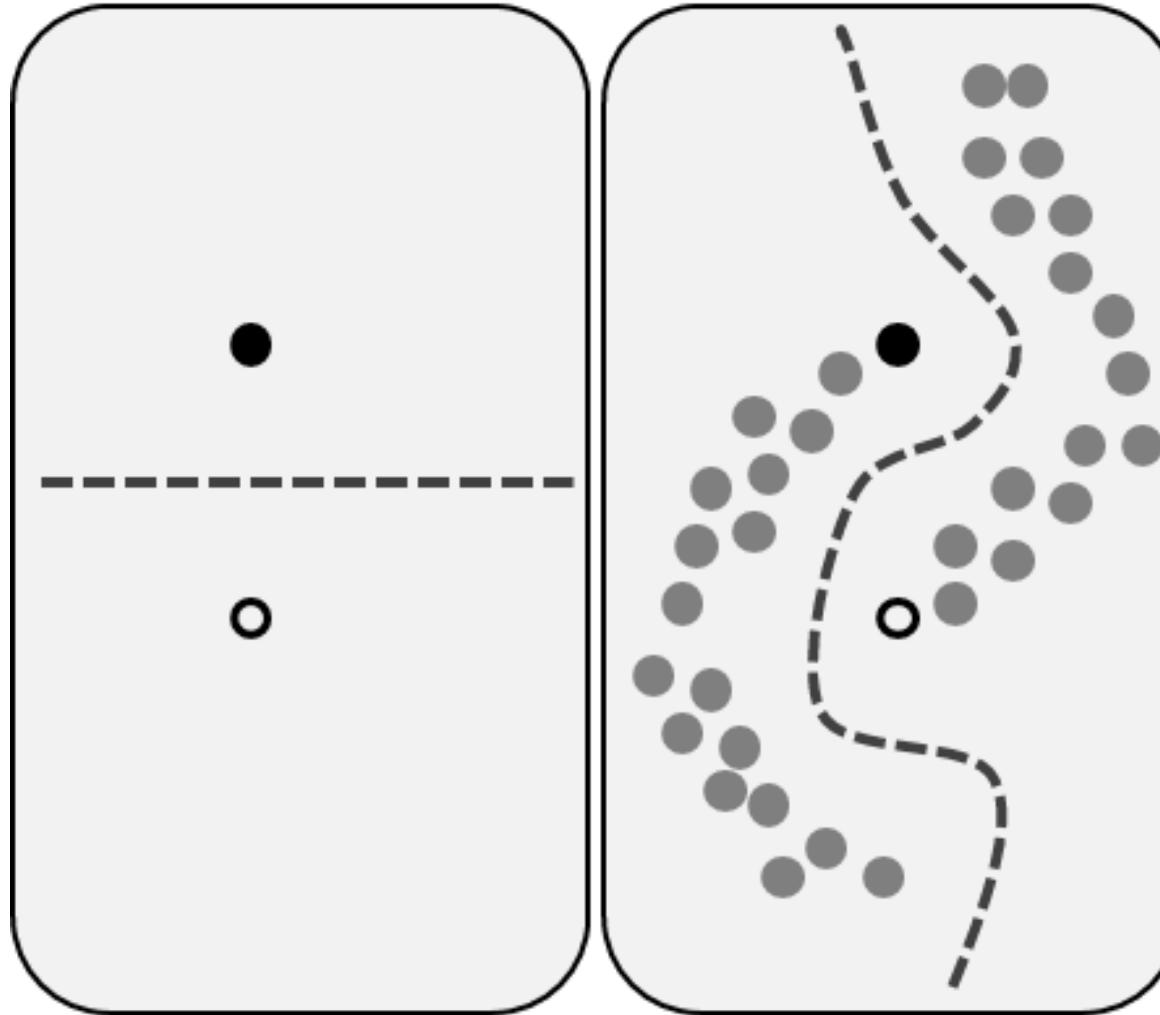
Зачастую получить данные (вектора признаков) довольно дешево, а разметить их довольно дорого, например из-за участия оценивающих людей.

Получить много данных можно, например:

- Веб-документы
- Речь
- Изображения и видео

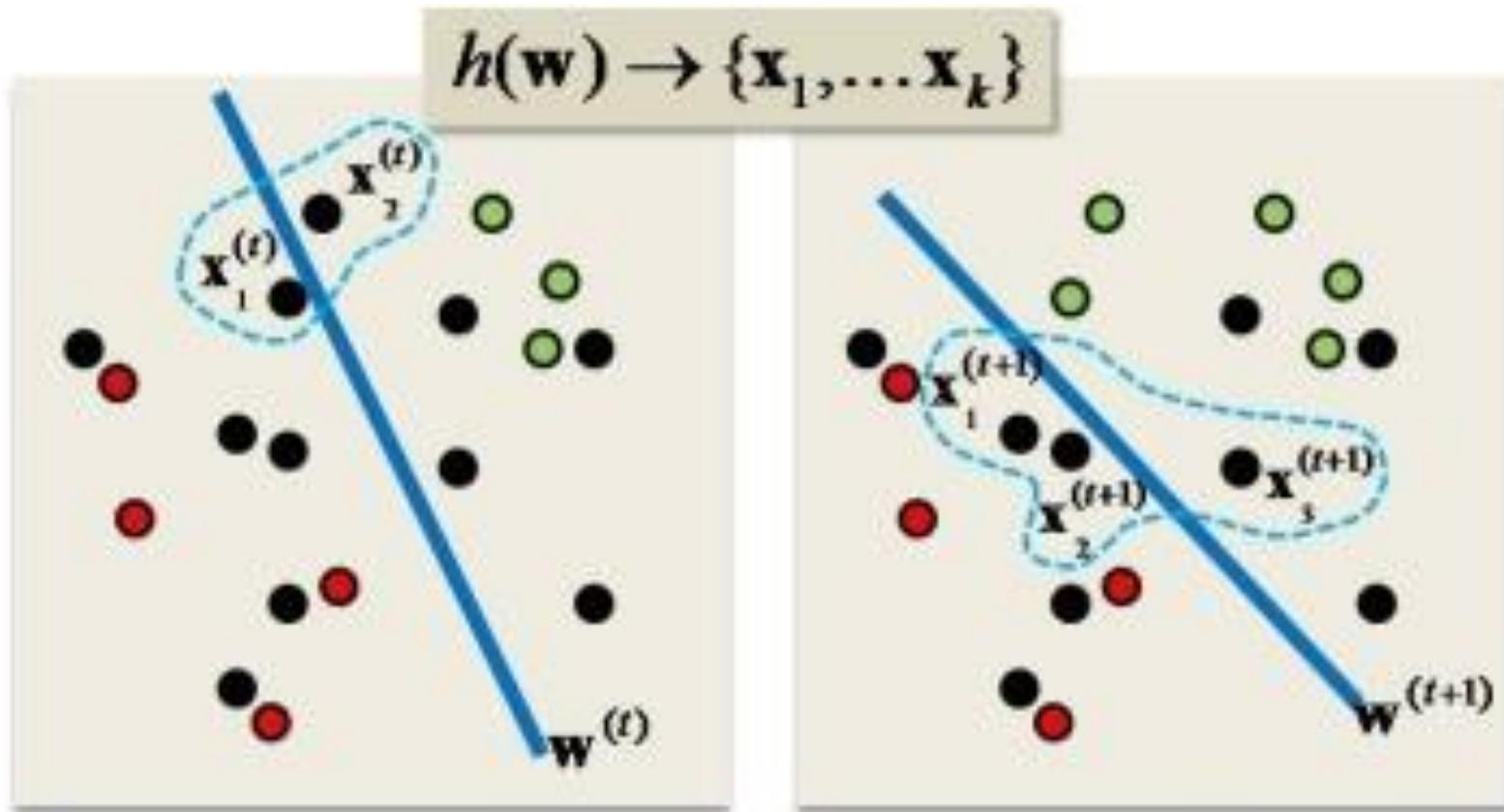
Тогда можно разметить не все вектора. Но информация о неразмеченных векторах все равно может пригодится.

Semi-supervised learning (обучение с частичным привлечением учителя)

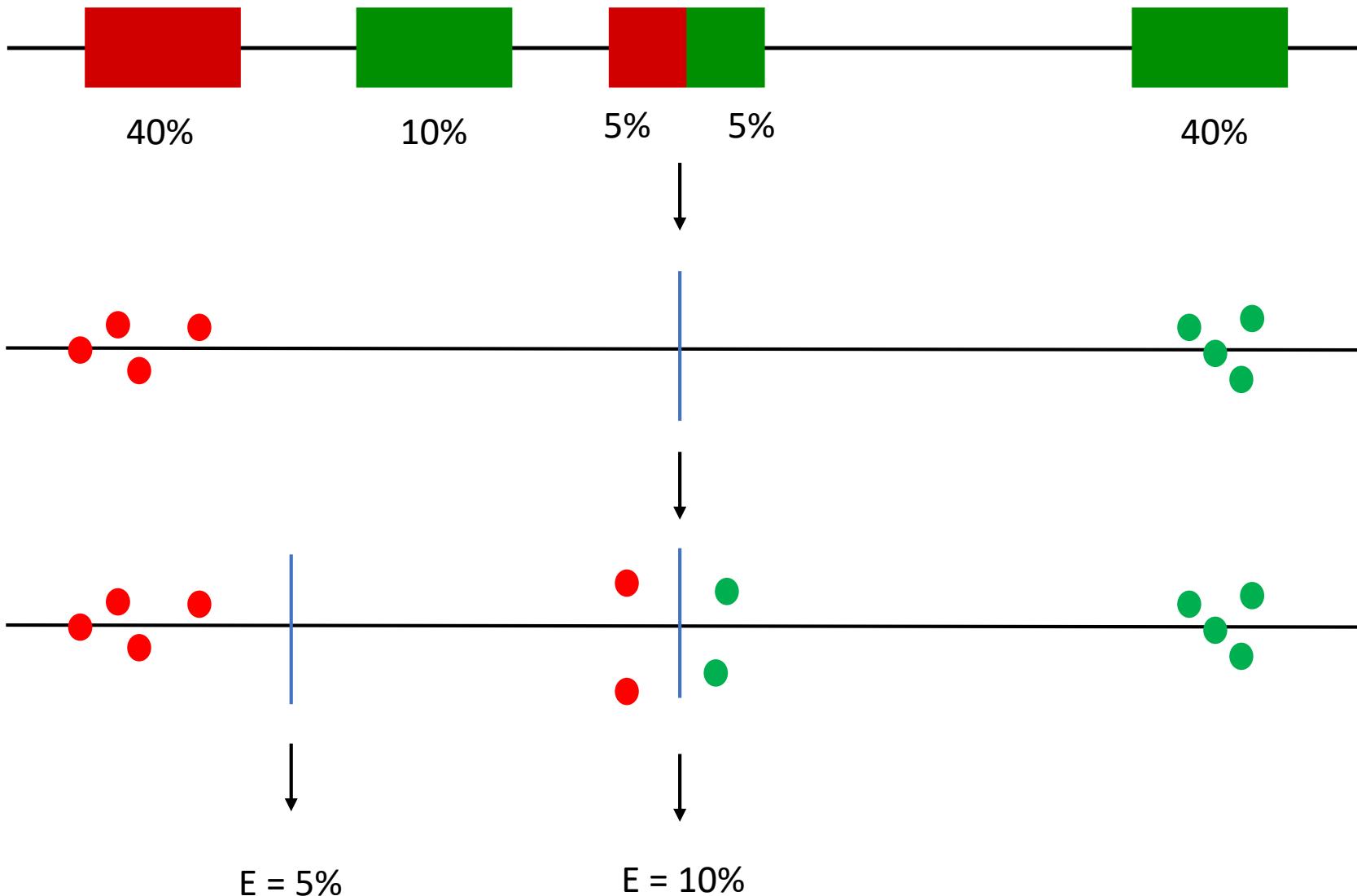


Active learning

Active learning – semi-supervised learning при котором мы сами выбираем вектора для оценки.

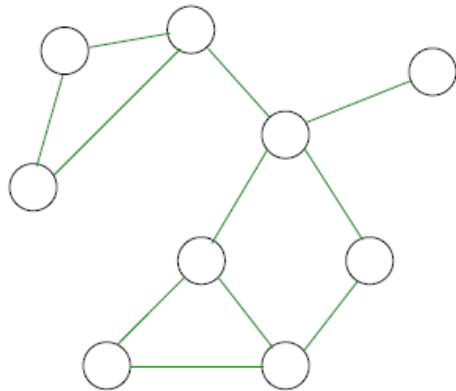


Sampling bias

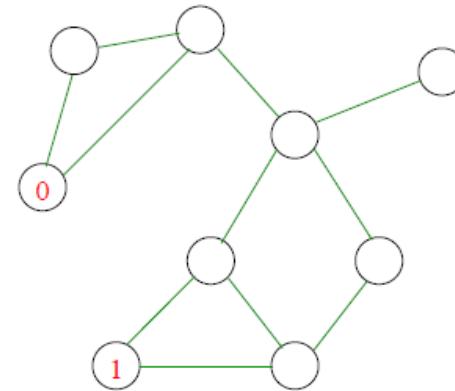


Распространение оценки Label propagation

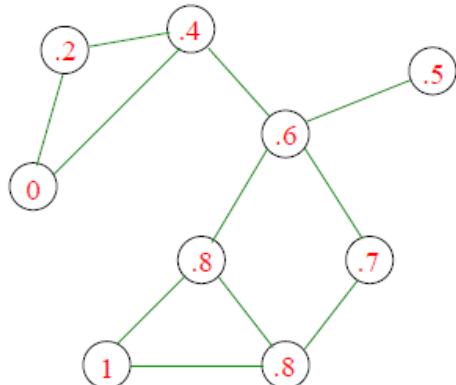
1) Построить граф близости



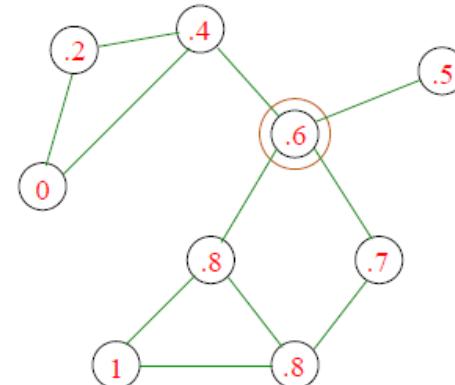
2) Оценить случайные точки



3) Распространить оценку

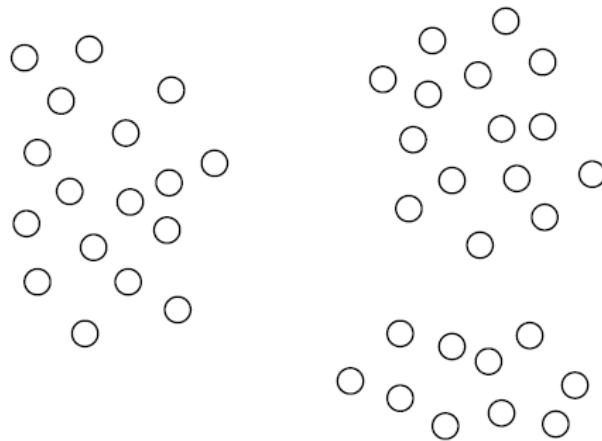


4) Оценить новую точку и вернуться в (3)

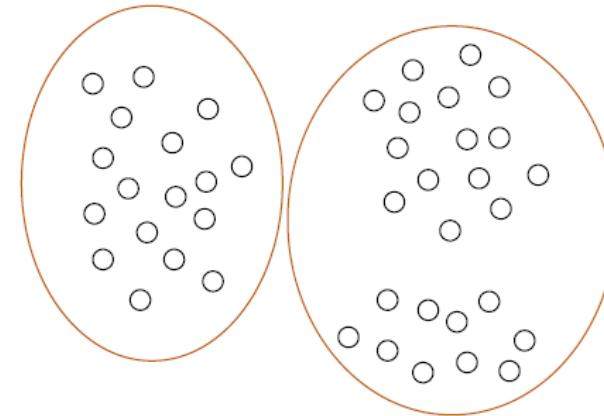


Переоценка кластеризации

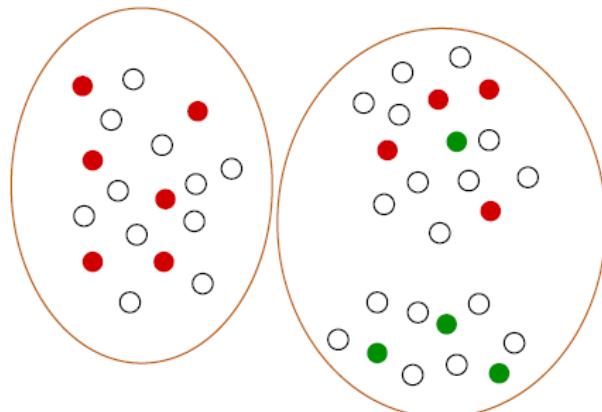
1) Неразмеченные данные



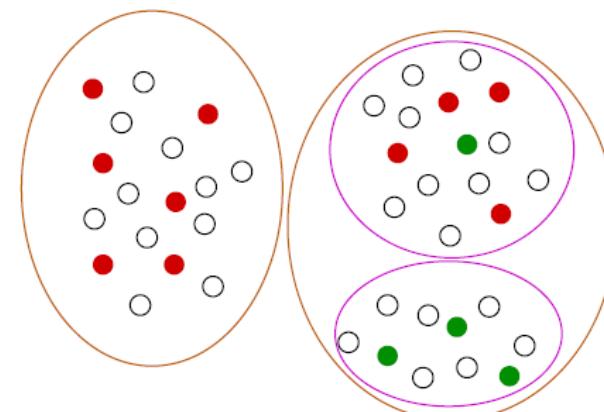
2) Найти кластеризацию



3) Разметить случайные точки в кластерах



4) Уточнить кластеризацию



Decision Trees

Логическая закономерность (Rule)

$$R_y: X \rightarrow \{0, 1\}$$

Виды закономерностей

Пороговые условия:

$$R(x) = [a \leq x_i \leq b]$$

Синдром пороговых условий:

$$R(x) = \left[\sum_{j \in J} [a_j \leq x_j \leq b_j] \geq d \right]$$

При $d = |J|$ - конъюнкция условий.

Оценки правил (классификаторов)

		y_i		
		= y	$\neq y$	
$R_y(x_i)$	1	True positive (TP)	False positive (FP)	Positive predictive value, Precision $\frac{TP}{TP + FP}$
	0	False negative (FN)	True negative (TN)	
		True positive rate, Recall $\frac{TP}{TP + FN}$		Accuracy $\frac{TP + TN}{TP + FP + TN + FN}$

$$F_1 score = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 \frac{precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN}$$

Пример

		y_i		
		$= y$	$\neq y$	
$R_y(x_i)$	1	0 (TP)	0 (FP)	Positive predictive value, Precision 0
	0	50 (FN)	950 (TN)	
		True positive rate, Recall 0		Accuracy 95%

$$F_1 score = 0$$

Пример #2

		y_i		
		= y	$\neq y$	
$R_y(x_i)$	1	50 (TP)	950 (FP)	Positive predictive value, Precision 5%
	0	0 (FN)	0 (TN)	
		True positive rate, Recall 100%		Accuracy 5%

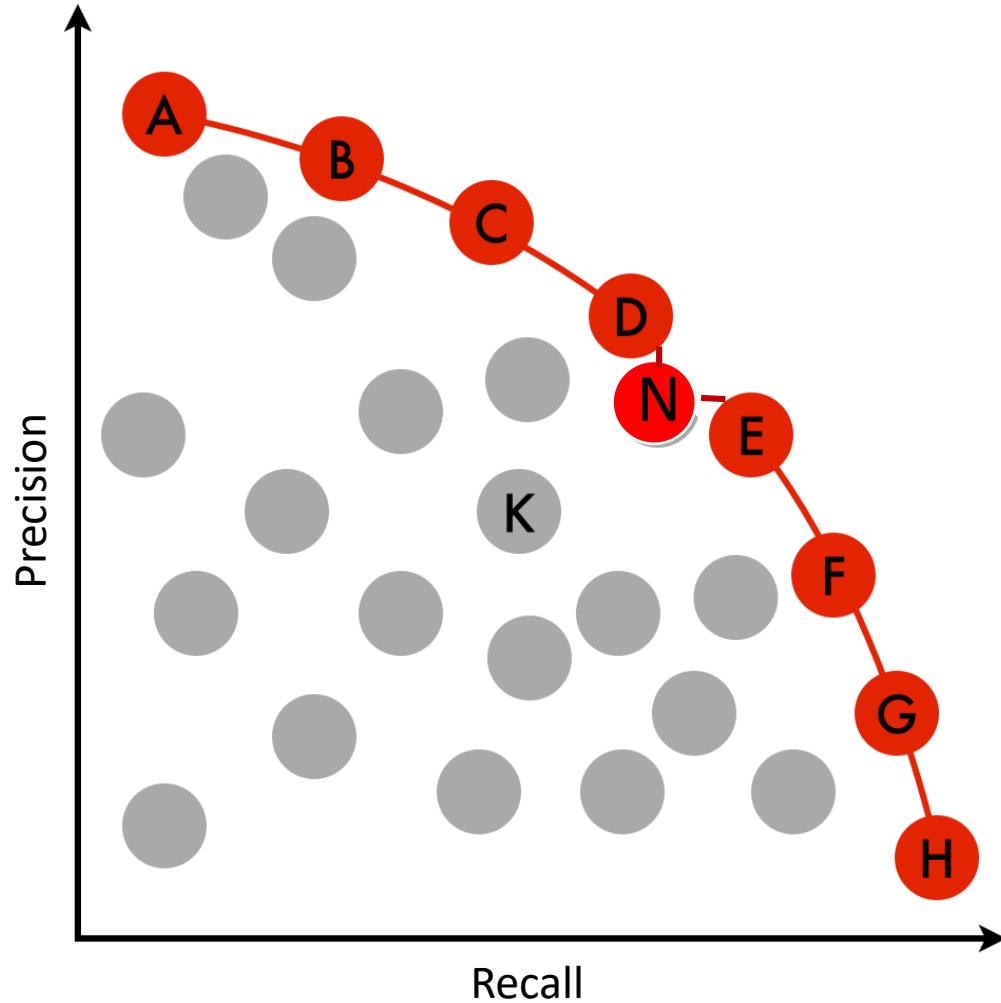
$$F_1 score = 10\%$$

Пример #3

		y_i		
		= y	$\neq y$	
$R_y(x_i)$	1	25 (TP)	475 (FP)	Positive predictive value, Precision 5%
	0	25 (FN)	475 (TN)	
		True positive rate, Recall 50%		Accuracy 50%

$$F_1 score = 9\%$$

Эффективность по Парето



● - Парето фронт, 1-я линия

N – тоже 1-я линия, K – 2-я

Эффективное по Парето правило – такое, что нет другого правила, у которого одновременно выше и precision и recall.

Оценки правил (классификаторов)

		y_i		
		= y	$\neq y$	
$R_y(x_i)$	1	True positive (TP)	False positive (FP)	Positive predictive value, Precision $\frac{TP}{TP + FP}$
	0	False negative (FN)	True negative (TN)	
		True positive rate, Recall $\frac{TP}{TP + FN}$	False positive rate $\frac{FP}{FP + TN}$	Accuracy $\frac{TP + TN}{TP + FP + TN + FN}$

$$F_1 score = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 \frac{precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN}$$

Пример #1

		y_i		
		$= y$	$\neq y$	
$R_y(x_i)$	1	0 (TP)	0 (FP)	Positive predictive value, Precision 0%
	0	50 (FN)	950 (TN)	
		True positive rate, Recall 0%	False positive rate 0%	Accuracy 95%

$$F_1 score = 0$$

Пример #2

		y_i		
		= y	$\neq y$	
$R_y(x_i)$	1	50 (TP)	950 (FP)	Positive predictive value, Precision 5%
	0	0 (FN)	0 (TN)	
		True positive rate, Recall 100%	False positive rate 100%	Accuracy 5%

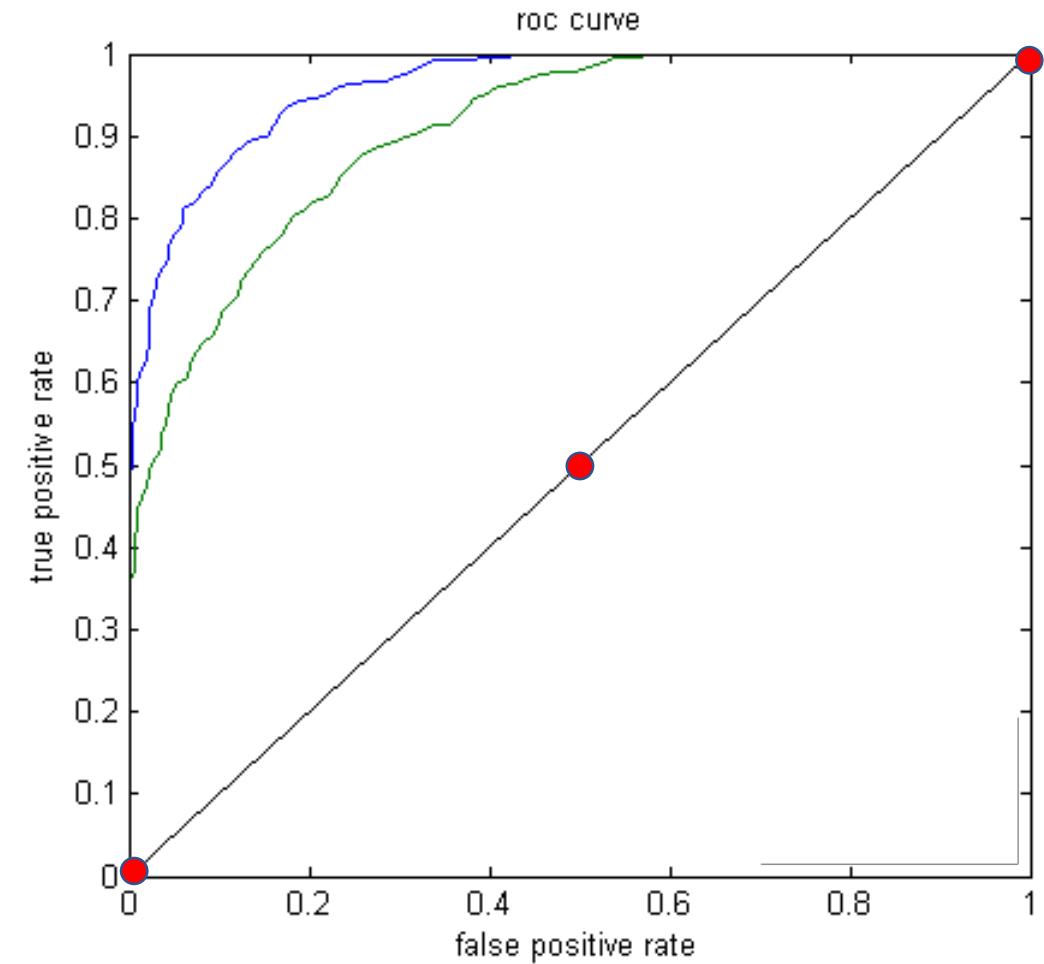
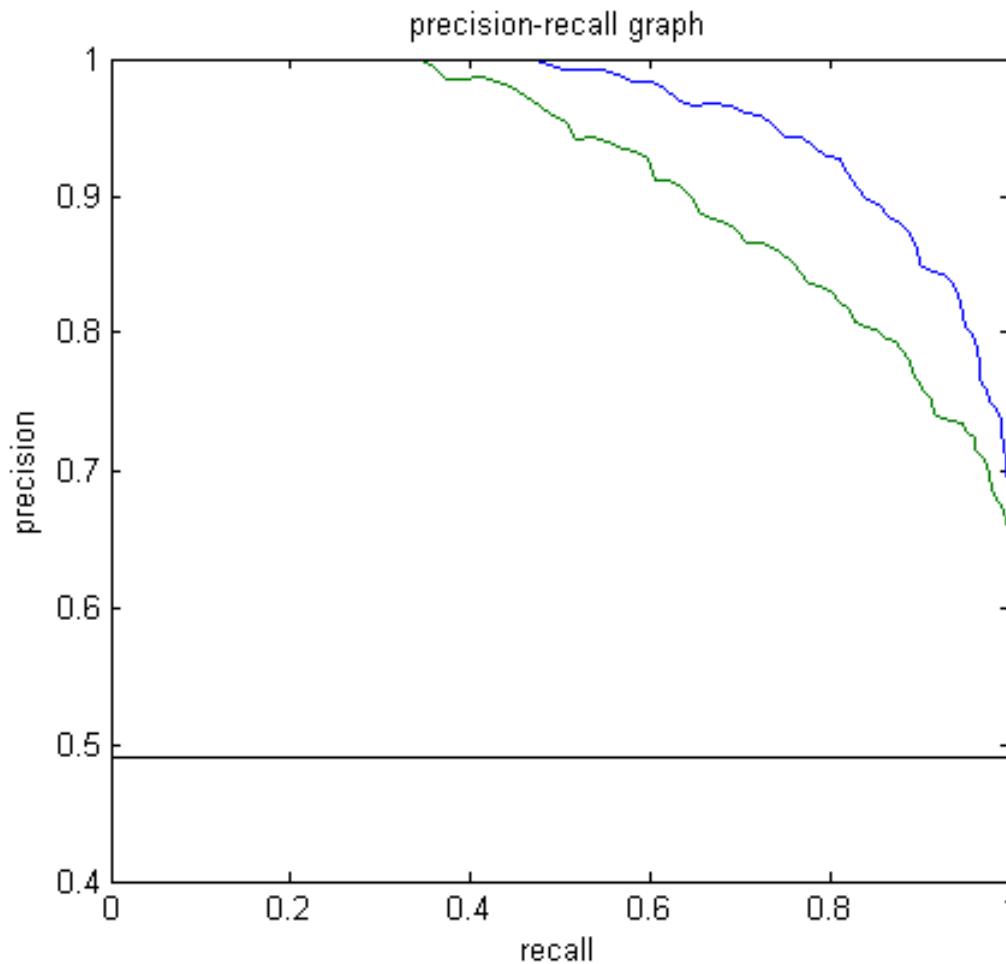
$$F_1 score = 10\%$$

Пример #3

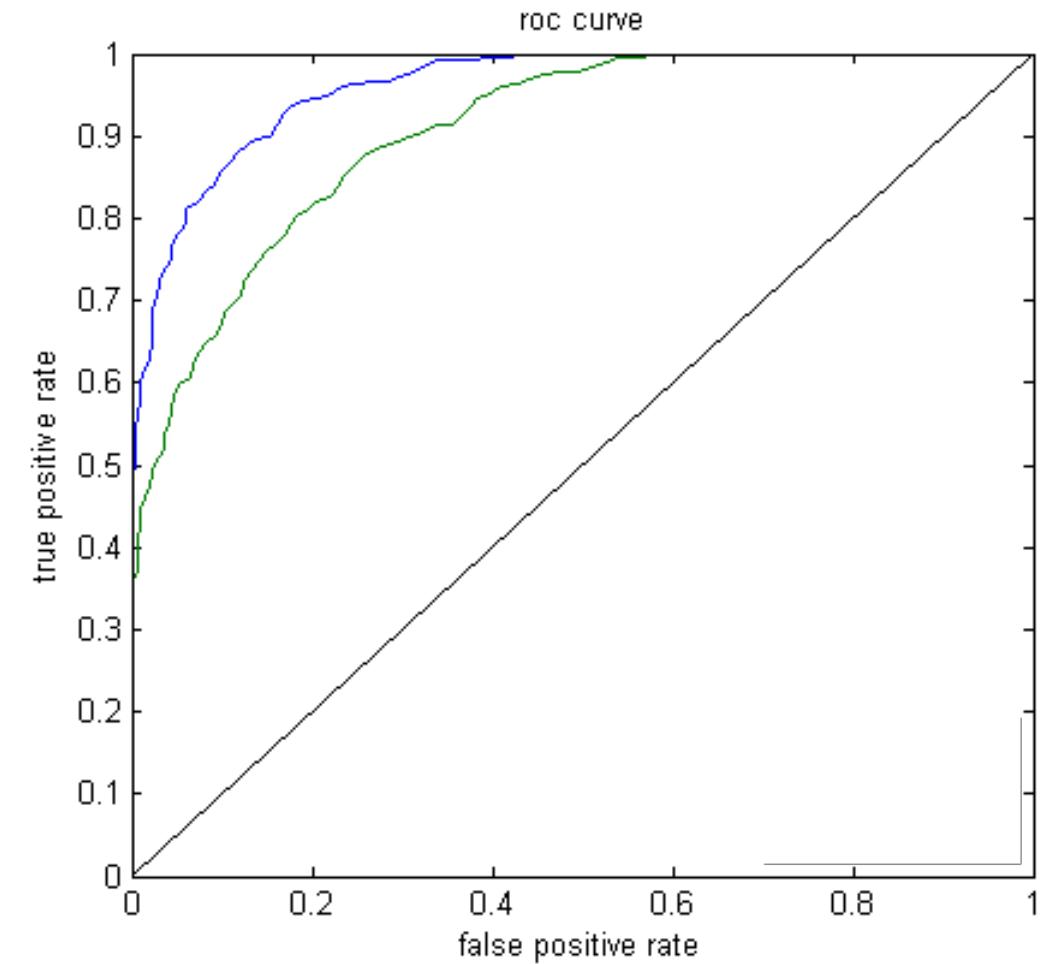
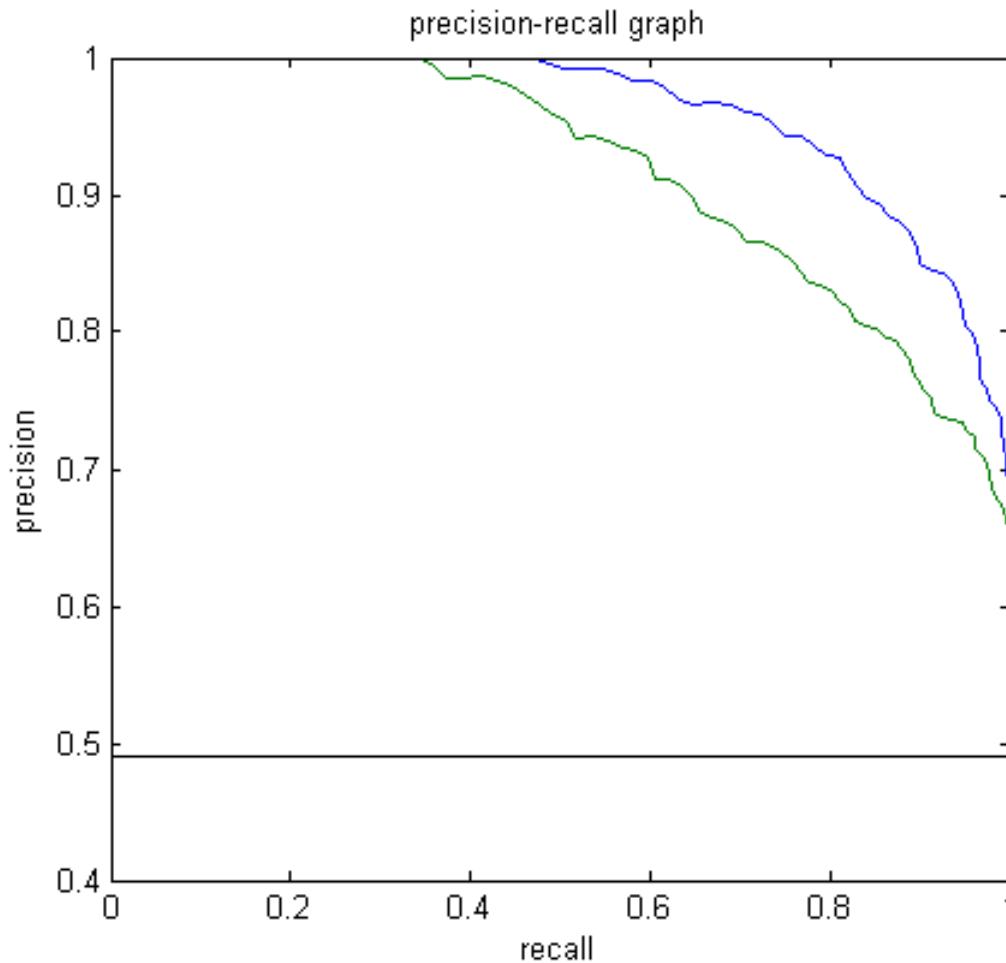
		y_i		
		$= y$	$\neq y$	
$R_y(x_i)$	1	25 (TP)	475 (FP)	Positive predictive value, Precision 5%
	0	25 (FN)	475 (TN)	
		True positive rate, Recall 50%	False positive rate 50%	Accuracy 50%

$$F_1 score = 9\%$$

ROC (Receiver Operating Characteristic)



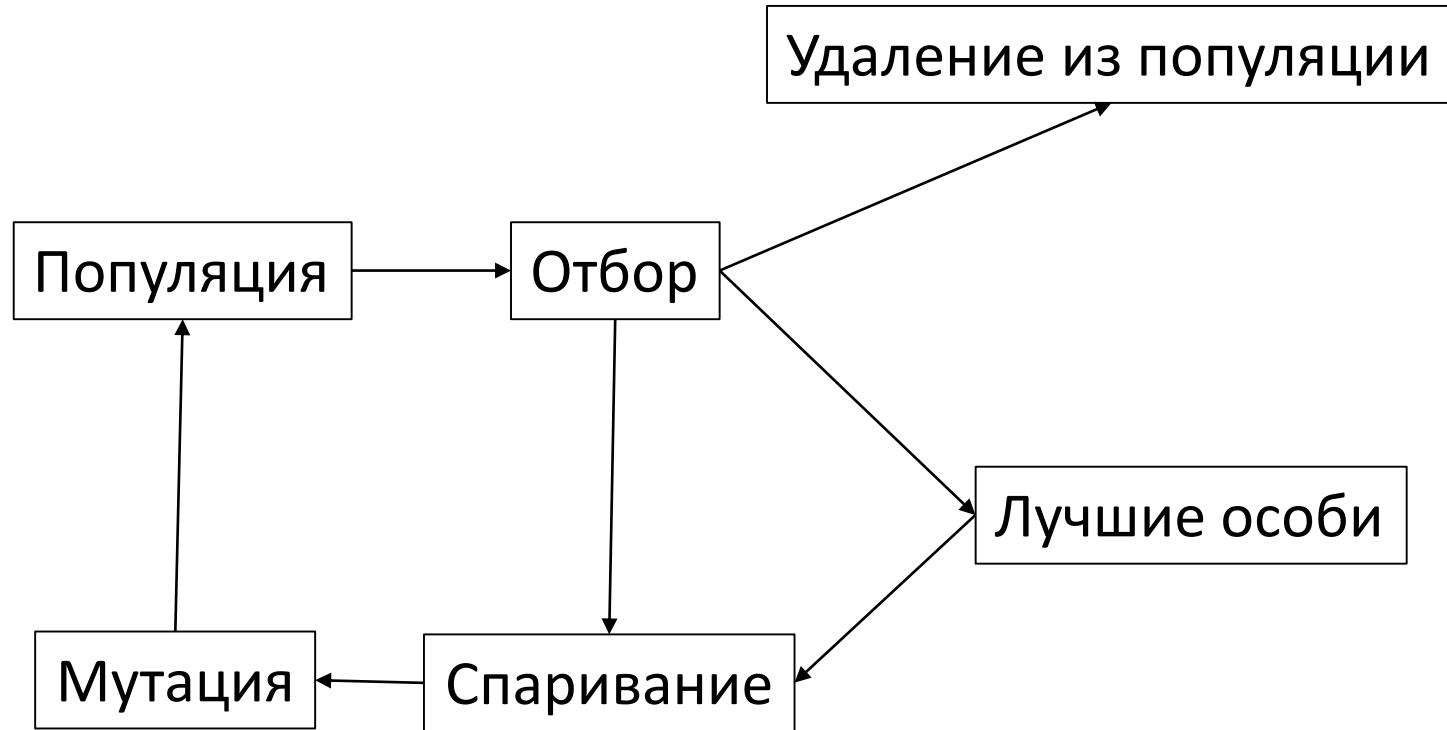
ROC (Receiver Operating Characteristic)



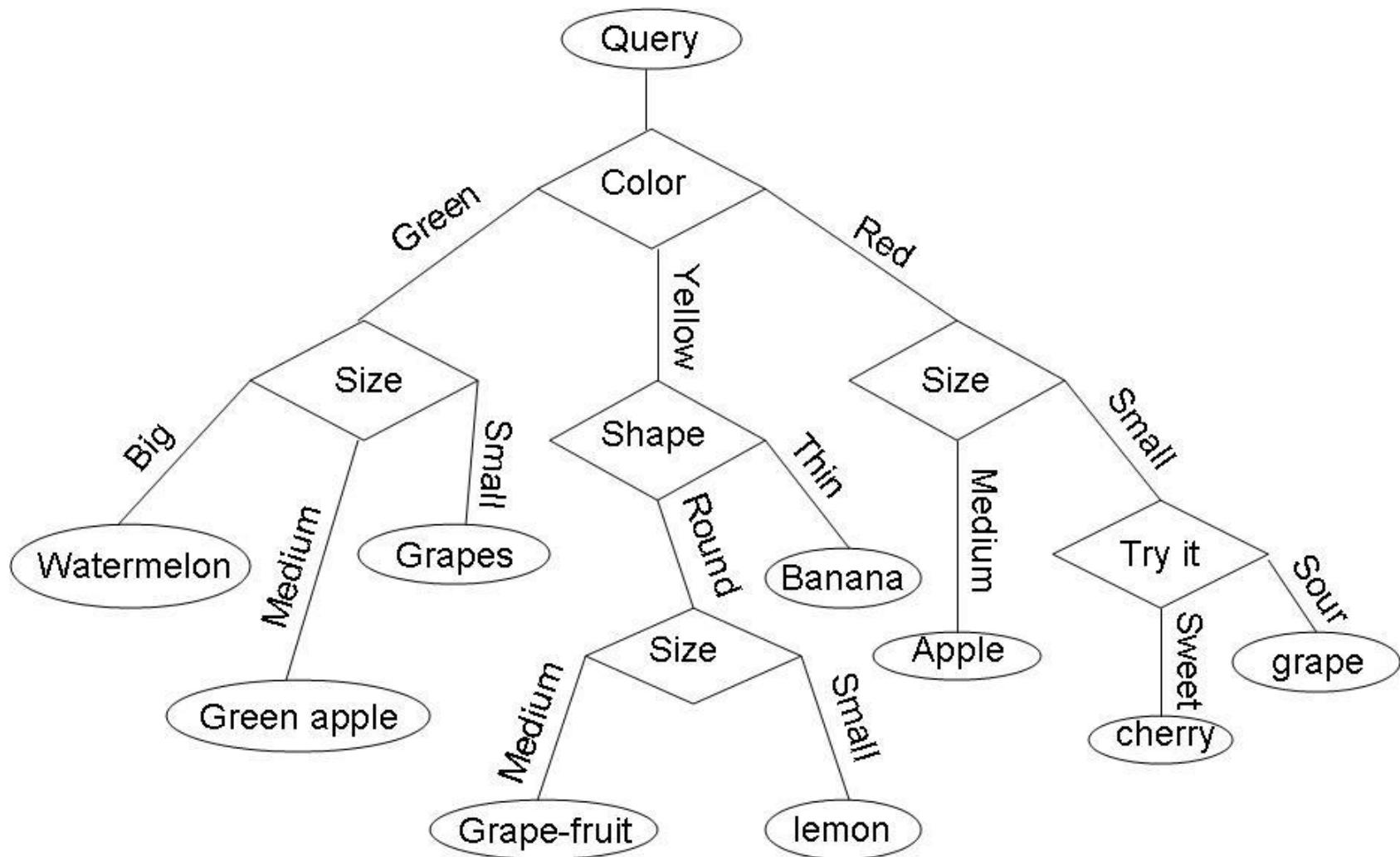
AUC – Area Under the ROC Curve

Отбор и генерация правил

Генетический алгоритм



Деревья решений



Оценки логических закономерностей

Information Gain (критерий информационного выигрыша):

$$IG(R) = H(X) - \frac{|R^1|}{|X|}H(R^1) - \frac{|R^0|}{|X|}H(R^0)$$

$$H(X) = - \sum_{y \in Y} \frac{|\{x_i : y_i = y\}|}{|X|} \times \log_2 \frac{|\{x_i : y_i = y\}|}{|X|} \quad \text{— энтропия}$$

Gini impurity (критерий Джини)

$$I_g(X) = \sum_{y \in Y} \frac{|\{x_i : y_i = y\}|}{X} \frac{|\{x_i : y_i \neq y\}|}{X}$$

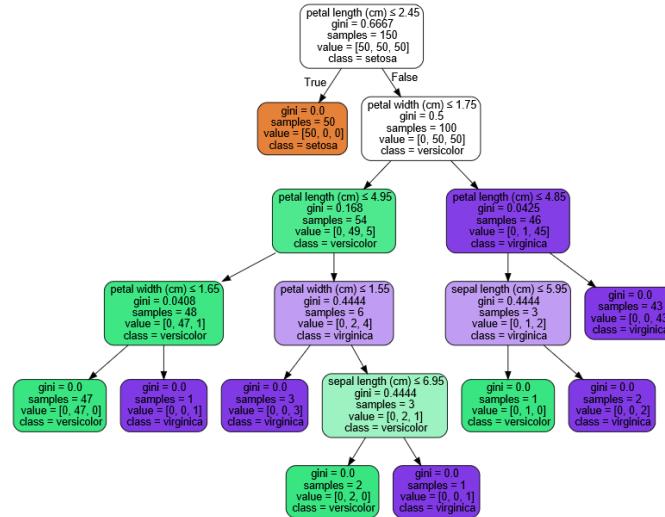
Вероятность того, что мы ошибемся, если будем элементам из множества приписывать класс согласно распределению классов в множестве.

Алгоритмы ID3 и C4.5 (Iterative Dichotomizer 3 и C4.5)

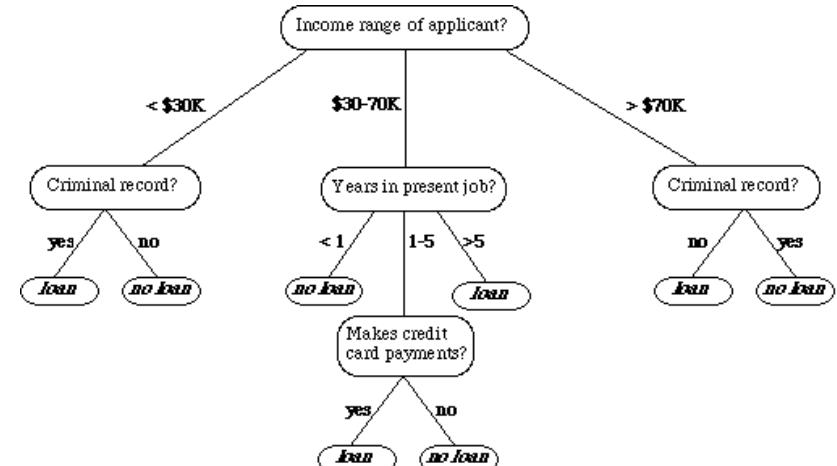
Автор: John Ross Quinlan

1. Если все объекты лежат в одном классе → вернуть лист с этим классом.
2. Найти правило с лучшим нормализованным информационным выигрышем или по Джини.
3. Правила с информационным выигрышем нет → вернуть лист с мажоритарным классом.
4. Разделить по этому правилу объекты.
5. Вызвать 1. для каждого листа, на который разделены объекты.

ID3



C4.5



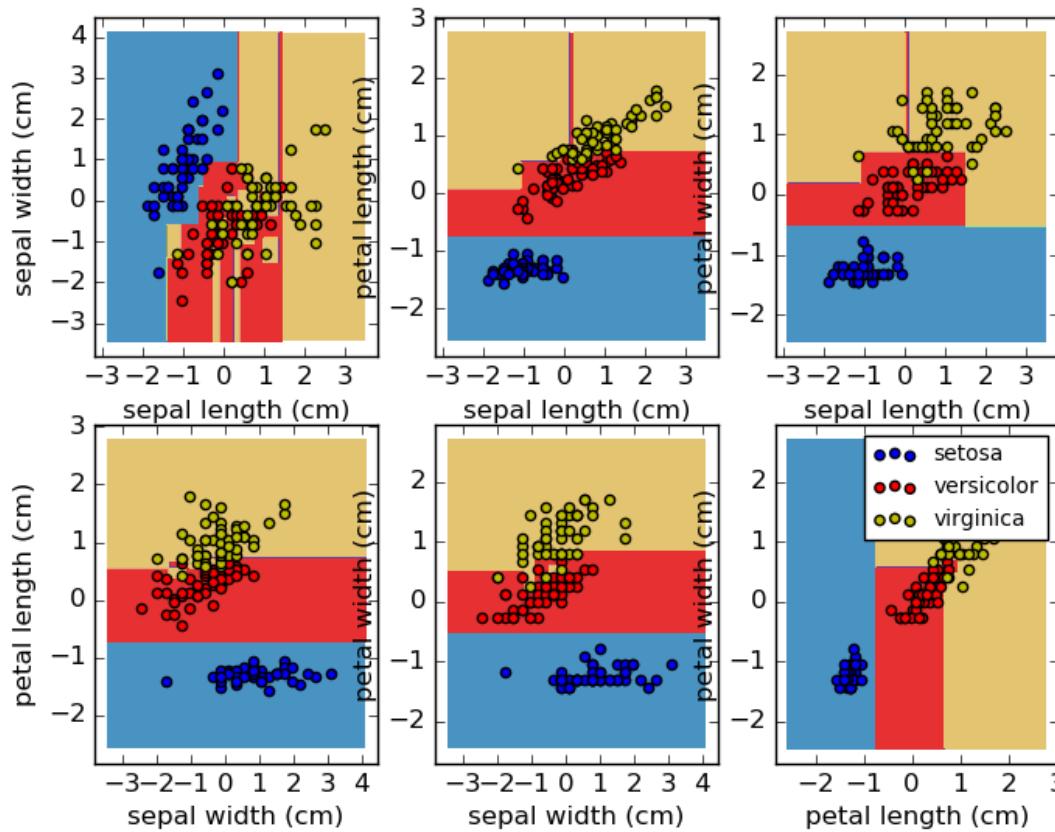
Алгоритм CART (Classification and Regression Trees)

Дисперсия:

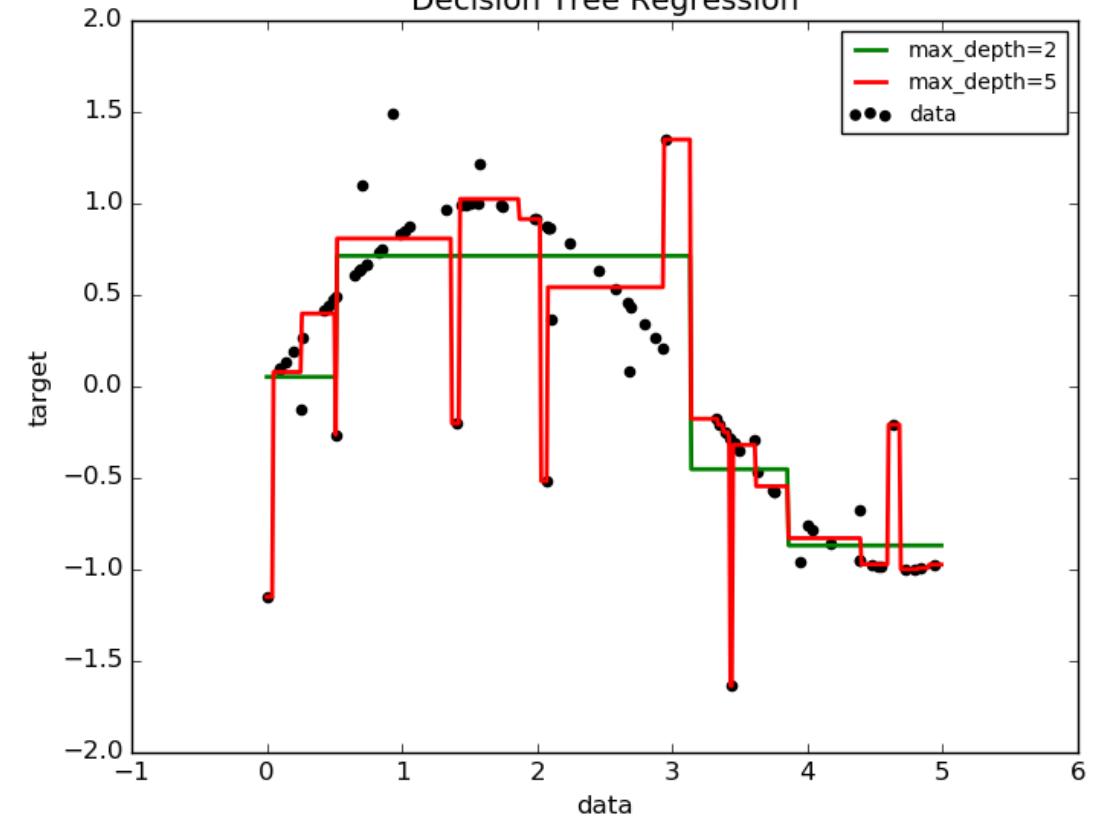
$$V(X) = \frac{1}{|X|^2} \sum_{x_i \in X} \sum_{x_j \in X} \frac{1}{2} (y_i - y_j)^2$$

Пример работы

Decision surface of a decision tree using paired features



Decision Tree Regression



Regularization and Train\Test

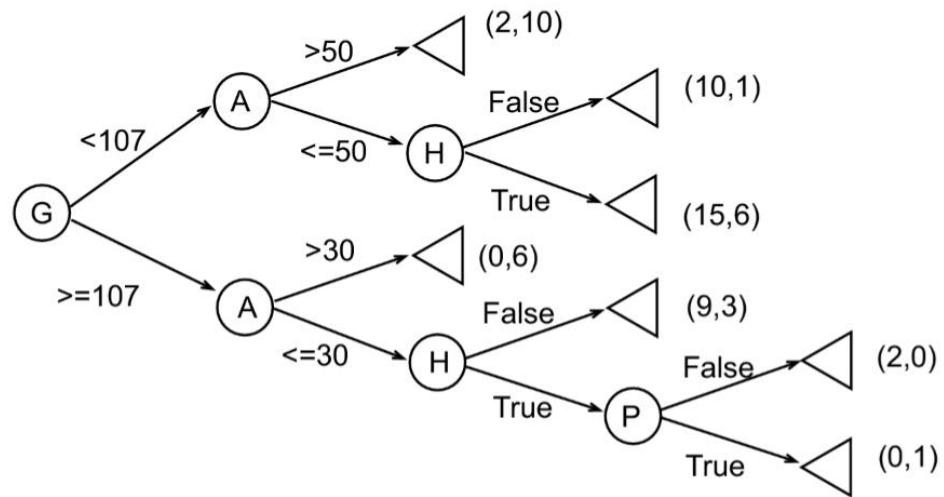
Выделим долю точек ($\approx 10 - 20\%$) в отдельную выборку - Test.

Обучать будем на оставшихся – Train.

По метрике на Test можно оптимизировать параметры модели (дерева), например – максимальную глубину.

Oblivious Decision Trees (Небрежные решающие деревья)

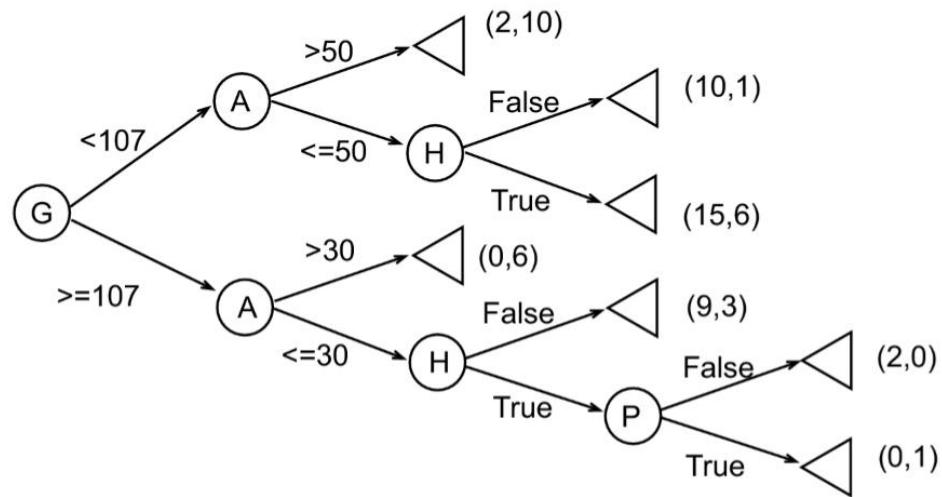
На одном уровне – одинаковые признаки.



(G)lucose level
(A)ge
(H)ypertension
(P)regnancy

Oblivious Decision Trees (Небрежные решающие деревья)

На одном уровне – одинаковые признаки.



(G)lucose level
(A)ge
(H)ypertension
(P)regnancy

MatrixNet (Yandex) – Oblivious Decision Trees, где правило для разделения тоже одинаковое на каждом уровне. В этом случае порядок правил не важен и оценку и удаление признаков легко организовать.

Random Forests (Случайные леса)

Обучим много деревьев на «случайных данных»

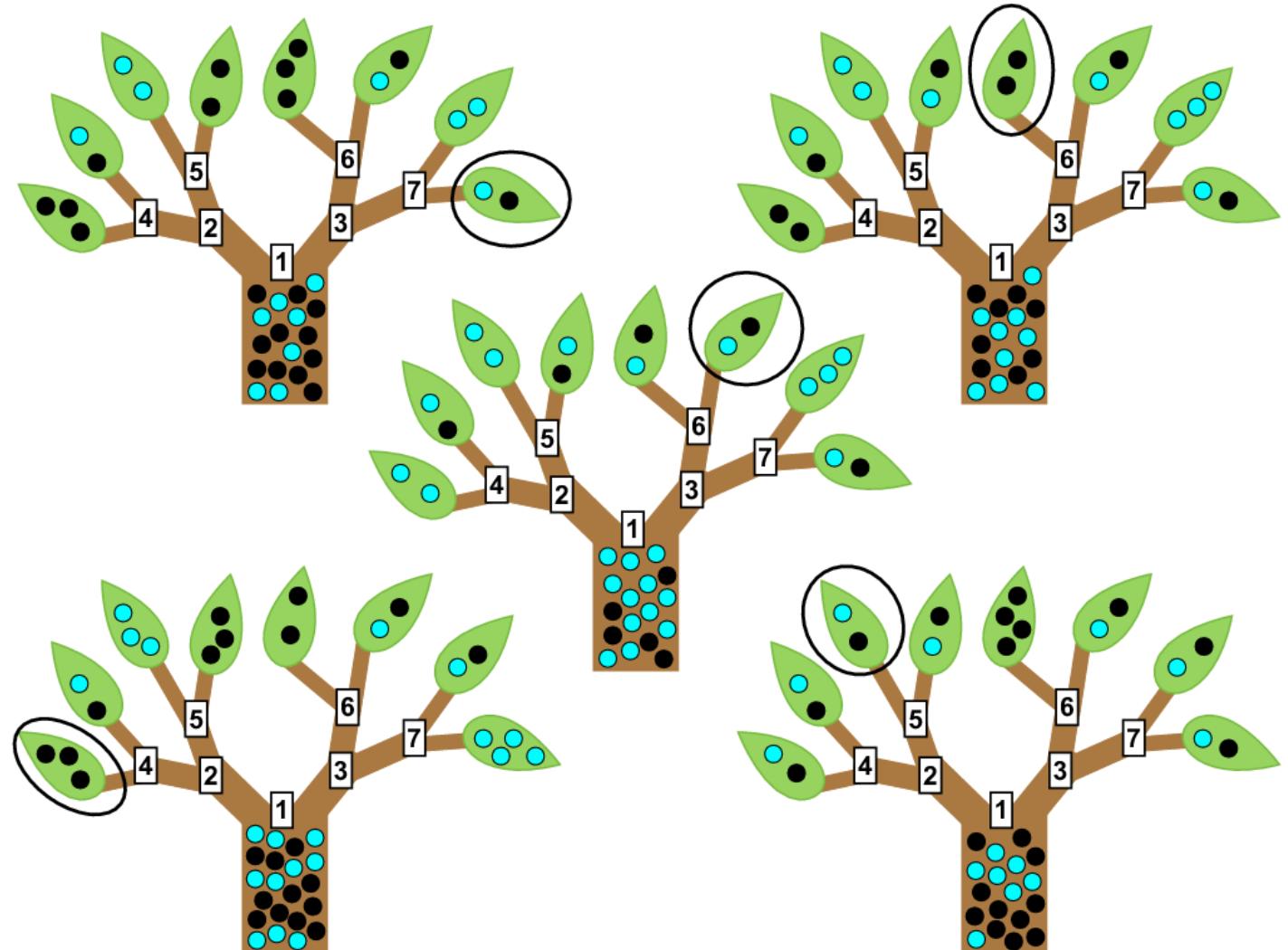
Случайные данные:

1) Датасеты с помощью **bagging (bootstrap aggregating)** – вытаскиваем из исходного датасета столько же данных, но с повторениями.

2) Берем случайное подмножество признаков.

Результат – голосование, вероятность, логарифм вероятностей.

Параметры – количество используемых примеров и признаков для построения каждого дерева.



Немножко истории статистики

Sampling

Случайная выборка

Jackknife resampling
(John Tukey, 1958)

Выкидываем поочередно по одному примеру

«like a Boy Scout's jackknife, it is a "rough and ready" tool that can solve a variety of problems even though specific problems may be more efficiently solved with a purpose-designed tool»

Bootstrapping (Bootstrap resampling)
(Bradley Efron, 1979)
«to pull oneself up by one's bootstraps»

Набираем выборку того же размера
с повторениями из исходной

Bagging (Bootstrap AGGREGatING)
(Leo Breiman, 1994)

Используем разные bootstrap*
выборки для обучения

Линейные классификаторы

и немножко теории...

Perceptron

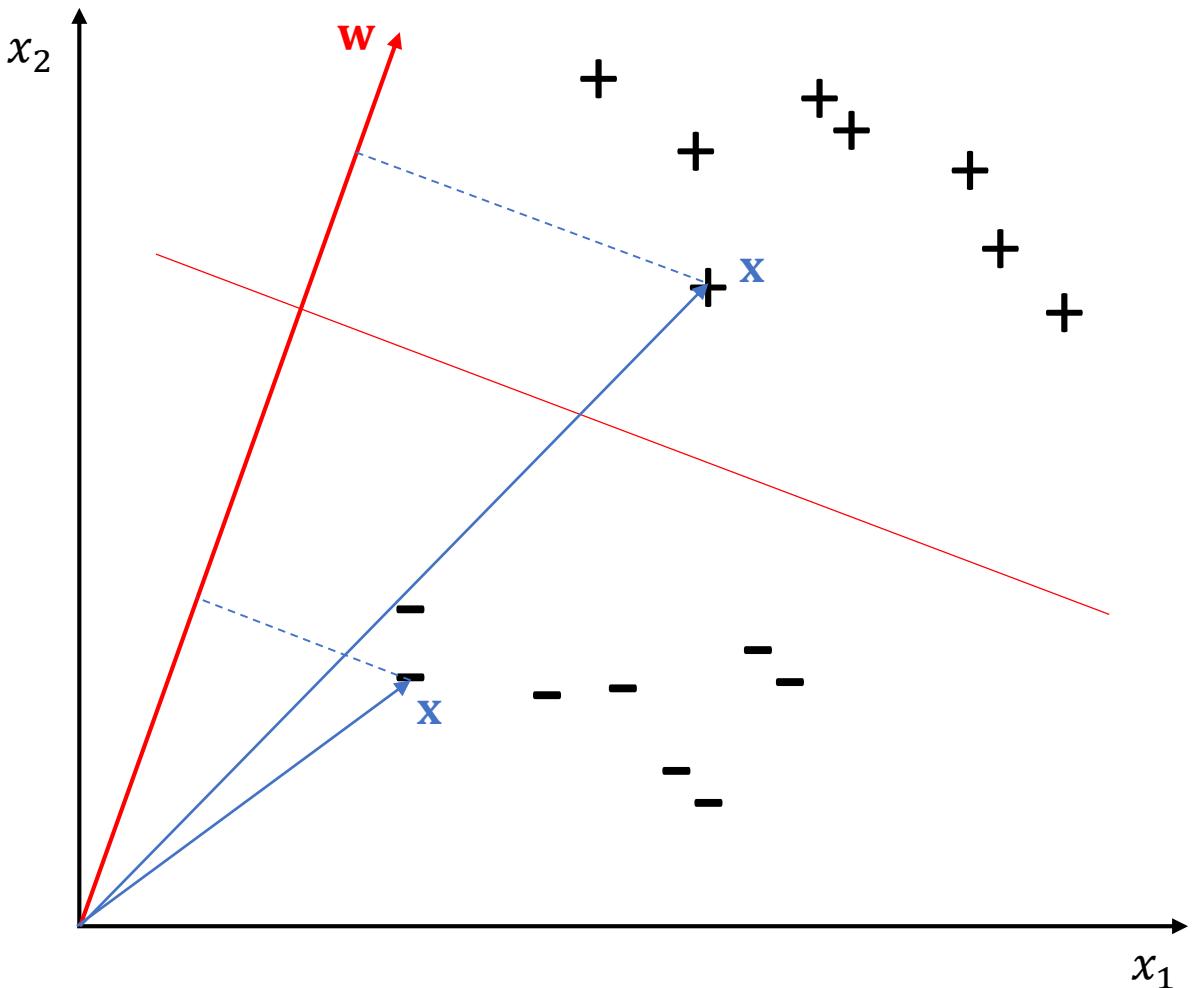
$$y \in \{-1,1\}$$

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

$$h(\mathbf{x}) = \text{sign} \left(\left(\sum_{i=1}^d w_i x_i \right) - w_0 \right)$$

$$h(\mathbf{x}) = \text{sign} \left(\sum_{i=0}^d w_i x_i \right)$$

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x})$$



Обучение перцептрана

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x})$$

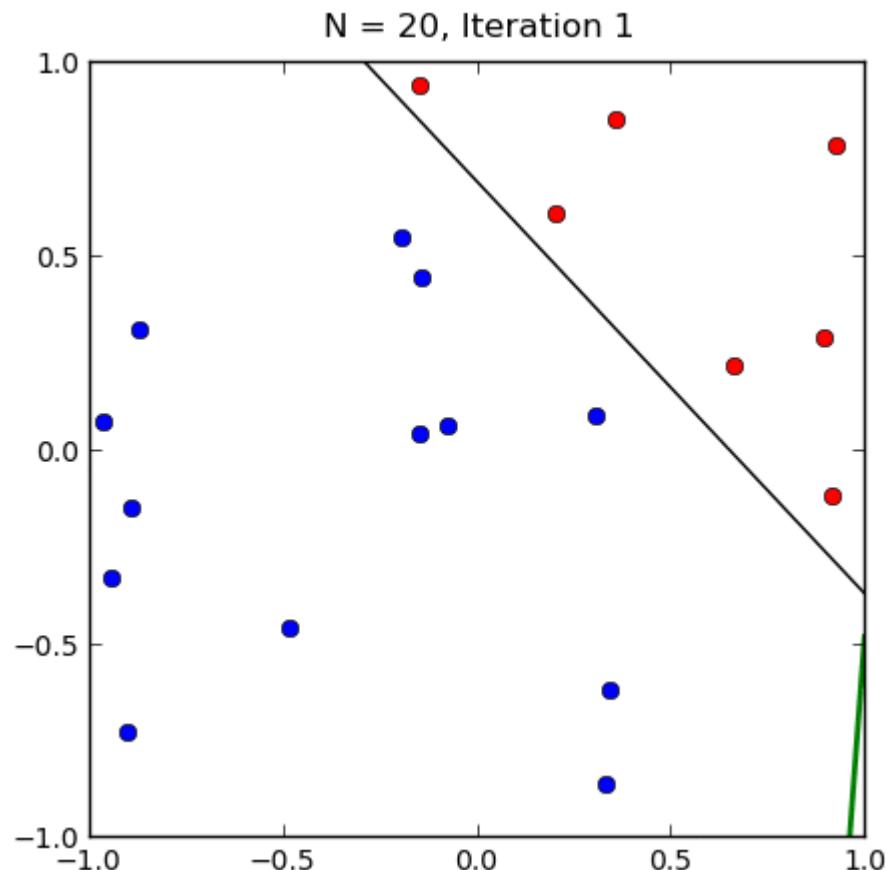
Data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$

Алгоритм:

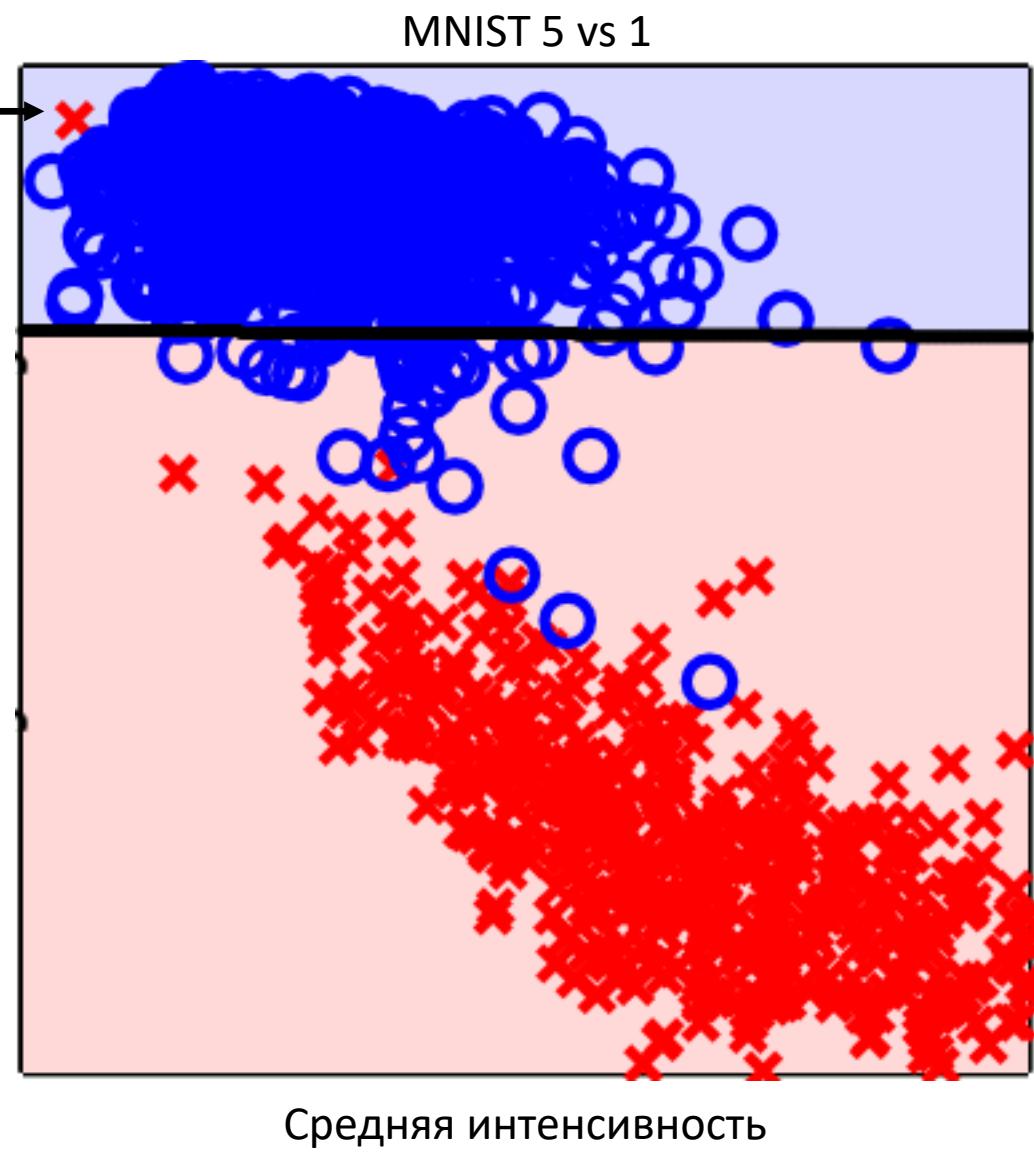
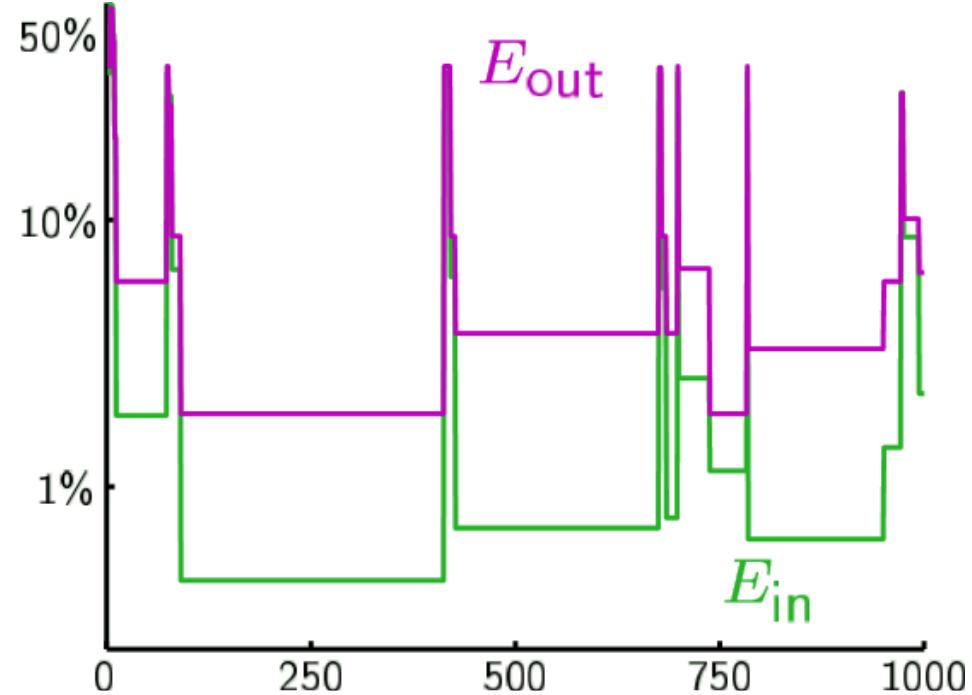
Начнем со случайного вектора w

Найдем такой \mathbf{x}_i , что $h(\mathbf{x}_i) \neq y_i$

$$\mathbf{w} \leftarrow \mathbf{w} + y_i \mathbf{x}_i$$



Перцептрон с карманом



Подсчет ошибки

$$E_{in}(h) = \frac{1}{N} \sum_{i=1}^N e(h(\mathbf{x}_i), f(\mathbf{x}_i)) \quad \text{in sample (в выборке)}$$

$$E_{out}(h) = E_{\mathbf{x}}[e(h(\mathbf{x}), f(\mathbf{x}))] \quad \text{out of sample (вне выборки)}$$

Неравенство Хёфдинга

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-\epsilon^2 N}$$

Ошибка обобщения
Generalization error

Иногда ошибкой обобщения называют E_{out}

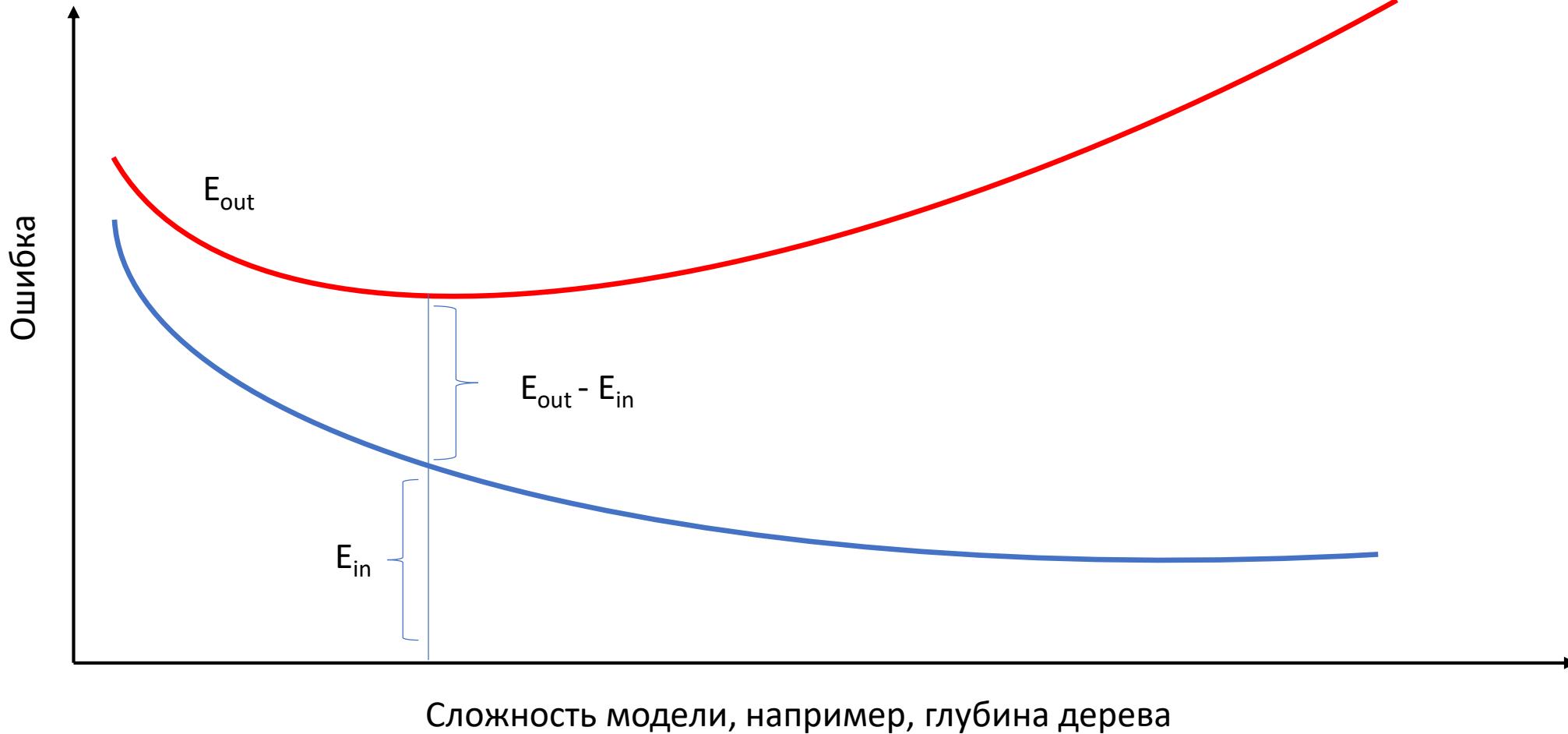
Неравенство Хёфдинга

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-\epsilon^2 N}$$

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq M2e^{-\epsilon^2 N}$$

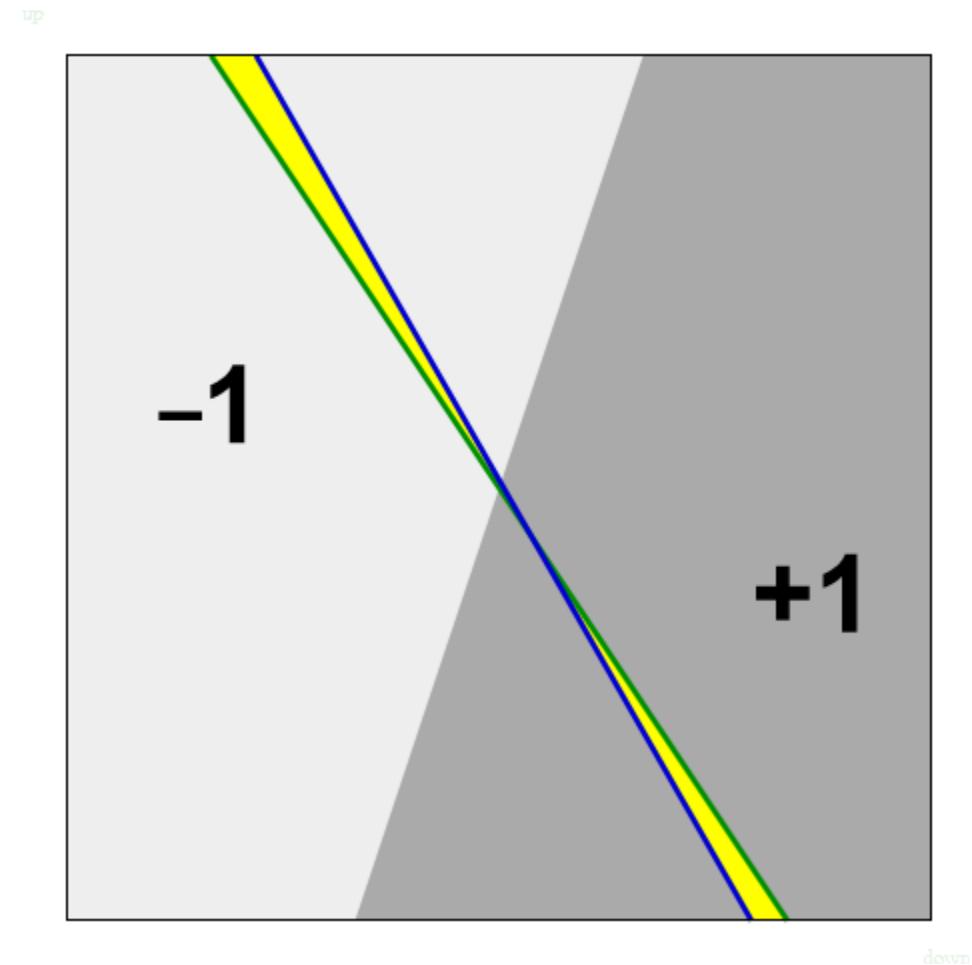
M – количество гипотез

Сложность модели



Близкие гипотезы

$$|E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| \approx |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)|$$



От гипотез к дихотомиям

- Гипотеза: $h: X \rightarrow \{-1, +1\}$
- Дихотомия: $h: \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \rightarrow \{-1, +1\}$
- Максимальное количество дихотомий: 2^N

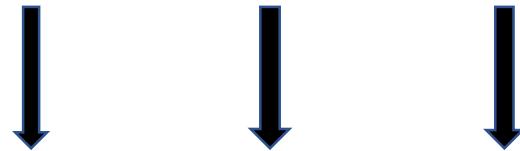
Функция роста (growth function)

$$m_H(N) = \max_{x_1, \dots, x_N} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

$$m_H(N) \leq 2^N$$

Неравенство Вапника-Червоненкиса

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq M 2e^{-\epsilon^2 N}$$



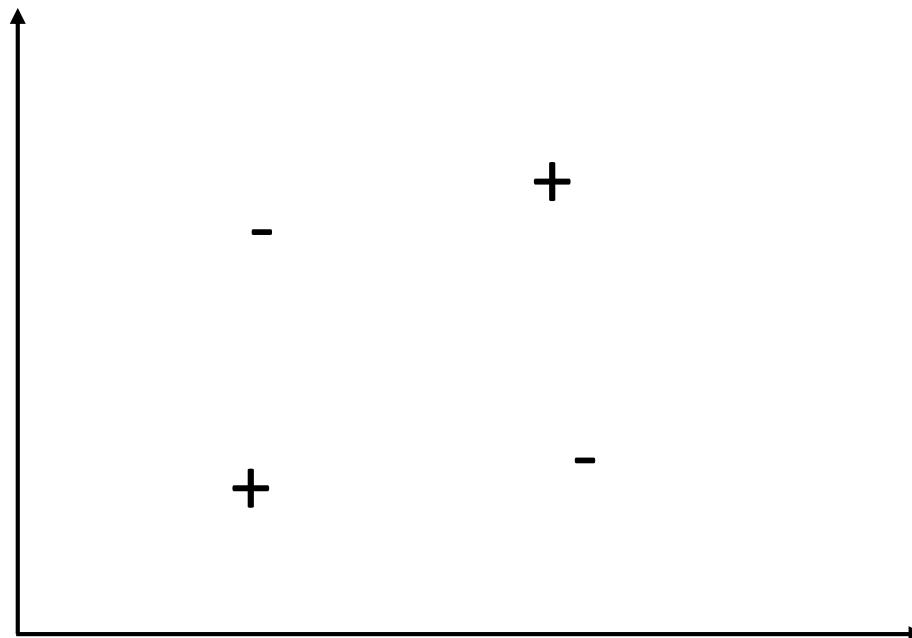
$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq m_H(2N) 4e^{-\frac{1}{8} \epsilon^2 N}$$

Точка “поломки” (Breakpoint)

Определение: если никакой набор данных размера k нельзя распределить на все возможные случаи набором гипотез H , то k – точка поломки для H .

$$\min(k: m_H(k) < 2^k)$$

Для 2D перцептона, $k = 4$.



Доказательство полиномиальности функции роста в присутствии точки поломки

- $B(N, k) = m_H(N)$ с точкой поломки k
- $B(N, k) = \alpha + 2\beta$
- $\alpha + \beta \leq B(N - 1, k)$
- $\beta \leq B(N - 1, k - 1)$
- $B(N, k) \leq B(N - 1, k) + B(N - 1, k - 1)$
- Докажем, что $B(N, k) \leq \sum_{i=0}^{k-1} C_N^i$

	x_1	x_2	\dots	x_{N-1}	x_N
α	+1	+1	\dots	+1	+1
	-1	+1	\dots	+1	-1
	\vdots	\vdots	\vdots	\vdots	\vdots
	+1	-1	\dots	-1	-1
	-1	+1	\dots	-1	+1
β	+1	-1	\dots	+1	+1
	-1	-1	\dots	+1	+1
	\vdots	\vdots	\vdots	\vdots	\vdots
	+1	-1	\dots	+1	+1
	-1	-1	\dots	-1	+1
β	+1	-1	\dots	+1	-1
	-1	-1	\dots	+1	-1
	\vdots	\vdots	\vdots	\vdots	\vdots
	+1	-1	\dots	+1	-1
	-1	-1	\dots	-1	-1

1 вариант для x_N

2 варианта для x_N

Индукция

$$B(N, k) \leq \sum_{i=0}^{k-1} C_N^i$$

$$B(N, 1) = 1, \quad B(1, k > 1) = 2$$

$$B(N, k) \leq B(N - 1, k) + B(N - 1, k - 1) \leq \sum_{i=0}^{k-1} C_{N-1}^i + \sum_{i=0}^{k-2} C_{N-1}^i$$

$$= 1 + \sum_{i=1}^{k-1} C_N^i + \sum_{i=1}^{k-1} C_{N-1}^{i-1} = 1 + \sum_{i=1}^{k-1} (C_{N-1}^i + C_{N-1}^{i-1}) = 1 + \sum_{i=1}^{k-1} C_N^i = \sum_{i=0}^{k-1} C_N^i$$

Размерность Вапника-Червоненкиса

$d_{VC}(H)$ для набора гипотез H , это наибольшее значение N ,
для которого $m_H(N) = 2^N$.

$$d_{VC}(H) = k - 1, \text{ где } k - \text{ точка поломки}$$

Функция роста и VC-размерность

$$m_H(N) \leq \sum_{i=0}^{k-1} C_N^i$$

$$m_H(N) \leq \sum_{i=0}^{d_{VC}} C_N^i \leq N^{d_{VC}} + 1$$

VC-размерность для перцептрана

Для размерности d , $d_{VC} = d + 1$

$$\mathbf{X} = \begin{bmatrix} -\mathbf{x}_1^T - \\ -\mathbf{x}_2^T - \\ -\mathbf{x}_3^T - \\ \vdots \\ -\mathbf{x}_4^T - \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ & & & \vdots & & \\ 1 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

$\text{sign}(\mathbf{X}\mathbf{w}) = \mathbf{y}$

$\mathbf{X}\mathbf{w} = \mathbf{y}$

$\mathbf{w} = \mathbf{X}^{-1}\mathbf{y}$

$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_{d+1}, \mathbf{x}_{d+2}$

$$\mathbf{x}_j = \sum_{i \neq j} \mathbf{x}_i a_i \quad \mathbf{w}^T \mathbf{x}_j = \sum_{i \neq j} \mathbf{w}^T \mathbf{x}_i a_i \quad y_i = \text{sign}(a_i) \quad y_j = -1$$

Сколько нужно данных?

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq m_H(2N)4e^{-\epsilon^{\frac{1}{8}}N}$$

$$\approx N^{d_{VC}} e^{-N}$$

$$N \geq 10 d_{VC}$$

Валидация

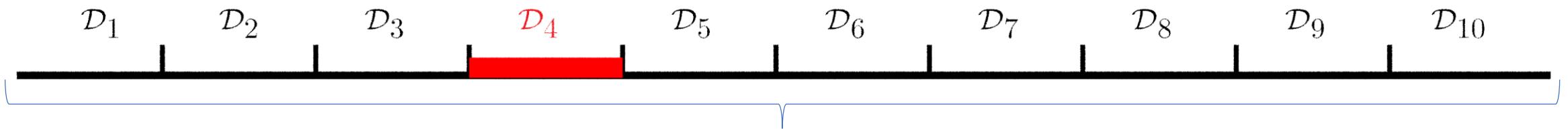
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_K, y_K) \in \mathcal{D}_{val}$$

$$E_{val}(h) = \frac{1}{K} \sum_{i=1}^K e(h(\mathbf{x}_i), f(\mathbf{x}_i))$$

$$P[|E_{val}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-\epsilon^2 N}$$

$$K = \frac{N}{5}$$

Кросс-валидация



D

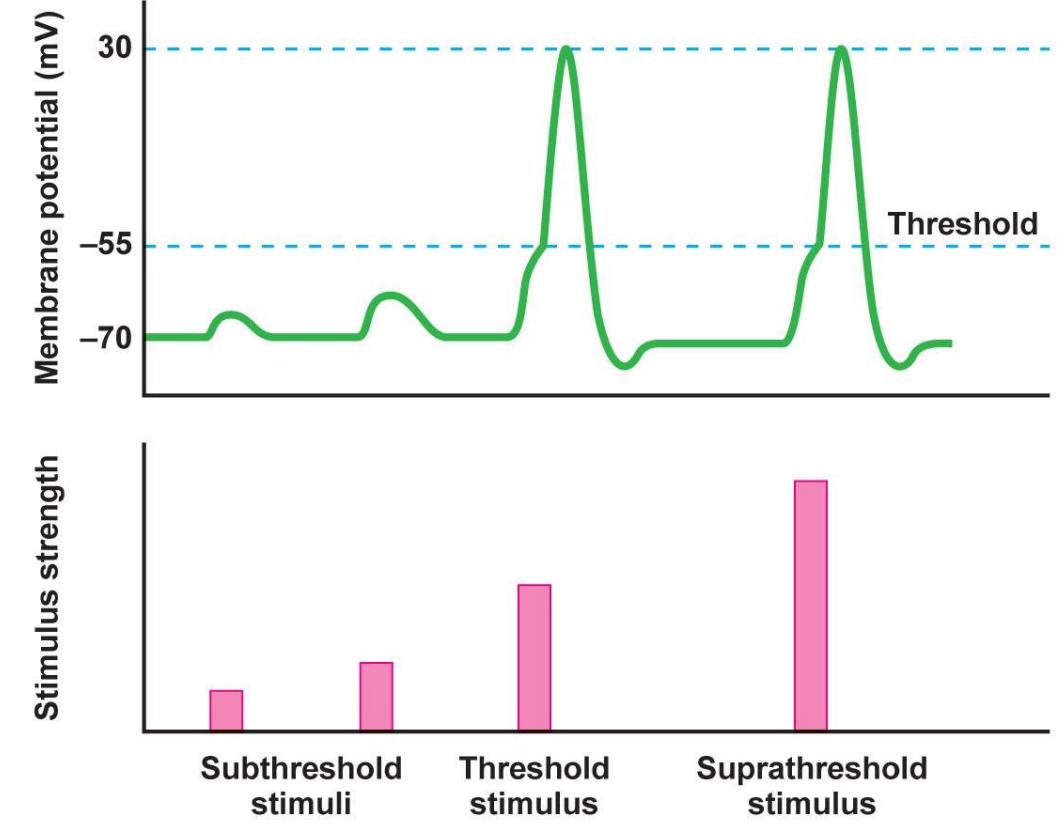
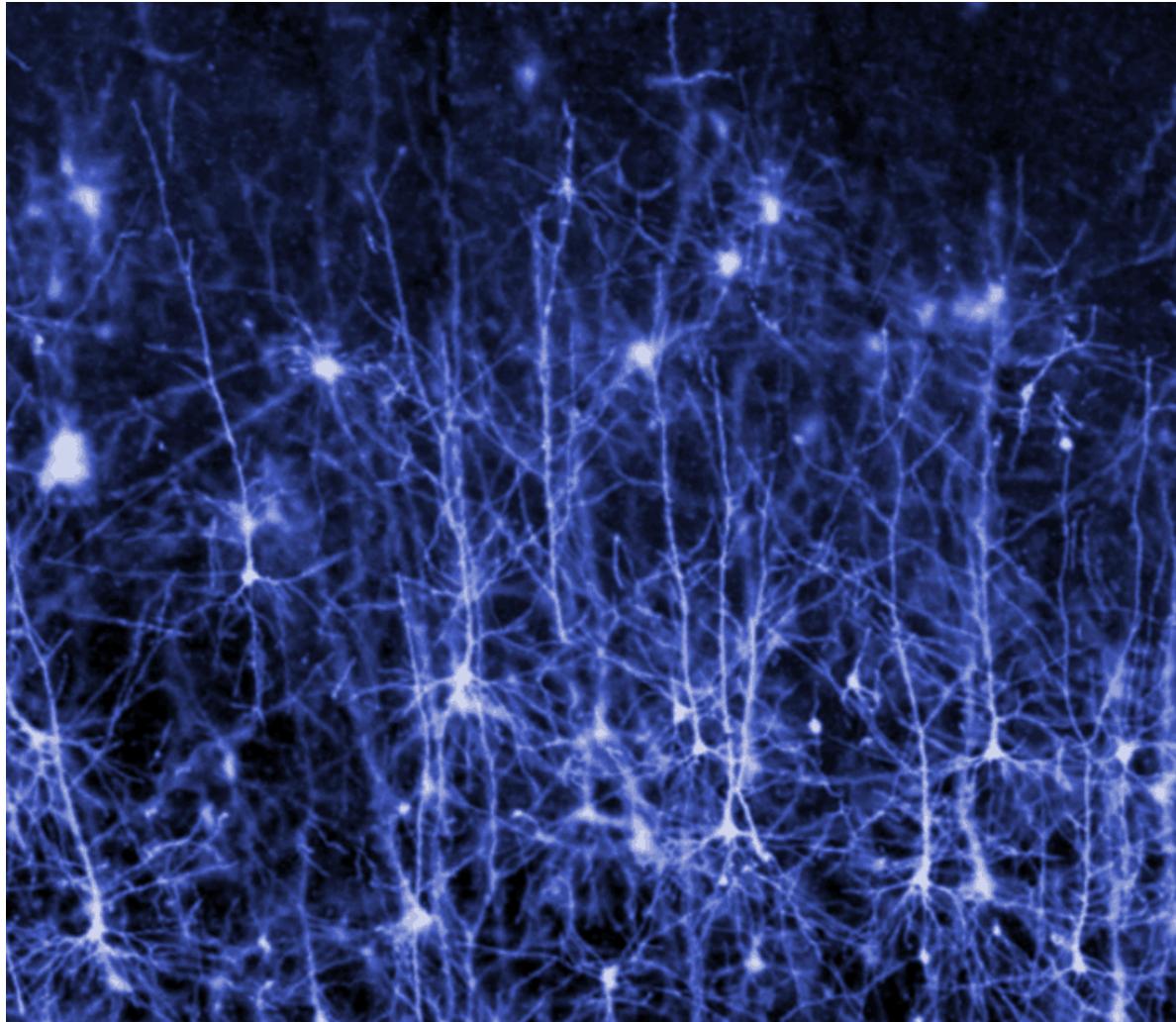
$$K = \frac{N}{10}$$

Train-Val-Test

- Обучаем алгоритм на **train**
- Оптимизируем алгоритм (гиперпараметры) на **val (cross-val)**
- Проверяем алгоритм на **test**

Нейронные сети

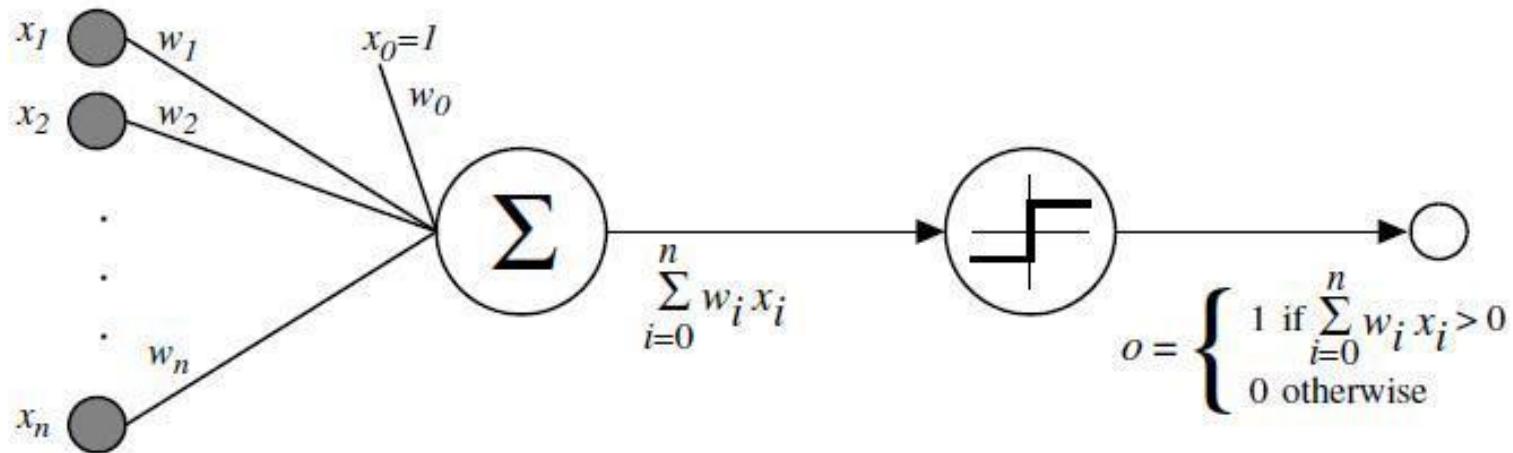
Нейронные сети



Принцип «все или ничего»

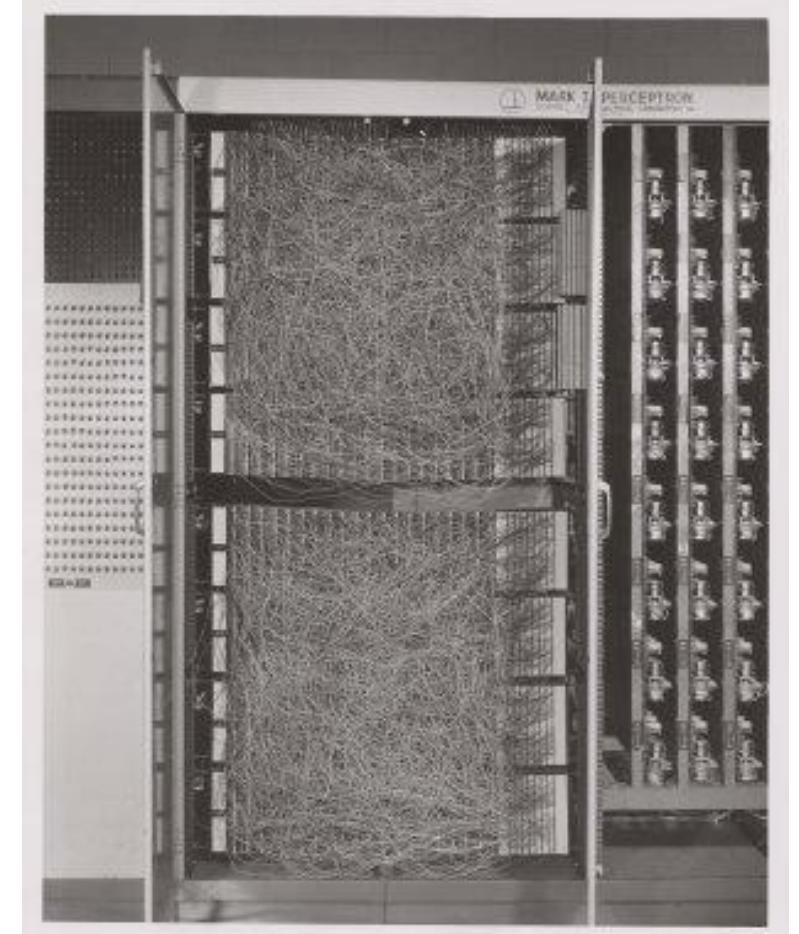
Нейронные сети

Перцептрон (Frank Rosenblatt, 1957)

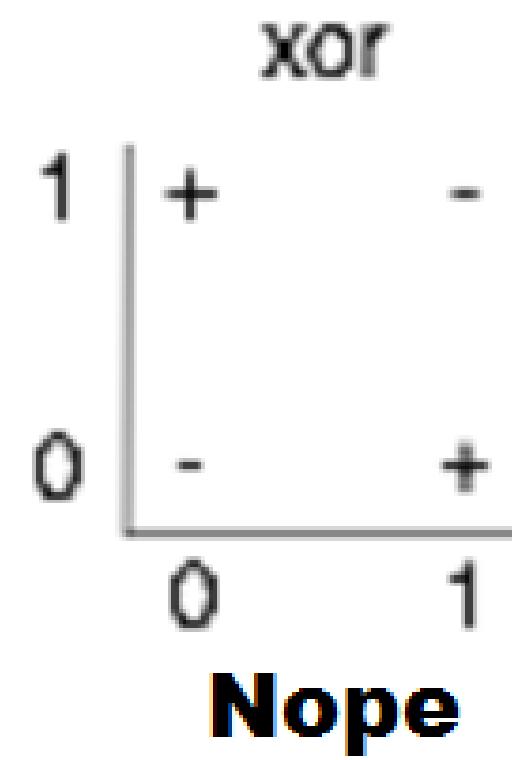
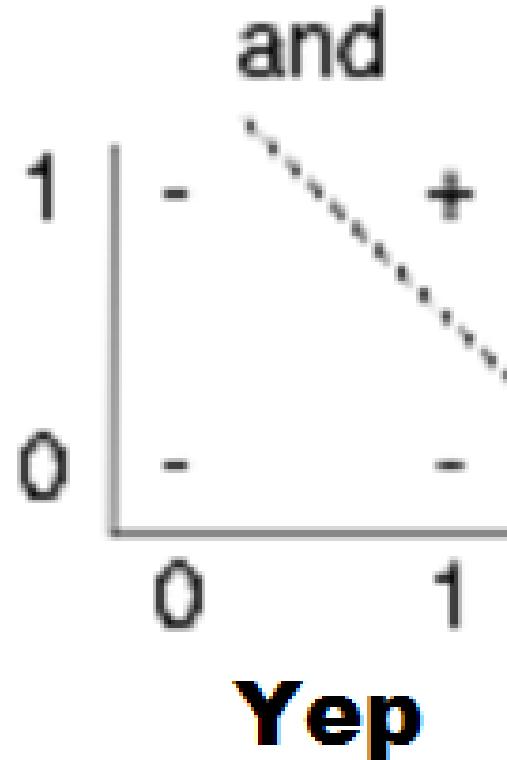
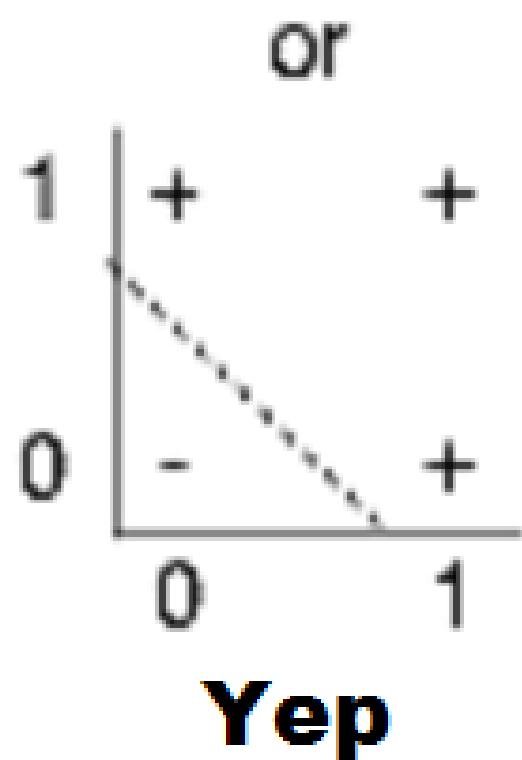


$$\text{sign}(\mathbf{w}^\top \mathbf{x}_n) \neq y_n$$

$$\mathbf{w} \leftarrow \mathbf{w} + y_n \mathbf{x}_n$$

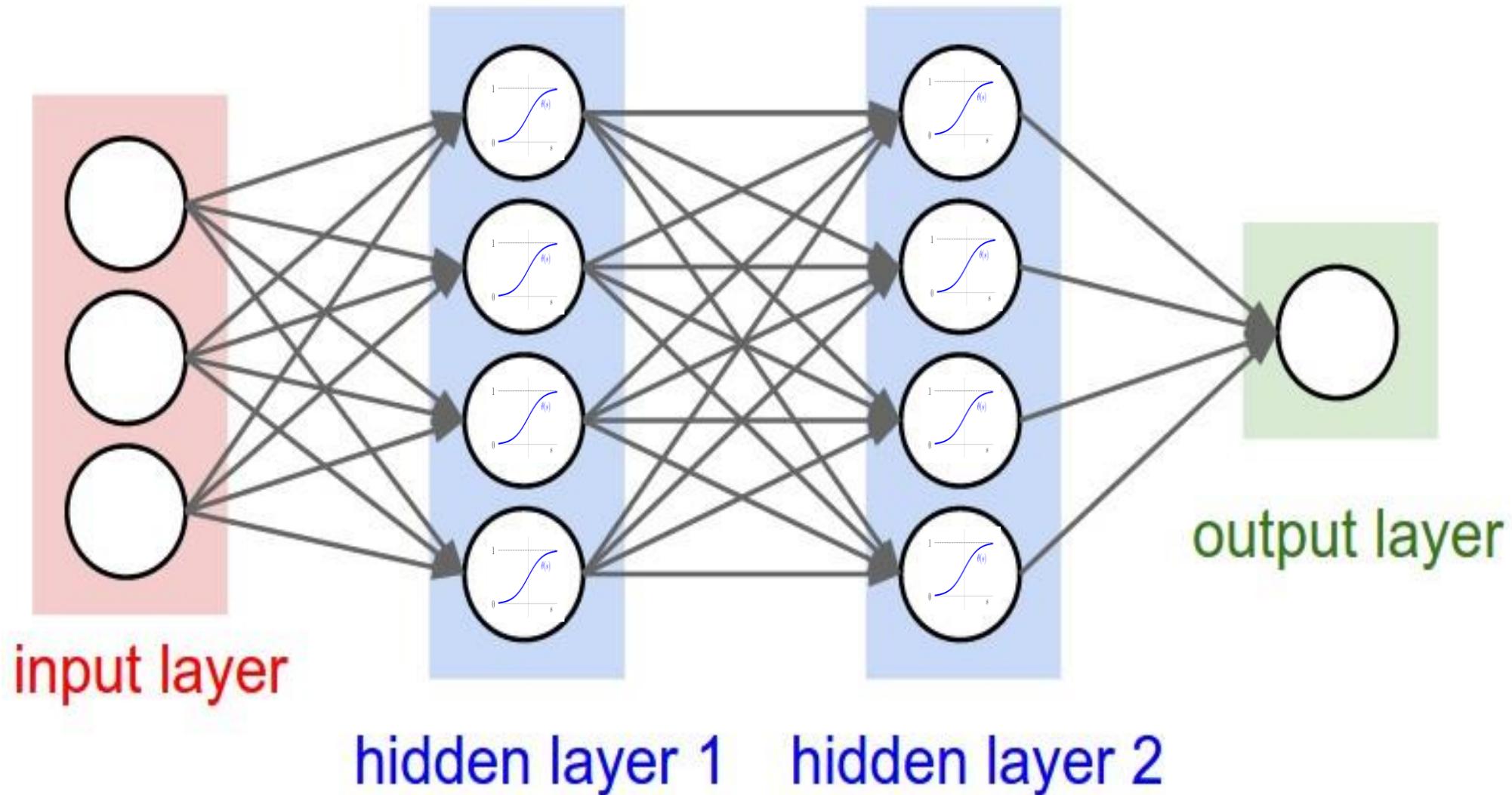


Главный недостаток Перцептронных нейронных сетей



Marvin Minsky and Seymour Papert, 1969

Нейронные сети с нелинейными преобразованиями

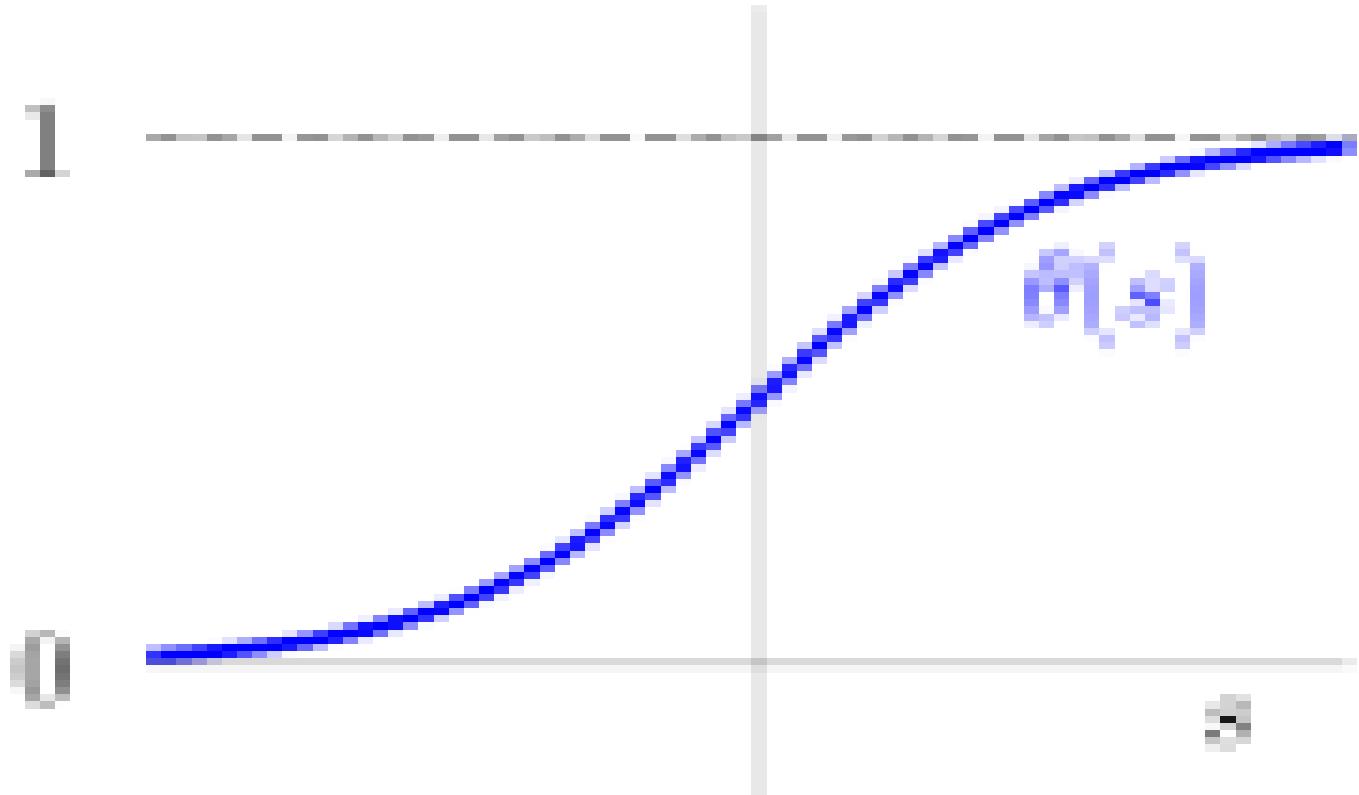


Логистическая функция

$$P(y | \mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{for } y = +1; \\ 1 - f(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

$$\sigma(s) = \frac{1}{1 + e^{-s}}$$

$$\sigma(-s) = 1 - \sigma(s)$$



Логистическая регрессия

$$P(y \mid \mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{for } y = +1; \\ 1 - f(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

$$\sigma(-s) = 1 - \sigma(s)$$

$$P(y \mid \mathbf{x}) = \sigma(y \mathbf{w}^\top \mathbf{x})$$

Правдоподобие:

$$\prod_{i=1}^N P(y_i \mid \mathbf{x}_i) = \prod_{i=1}^N \sigma(y_i \mathbf{w}^\top \mathbf{x}_i)$$

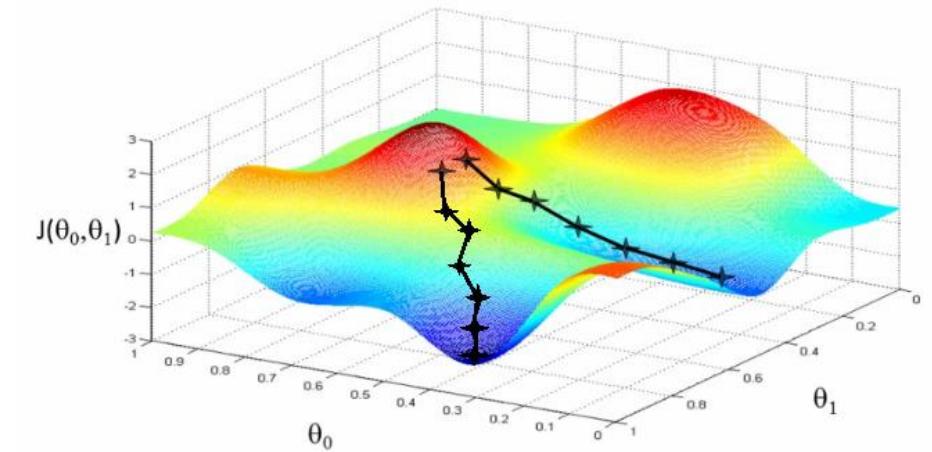
Функция потерь логистической регрессии

$$L(w) = -\frac{1}{N} \ln \left(\prod_{i=1}^N \sigma(y_i w^T x_i) \right) = \frac{1}{N} \sum_{i=1}^N \ln \left(\frac{1}{\sigma(y_i w^T x_i)} \right)$$

$$= \frac{1}{N} \sum_{i=1}^N \ln(1 + e^{-y_i w^T x_i})$$

Градиентный спуск (Gradient Descent)

$$w(t + 1) = w(t) - \eta \frac{\partial C(w)}{\partial w}$$



Стохастический градиентный спуск (SGD): $C(w)$ считается по малой части обучающей выборки (вплоть до одного примера).

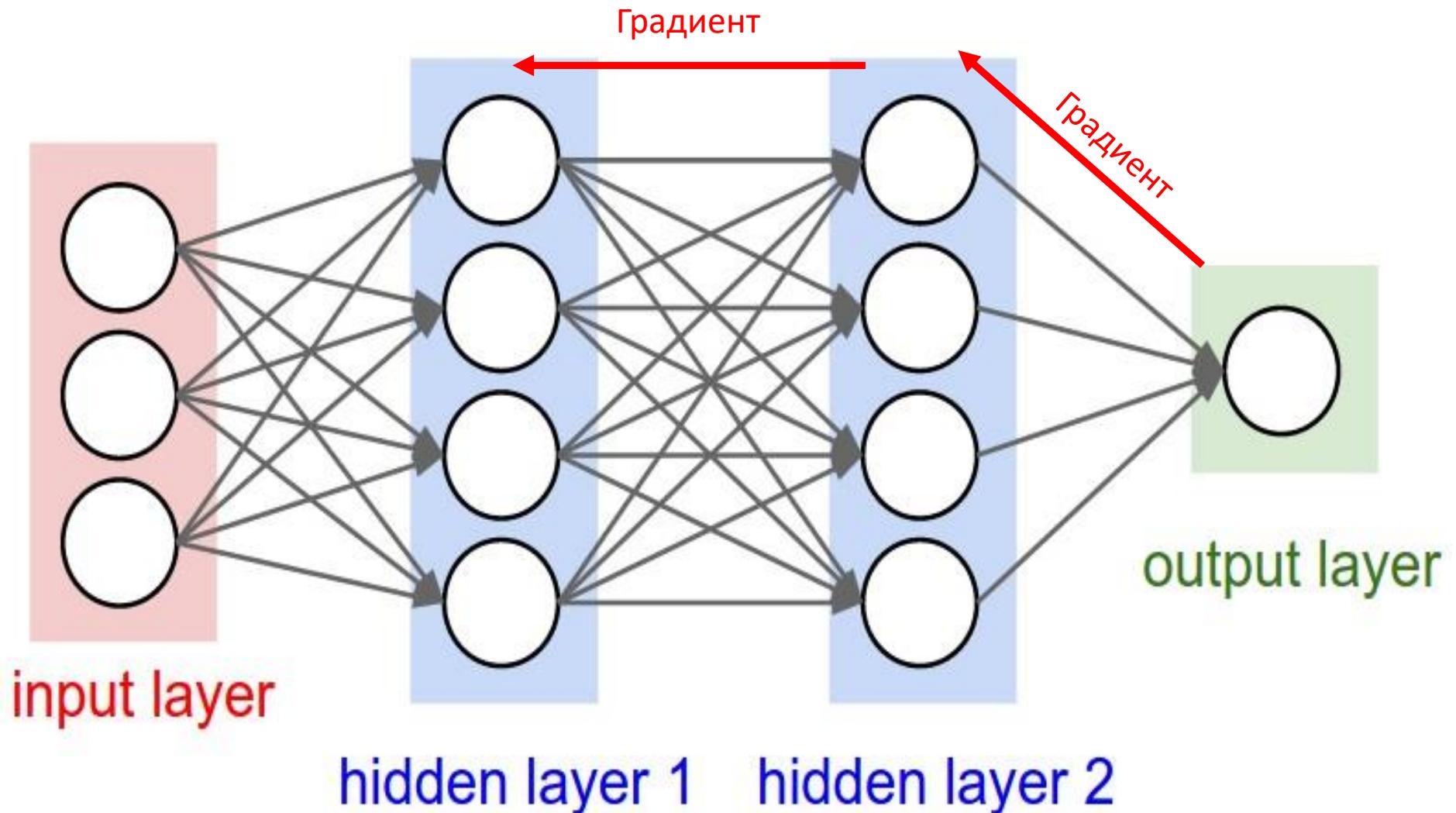
Градиентный спуск

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \left(-\frac{1}{N} \sum_{i=1}^N \frac{y_i \mathbf{x}_i}{1 + e^{y_i \mathbf{w}^\top \mathbf{x}_i}} \right)$$

Стохастический градиентный спуск

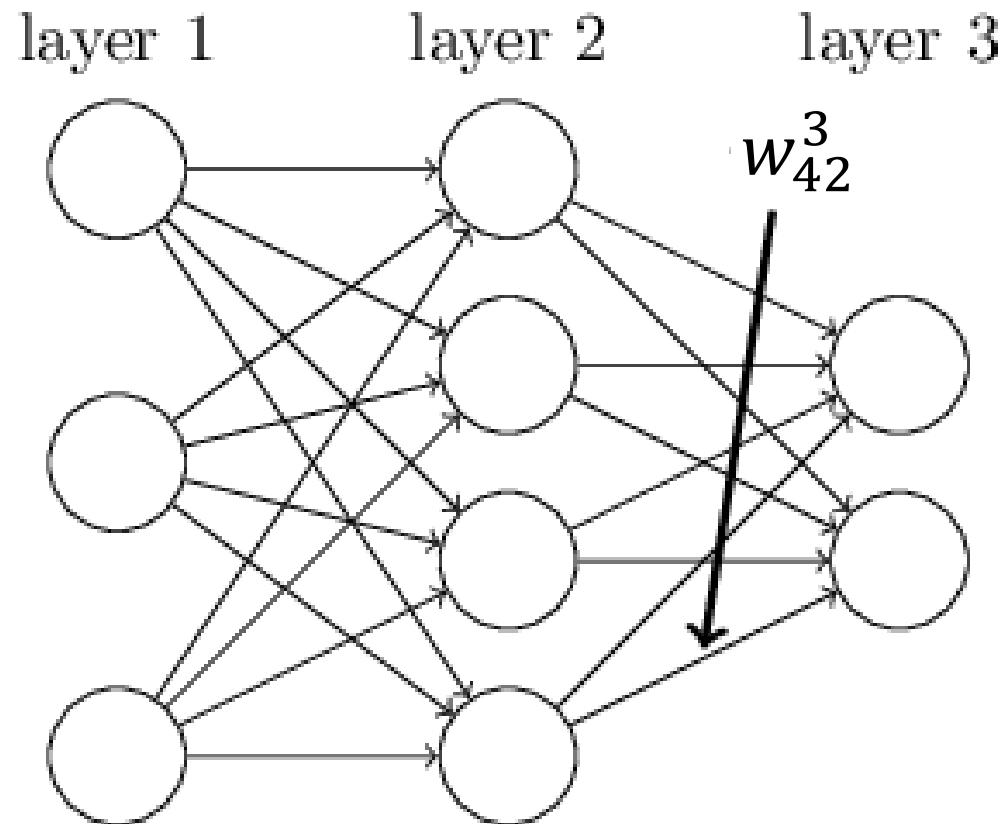
$$\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{\partial C(\mathbf{w}^\top \mathbf{x}_i, y_i)}{\partial \mathbf{w}}$$

Back-propagation



David Rumelhart, Geoffrey Hinton and Ronald Williams, 1986

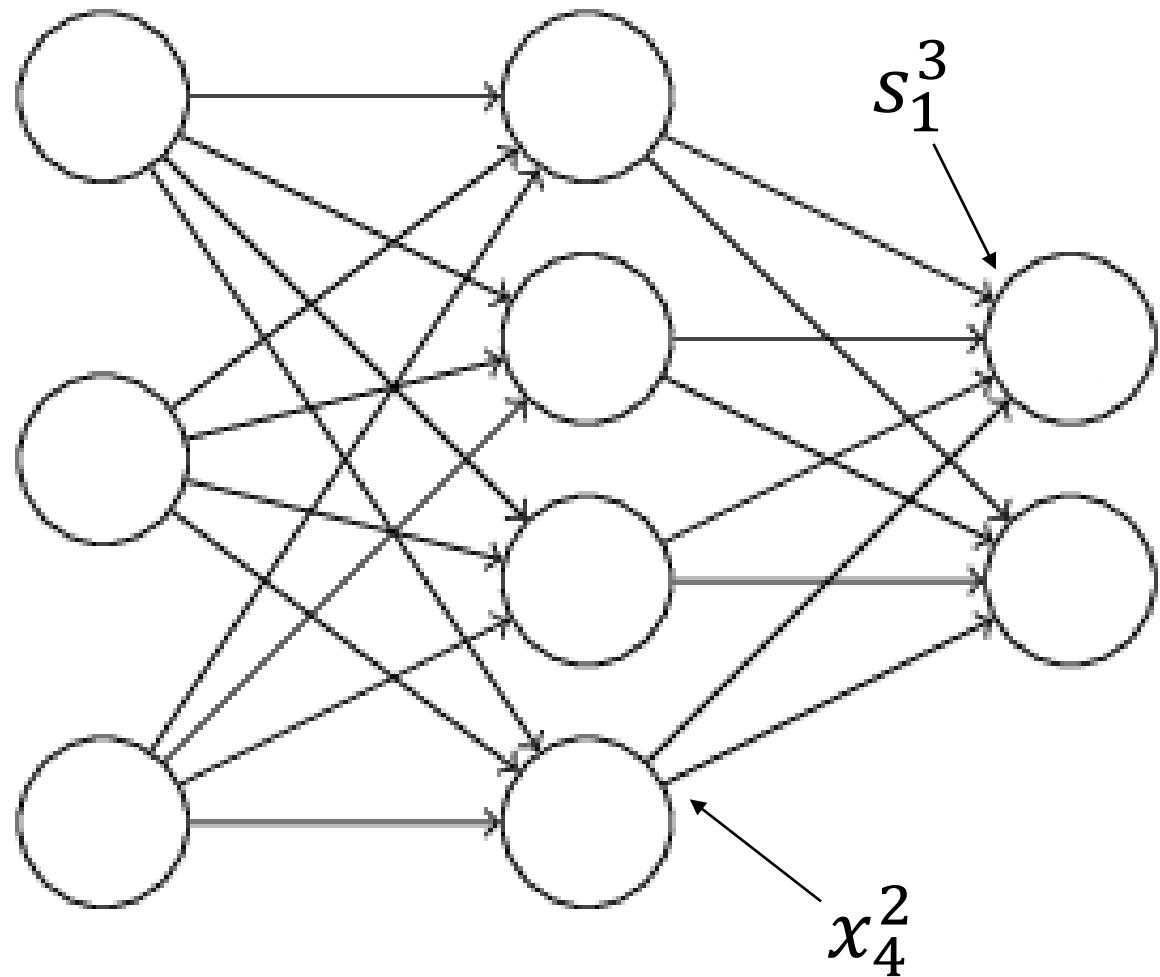
Back-propagation



w^l_{jk} - это коэффициент (вес) между j -ым нейроном слоя $l - 1$, и k -ым нейроном слоя l

Back-propagation

layer 1 layer 2 layer 3



s_j^l - поступающий «заряд» (activation) в j -ом нейроне слоя l на j -ом нейроне слоя l

$$s_j^l = \sum w_{ij}^l x_i^{l-1}$$

x_j^l - «заряд» после функции активации (activation function) в j -ом нейроне слоя l на j -ом нейроне слоя l

$$x_j^l = \sigma(s_j^l) = \sigma\left(\sum w_{ij}^l x_i^{l-1}\right)$$

$$x_j^1 = x_j$$

Back-propagation

Задача – оценить $\nabla C_{w_{ij}^l}(w)$: $\frac{\partial C(w)}{\partial w_{ij}^l} = \frac{\partial C(w)}{\partial s_j^l} \times \frac{\partial s_j^l}{\partial w_{ij}^l}$

$\frac{\partial s_j^l}{\partial w_{ij}^l} = x_i^{l-1}$ - посчитано $\frac{\partial C(w)}{\partial s_j^l} = \delta_j^l$ - посчитаем...

Back-propagation

Последний слой:

$$\delta_i^L = \frac{\partial C(w)}{\partial s_j^L}, \quad C(w) = f(x^L), \quad x_i^L = \sigma(s_i^L)$$

Предыдущие слои:

$$\delta_i^{l-1} = \frac{\partial C(w)}{\partial s_j^{l-1}} = \sum \frac{\partial C(w)}{\partial s_j^l} \times \frac{\partial s_j^l}{\partial x_i^{l-1}} \times \frac{\partial x_i^{l-1}}{\partial s_i^{l-1}} = \sum \delta_j^l \times w_{ij}^l \times \sigma'(s_i^{l-1})$$

Back-propagation

Алгоритм:

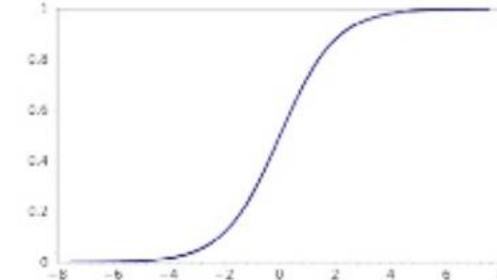
1. Инициализировать веса случайно*
2. Forward: Посчитать все x и s
3. Backward: Посчитать все δ
4. $w_{ij}^l \leftarrow w_{ij}^l - \eta x_i^{l-1} \delta_j^l$
5. $\rightarrow 2.$

Функции активации:

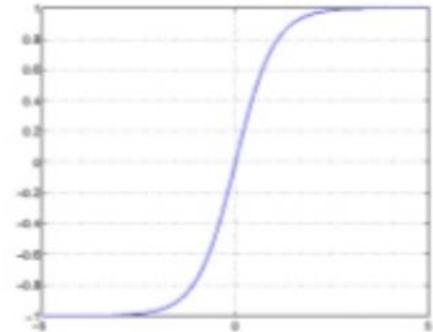
$$\text{Сигмоид: } \sigma(s) = \frac{1}{1+e^{-s}}$$

$$\text{Tanh: } \sigma(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

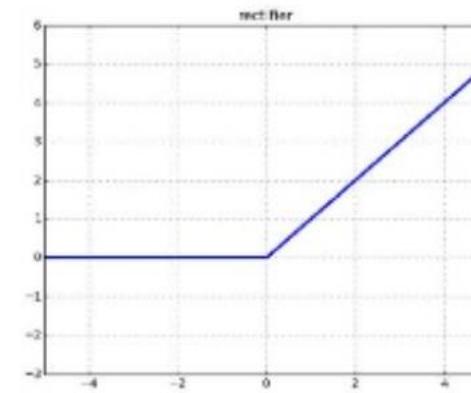
Rectified Linear Unit (ReLU):
 $\sigma(s) = \max(0, s)$



Sigmoid

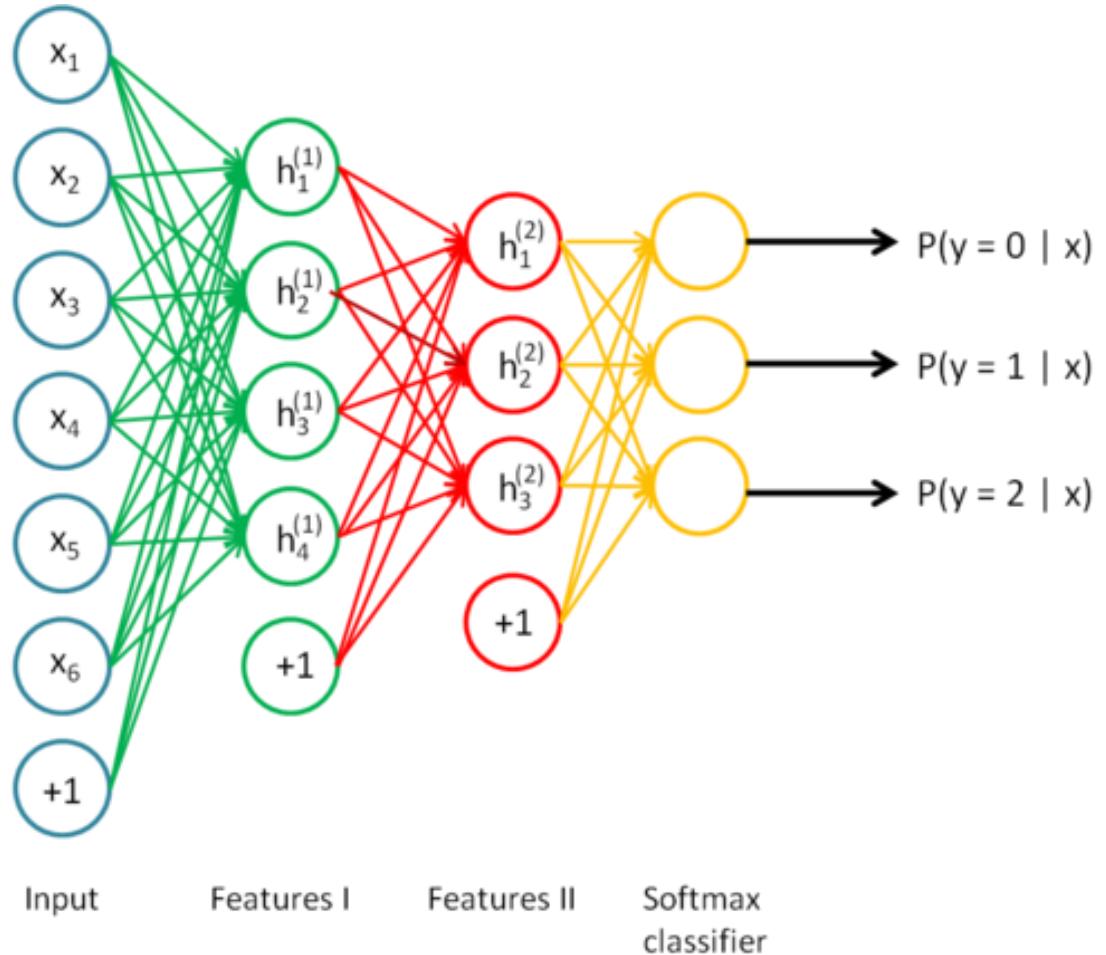


Tanh



ReLU

Функция потерь для мультиклассификации Softmax и Cross-Entropy



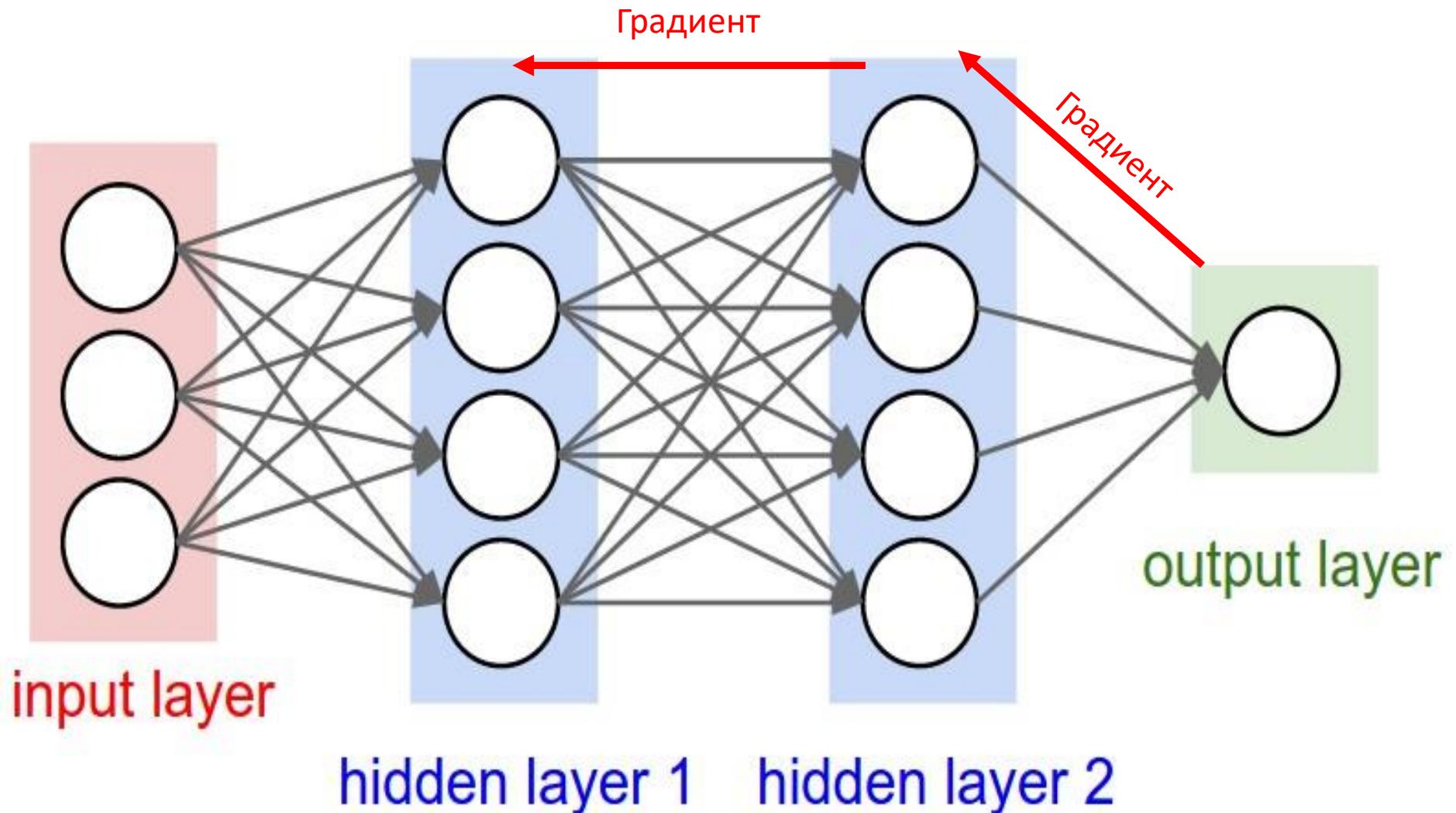
$$x_j^L = \sigma^L(s_j^L) = \frac{e^{s_j^L}}{\sum e^{s_i^L}}$$

$$C(w) = - \sum o_i \log(x_i^L)$$

$$o_i = \begin{cases} 1, & \text{если } y = i \\ 0, & \text{если } y \neq i \end{cases}$$

Deep Learning

Back-propagation



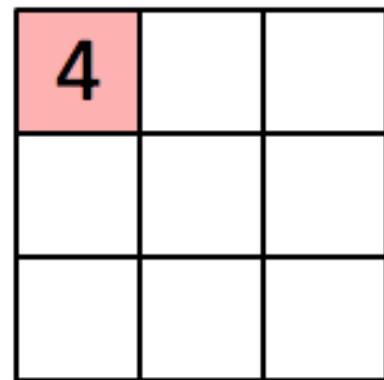
David Rumelhart, Geoffrey Hinton and Ronald Williams, 1986

Свертка (Convolution)

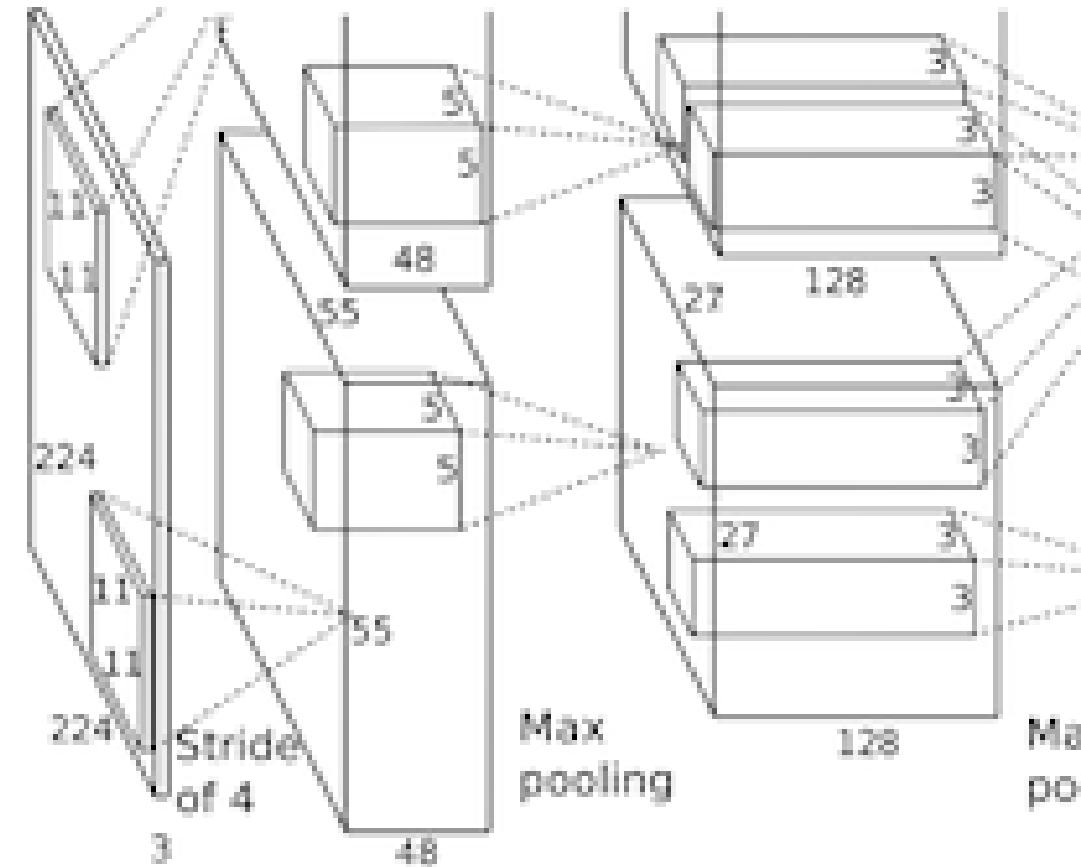
1 <small>x1</small>	1 <small>x0</small>	1 <small>x1</small>	0	0
0 <small>x0</small>	1 <small>x1</small>	1 <small>x0</small>	1	0
0 <small>x1</small>	0 <small>x0</small>	1 <small>x1</small>	1	1
0	0	1	1	0
0	1	1	0	0

Image

Kernel 3x3, Stride 1



Convolved Feature

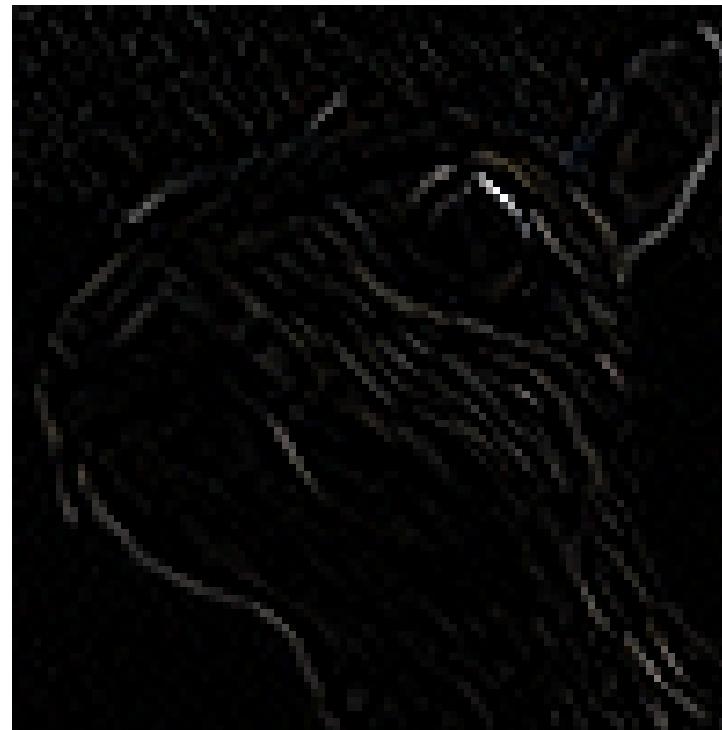
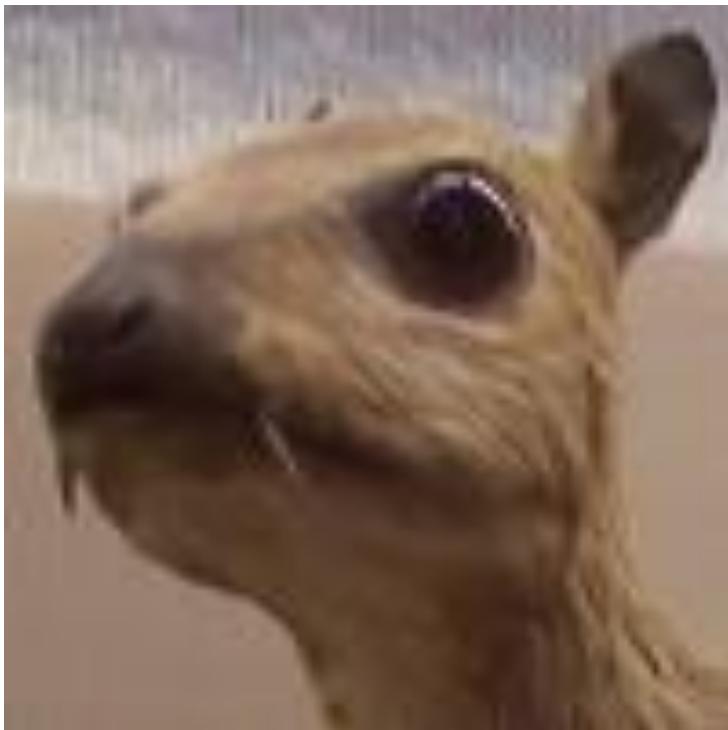


Примеры сверток

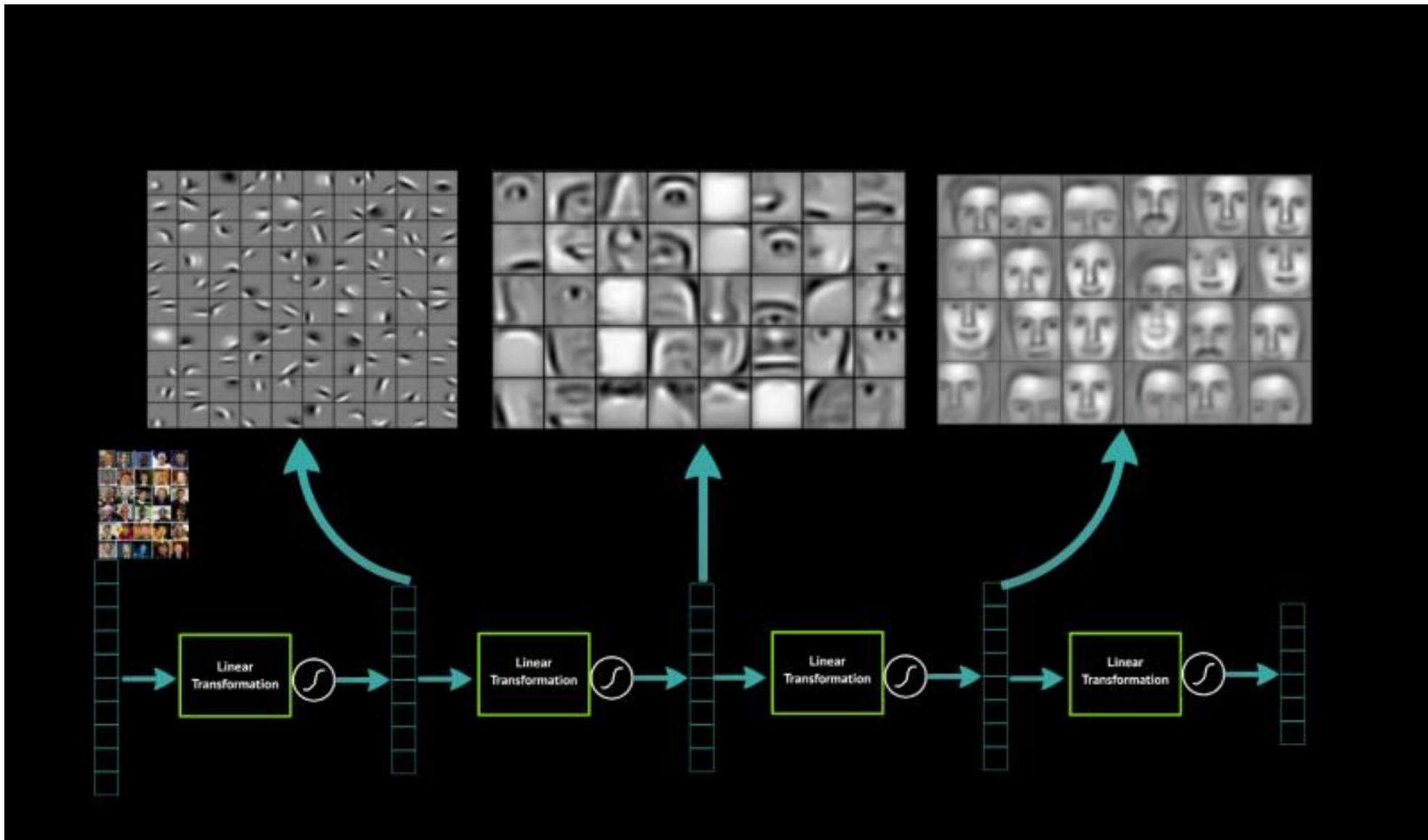
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Свертка (Convolution)



Pooling

Kernel 2x2

Stride 2

12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

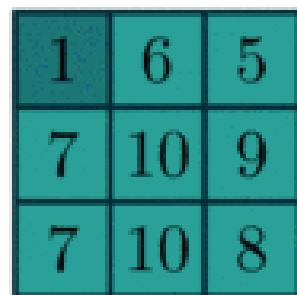
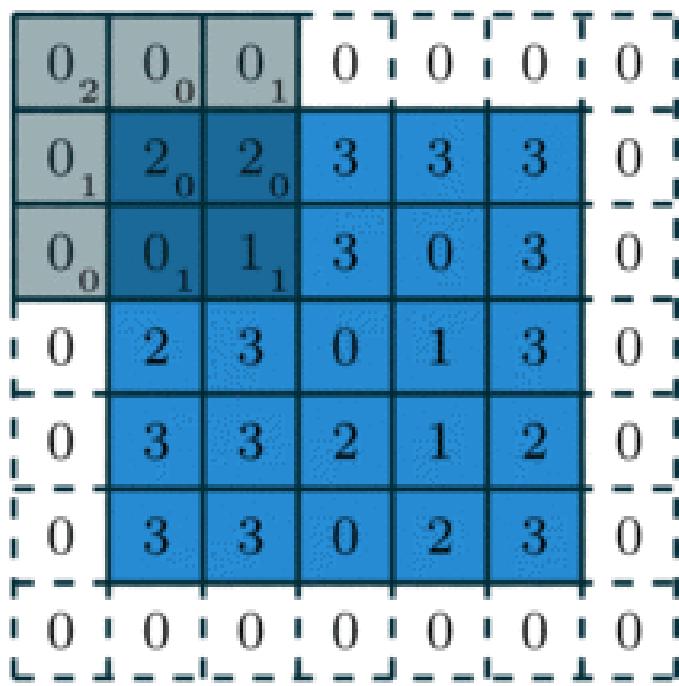
max pooling

20	30
112	37

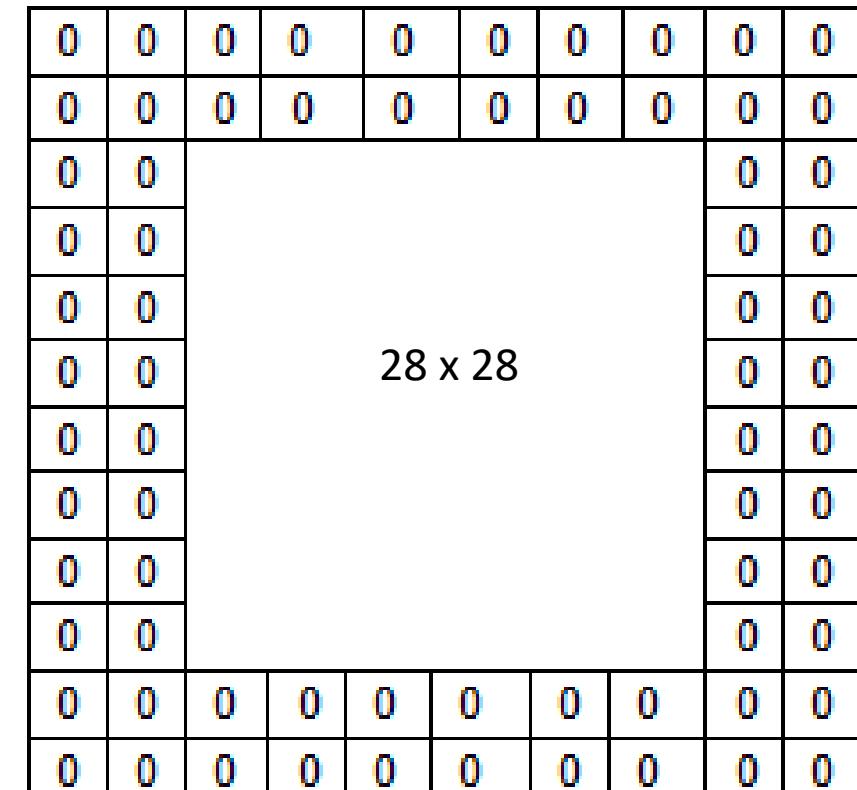
average pooling

13	8
79	20

Padding



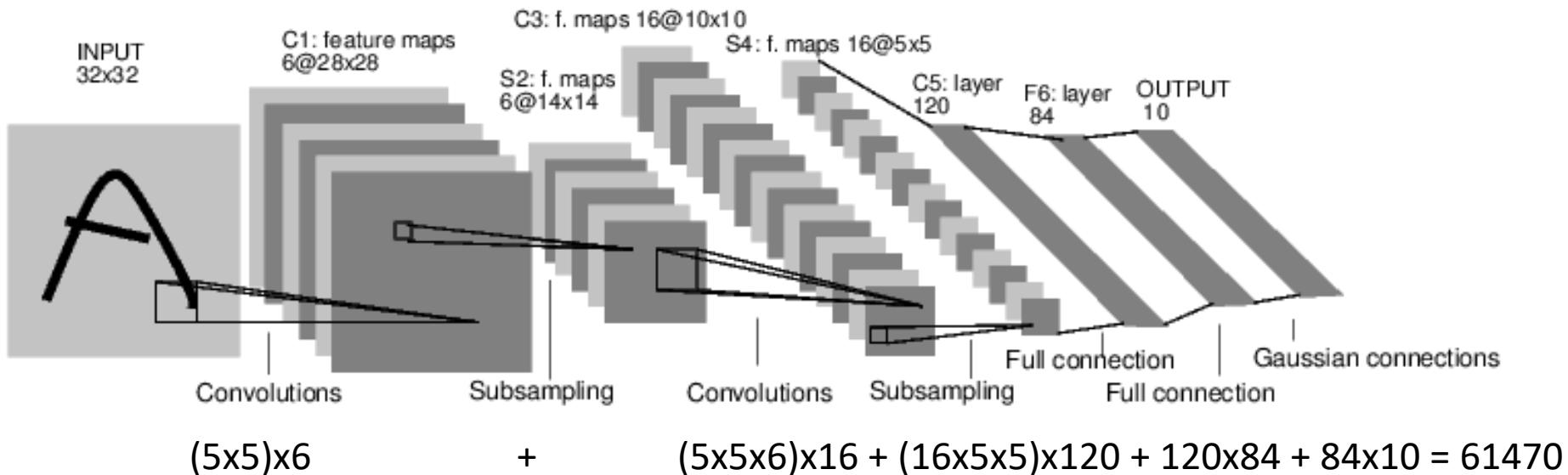
32



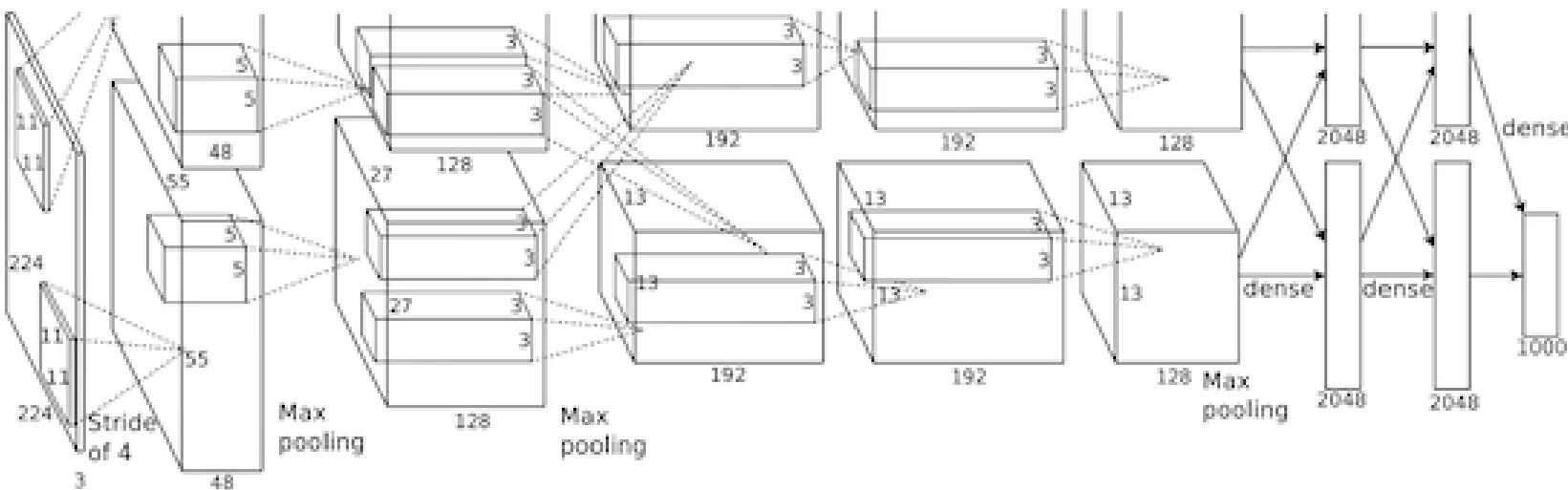
→

Deep learning

LeNet-5 (1998)



AlexNet (2012)



$$(11 \times 11 \times 3) \times 48 + (5 \times 5 \times 48) \times 128 + (3 \times 3 \times 128) \times 192 \times 2 + (3 \times 3 \times 192) \times 192 + (3 \times 3 \times 192) \times 128 + (13 \times 13 \times 128) \times 2048 \times 2 + 2048 \times 2048 \times 2 + 2048 \times 1000 = 100\,207\,632$$

Работа нейронной сети

Deep Visualization Toolbox

yosinski.com/deepvis

#deepvis



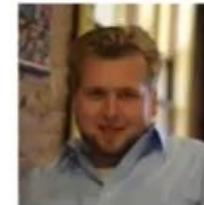
Jason Yosinski



Jeff Clune



Anh Nguyen



Thomas Fuchs



Hod Lipson



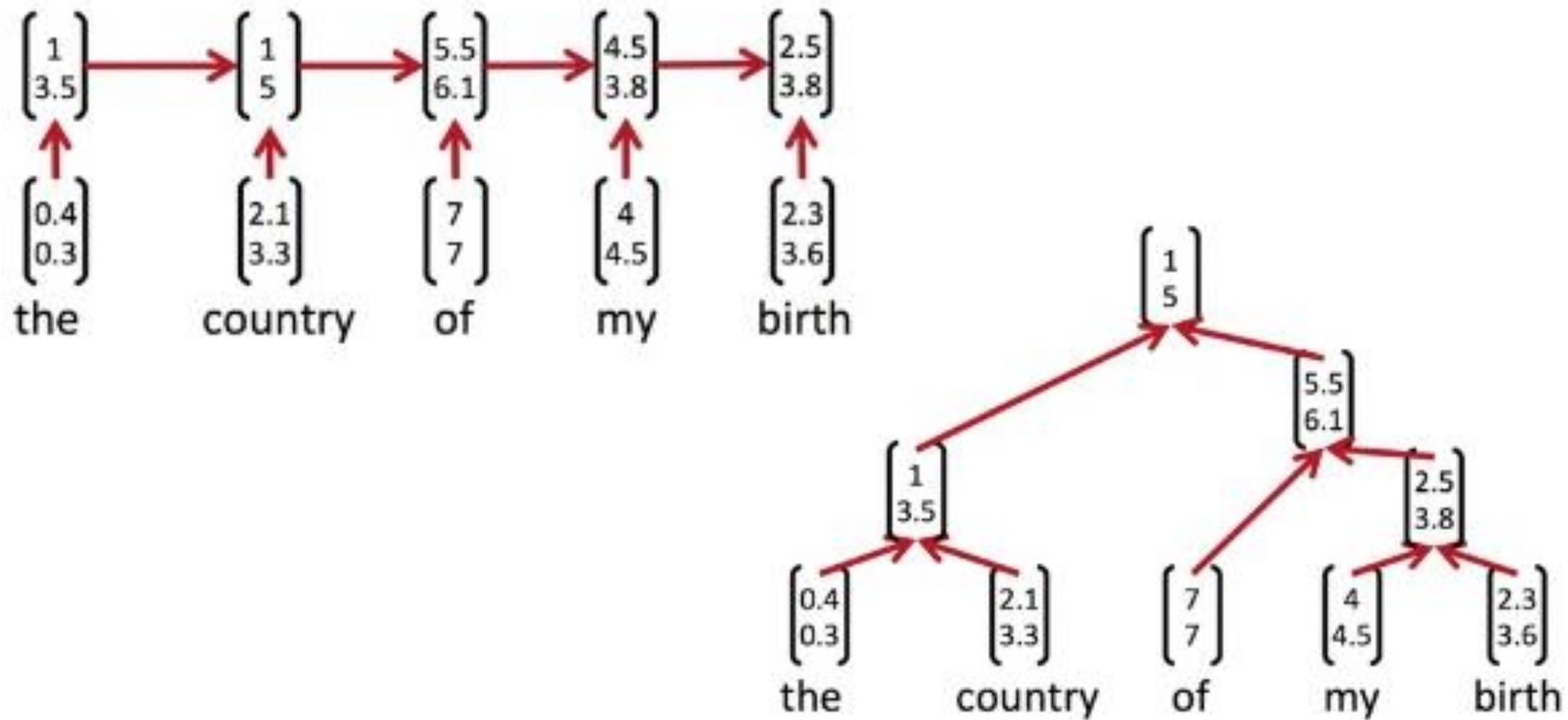
Cornell University

UNIVERSITY
OF WYOMING

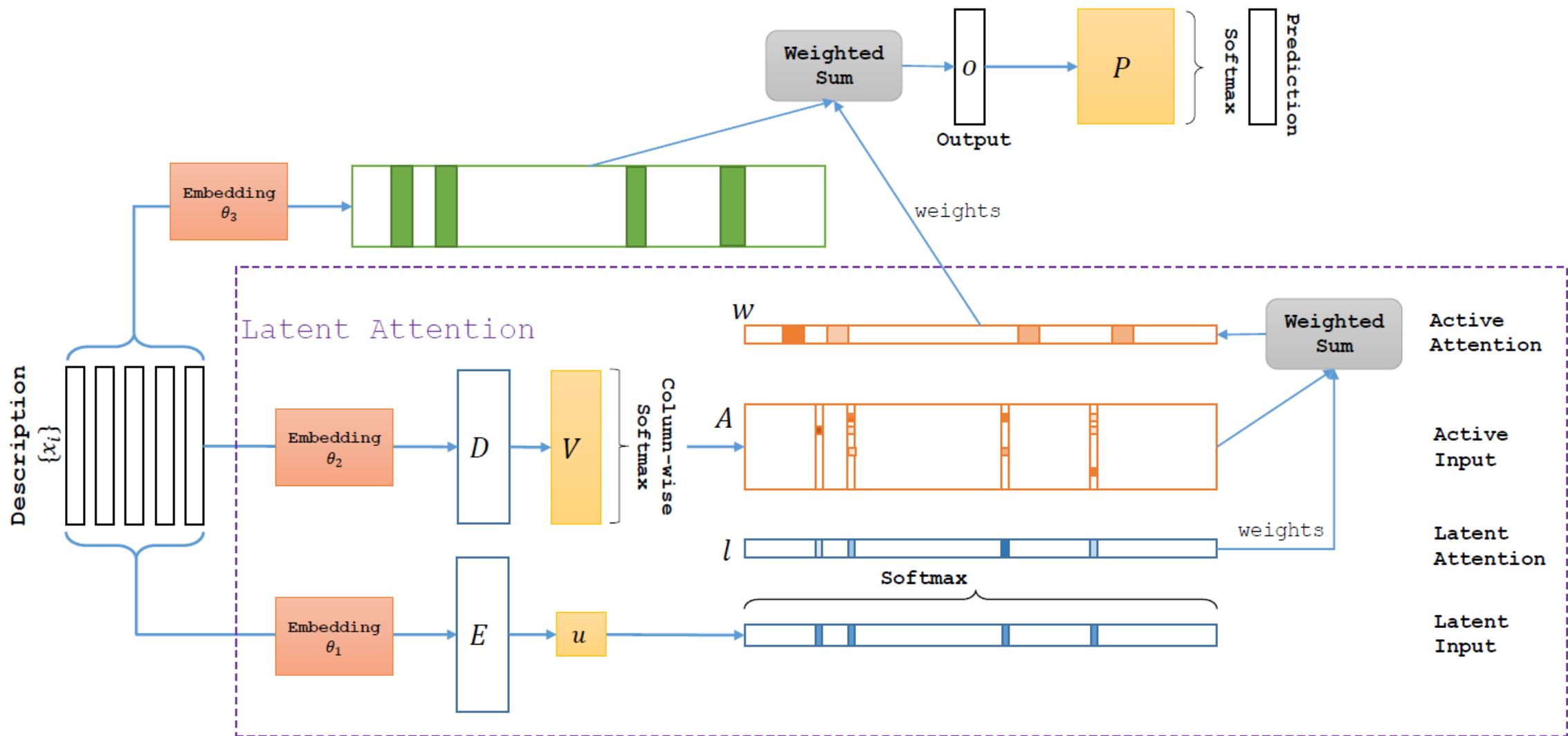


Jet Propulsion Laboratory
California Institute of Technology

Recurrent and Recursive Neural Networks



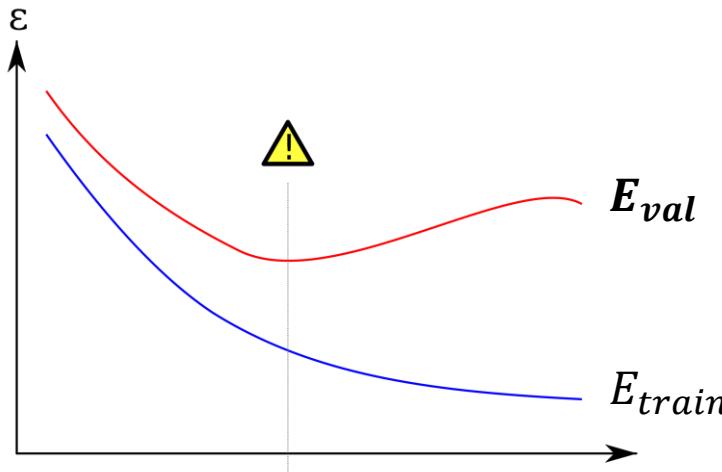
Complex DNN Architecture and Layers



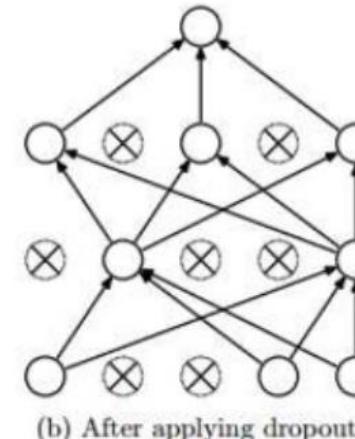
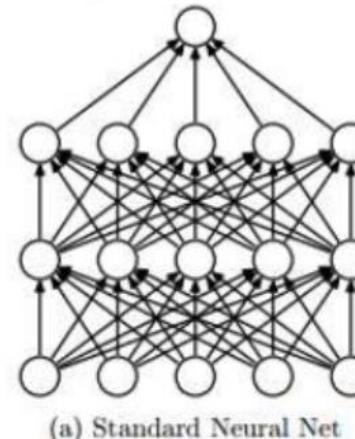
Регуляризация

$$\text{L2 регуляризация} \quad - \quad C'(w) = C(w) + \frac{\lambda}{2N} \|w\|_2^2$$

Ранняя остановка (Early Stopping):



Dropout:



Искусственное увеличение обучающей выборки (Artificial expansion)

Из-за большого количества гиперпараметров нужно перейти:

Train,Test \rightarrow Train, Validate, Test.

Подготовка к домашнему заданию

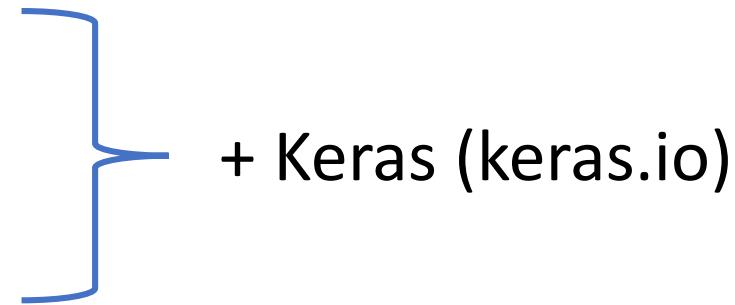
Пакеты deep learning:

Theano (deeplearning.net/software/theano)

TensorFlow (www.tensorflow.org)

Torch (torch.ch) - PyTorch

Deeplearning4j (deeplearning4j.org)

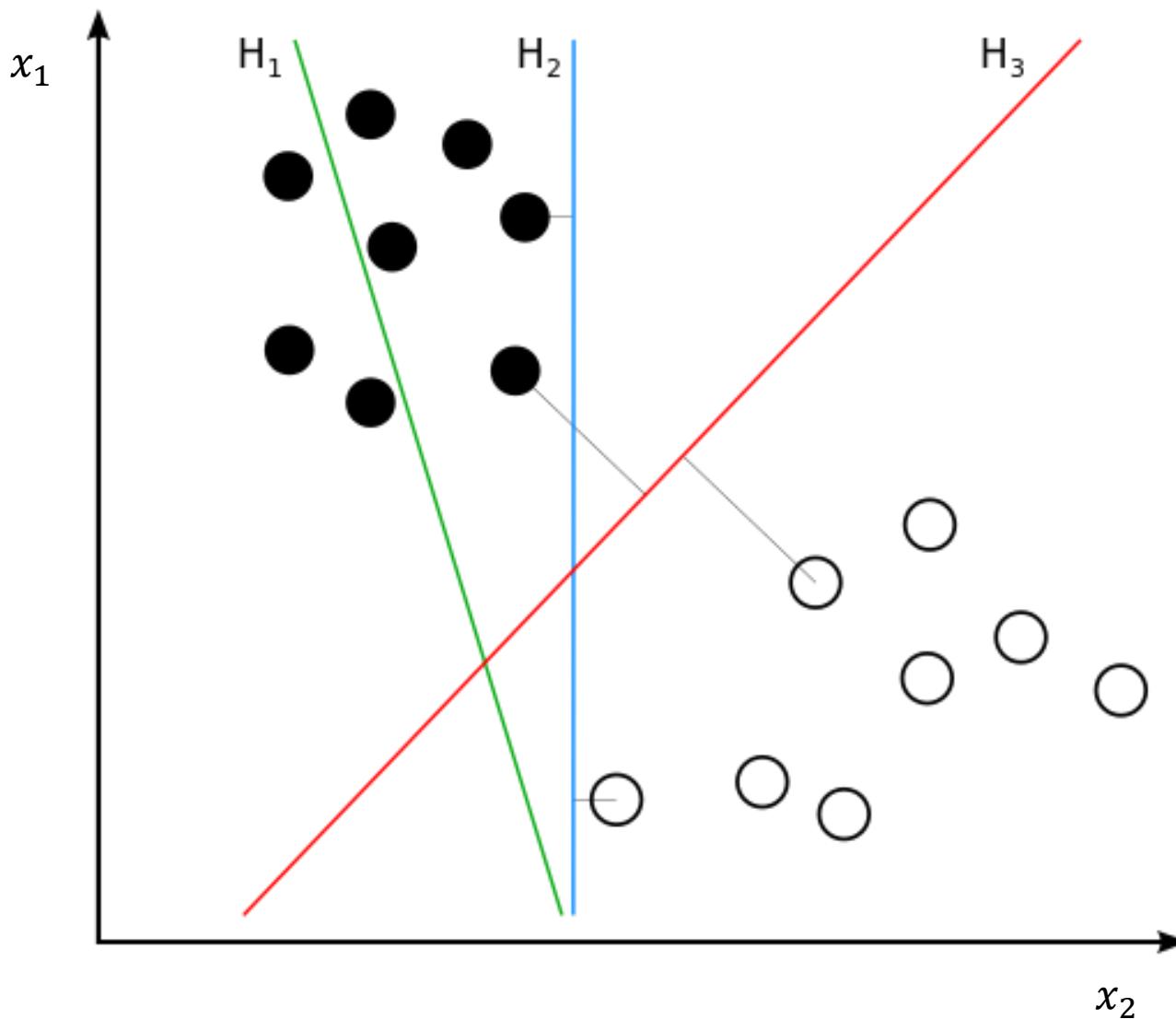


SVM

Support Vector Machines

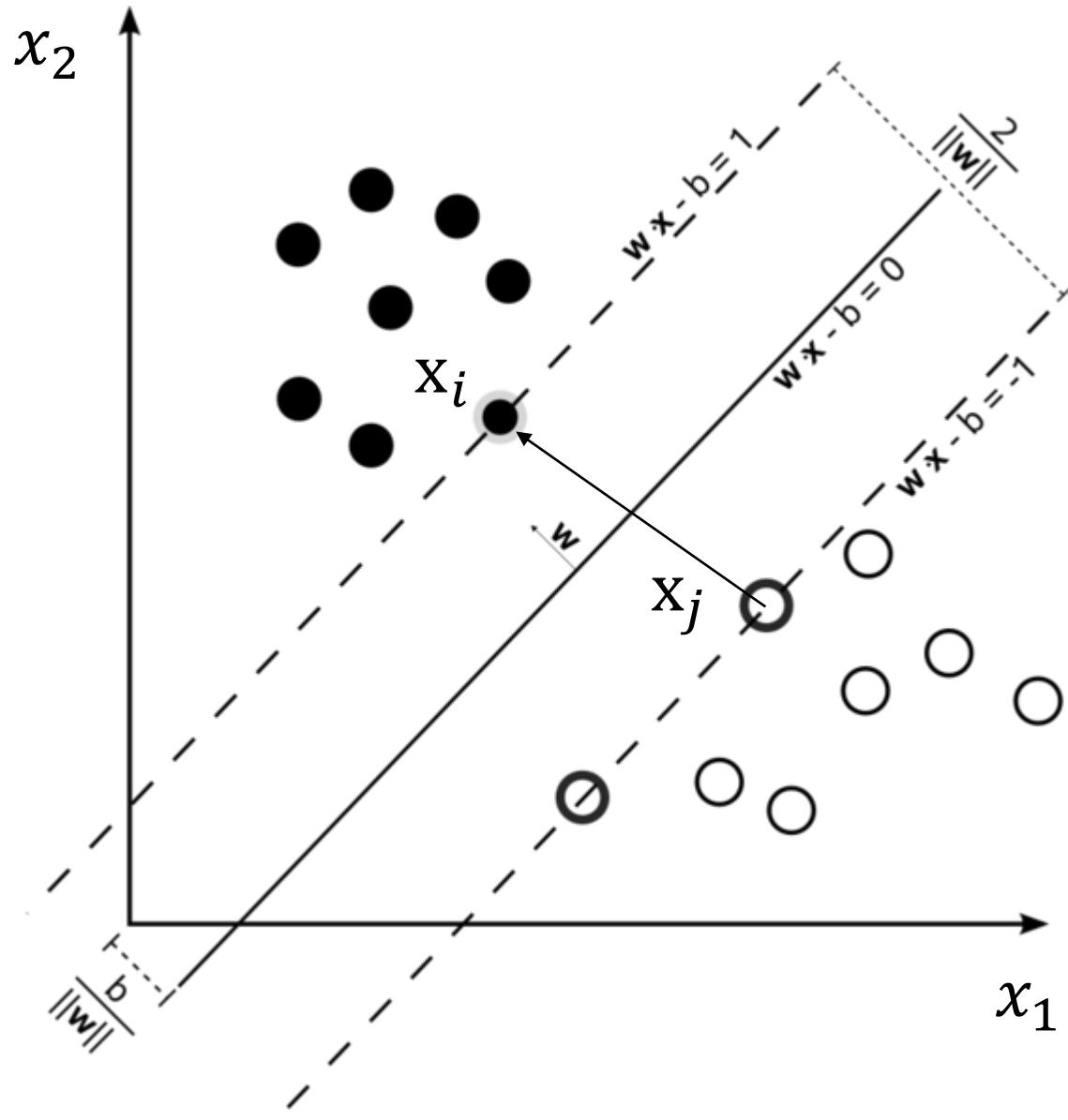
Метод Опорных Векторов

SVM (Support Vector Machines)



Максимизация отступа

Линейно разделимая выборка



Нормализуем w и b , так чтобы,

$$\min |w^T x_i - b| = \min (y_i(w^T x_i - b)) = 1,$$

Тогда максимальная ширина полосы –
проекция вектора $x_i - x_j$ на нормаль $\frac{w}{\|w\|}$, т.е:

$$\frac{w^T(x_i - x_j)}{\|w\|} = \frac{w^T x_i - b - (w^T x_j - b)}{\|w\|} = \frac{2}{\|w\|}$$

Тогда задача – это максимизация $\frac{2}{\|w\|}$

или минимизация $\|w\|$, или $w^T w$

при условии $y_i(w^T x_i - b) \geq 1$

Задача оптимизации

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

Теорема Каруша-Куна-Такера (ККТ)

Задача оптимизации с ограничениями:

$$\begin{cases} \min_z f(z) \\ g_i(z) \leq 0 \\ h_i(z) = 0 \end{cases}$$

Если z^* – точка локального минимума, то существуют множители Лагранджа α_i^* и β_j^* для:

$$\mathcal{L}(z, \alpha, \beta) = f(z) + \sum_{i=1}^m \alpha_i g_i(z) + \sum_{j=1}^k \beta_j h_j(z),$$

такие, что :

$$\begin{cases} \frac{\partial}{\partial z_i} \mathcal{L}(z^*, \alpha^*, \beta^*) = 0, \\ \frac{\partial}{\partial \beta_i} \mathcal{L}(z^*, \alpha^*, \beta^*) = 0, \\ \alpha_i g_i(z^*) = 0, \\ \alpha_i^* \geq 0 \end{cases}$$

И задачу можно свести к двойственной: $\max_{\alpha, \beta} \mathcal{L}(z, \alpha, \beta)$

Двойственная задача оптимизации

$$\begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ y_i(w^T x_i - b) \geq 1 \end{cases} \rightarrow \begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ -(y_i(w^T x_i - b) - 1) \leq 0 \end{cases}$$



$$\underbrace{\mathcal{L}(w, b, \alpha)}_{z} = \frac{1}{2} (w^T w) - \sum_{i=1}^N \alpha_i (y_i (w^T x_i - b) - 1);$$

Двойственная задача оптимизации

$$\begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ y_i(w^T x_i - b) \geq 1 \end{cases} \rightarrow \begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ -(y_i(w^T x_i - b) - 1) \leq 0 \end{cases}$$



$$\underbrace{\mathcal{L}(w, b, \alpha)}_{z} = \frac{1}{2} (w^T w) - \sum_{i=1}^N \alpha_i (y_i (w^T x_i - b) - 1);$$

$$\alpha_i \geq 0; \quad \alpha_i (y_i (w^T x_i - b) - 1) = 0$$

Решение двойственной задачи

$$\mathcal{L}(w, \alpha, b) = \frac{1}{2} (w^T w) - \sum_{i=1}^N \alpha_i (y_i (w^T x_i - b) - 1); \quad \alpha_i \geq 0; \quad \alpha_i (y_i (w^T x_i - b) - 1) = 0$$

$$\nabla_w \mathcal{L}(w, \alpha, b) = w - \sum_{i=1}^N \alpha_i y_i x_i = 0 \rightarrow w = \sum_{i=1}^N \alpha_i y_i x_i$$

$$\frac{\partial}{\partial b} \mathcal{L}(w, \alpha, b) = \sum_{i=1}^N \alpha_i y_i = 0$$

$$\begin{aligned} \mathcal{L}(w, \alpha, b) &= \frac{1}{2} (w^T w) - \sum_{i=1}^N \alpha_i (y_i (w^T x_i - b) - 1) = \frac{1}{2} \sum \sum y_i y_j \alpha_i \alpha_j x_i^T x_j - \sum \sum y_i y_j \alpha_i \alpha_j x_i^T x_j + \sum \alpha_i = \\ &= \mathcal{L}(w, \alpha, b) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N y_i y_j \alpha_i \alpha_j x_i^T x_j \end{aligned}$$

Алгоритм оптимизации – выбираем пару α_i, α_j и оптимизируем \mathcal{L} при остальных α фиксированных.

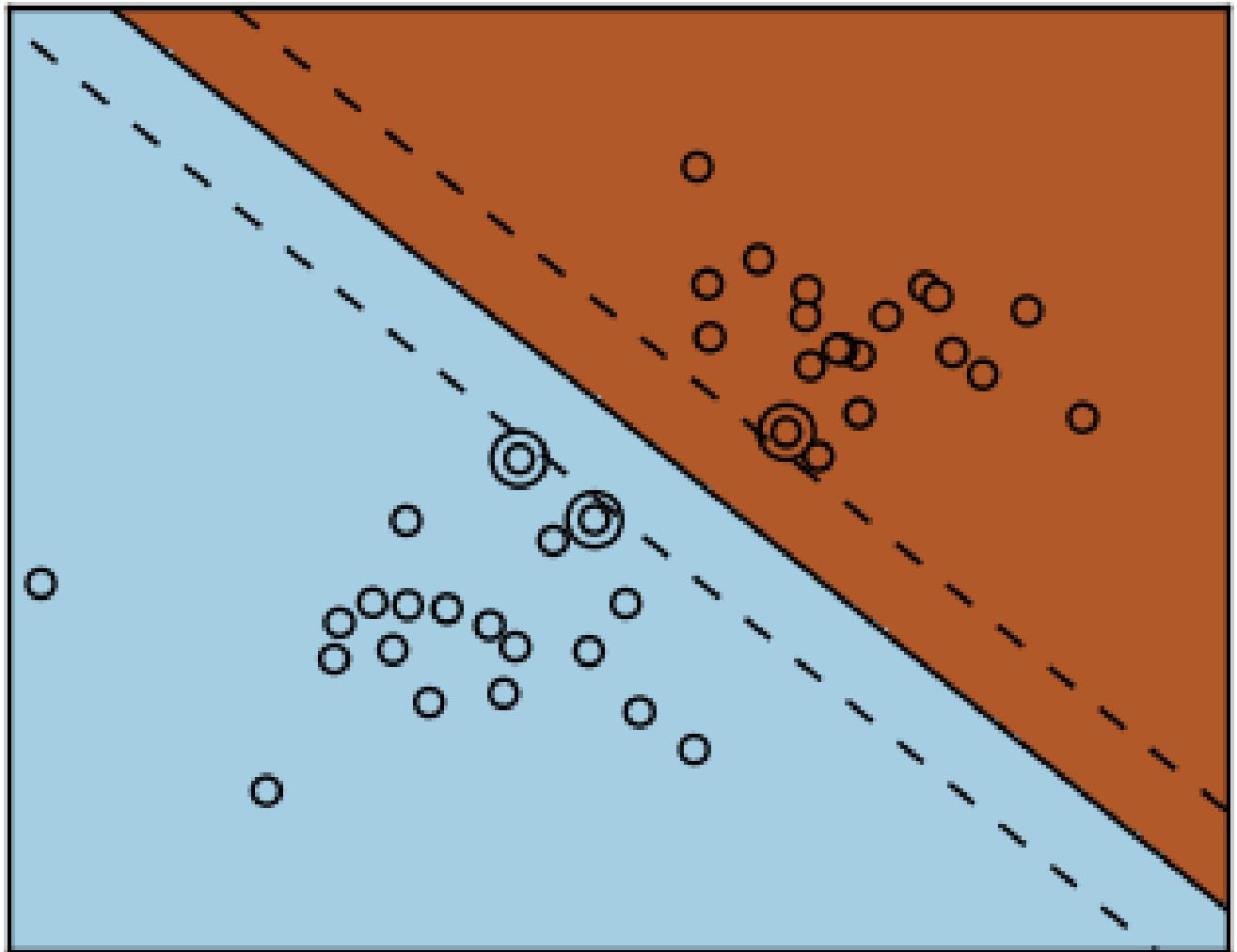
Опорные вектора

$$w = \sum_{i=1}^N \alpha_i y_i x_i$$

$$\alpha_i(y_i(w^T x_i - b) - 1) = 0$$

$$x_i: \alpha_i > 0$$

$$y_i(w^T x_i - b) - 1 = 0$$



Неразделимая выборка

Линейно разделимая выборка:

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

Неразделимая выборка:

$$\begin{cases} \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \rightarrow \min \\ y_i(w^T x_i - b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases}$$

Применение ККТ к задаче SVM на неразделимой выборке

$$\mathcal{L}(\underbrace{\mathbf{w}, b, \xi}_{\mathbf{z}}, \underbrace{\alpha, r}_{\alpha}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i - b) - 1 + \xi_i) - \sum_{j=1}^N r_j \xi_i$$

$$= \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i - b) - 1) - \sum \xi_i (r_i + \alpha_i - C)$$

$$\alpha_i, r_i \geq 0; \quad \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i - b) - 1 + \xi_i) = 0; \quad r_i \xi_i = 0$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial b} = \frac{\partial \mathcal{L}}{\partial \xi} = 0 \Rightarrow \begin{cases} \mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i \\ \sum \alpha_i y_i = 0 \\ \alpha_i = C - r_i \Rightarrow 0 \leq \alpha_i \leq C \end{cases}$$

Вид двойственной задачи

$$\left\{ \begin{array}{l} w = \sum \alpha_i y_i x_i \\ \sum \alpha_i y_i = 0 \\ \alpha_i = C - r_i \end{array} \right. \Rightarrow \quad \begin{aligned} \mathcal{L}(\alpha) &= \frac{1}{2} w^T w - \sum \alpha_i (y_i (w^T x_i - b) - 1) - \sum \xi_i (r_i + \alpha_i - C) = \\ &= \frac{1}{2} \sum \sum y_i y_j \alpha_i \alpha_j x_i^T x_j - \sum \sum y_i y_j \alpha_i \alpha_j x_i^T x_j + \sum \alpha_i \end{aligned}$$

$$\left\{ \begin{array}{l} \max_{\alpha} \mathcal{L}(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum y_i y_j \alpha_i \alpha_j x_i^T x_j \\ 0 \leq \alpha_i \leq C \\ \sum \alpha_i y_i = 0 \end{array} \right.$$

Типы объектов

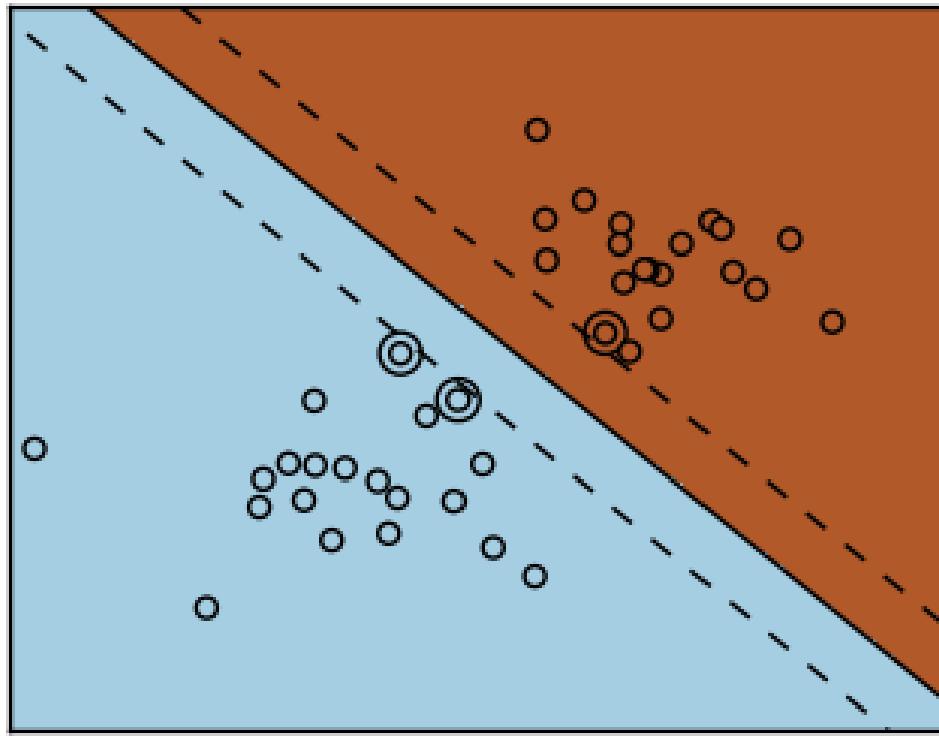
$$w = \sum \alpha_i y_i x_i; \quad y_i(w^T x_i - b) \geq 1 - \xi_i$$

$$\alpha_i = C - r_i; \quad \alpha_i(y_i(w^T x_i - b) - 1 + \xi_i) = 0; \quad r_i \xi_i = 0$$

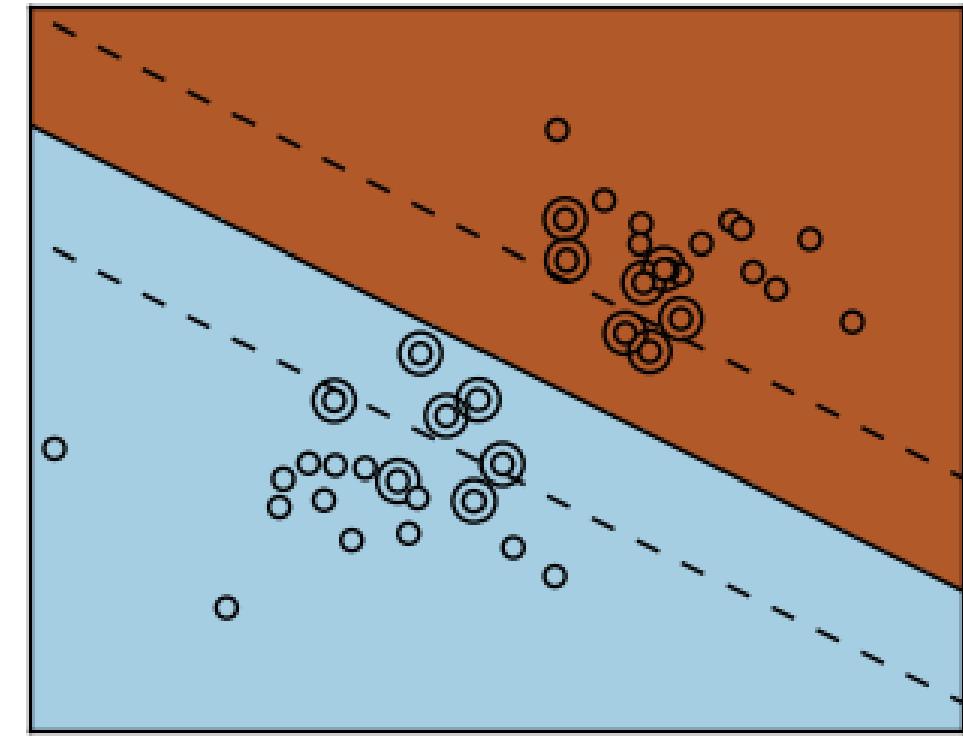
1. $\alpha_i = 0; \xi_i = 0; y_i(w^T x_i - b) \geq 1$ – Внутренние объекты
2. $0 < \alpha_i < C; \xi_i = 0; y_i(w^T x_i - b) = 1$ – Опорные объекты
3. $\alpha_i = C; \xi_i > 0; y_i(w^T x_i - b) \leq 1$ – Опорные объекты – нарушители

Влияние константы

$$\min \left(\frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \right)$$

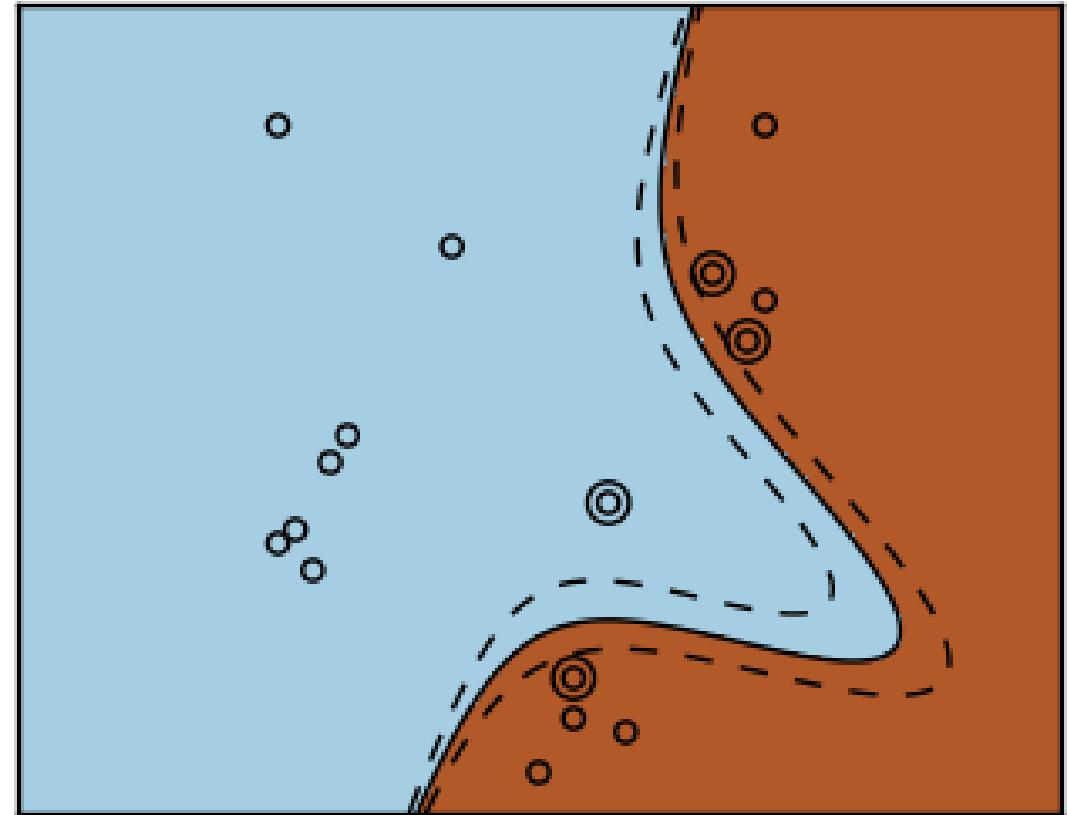
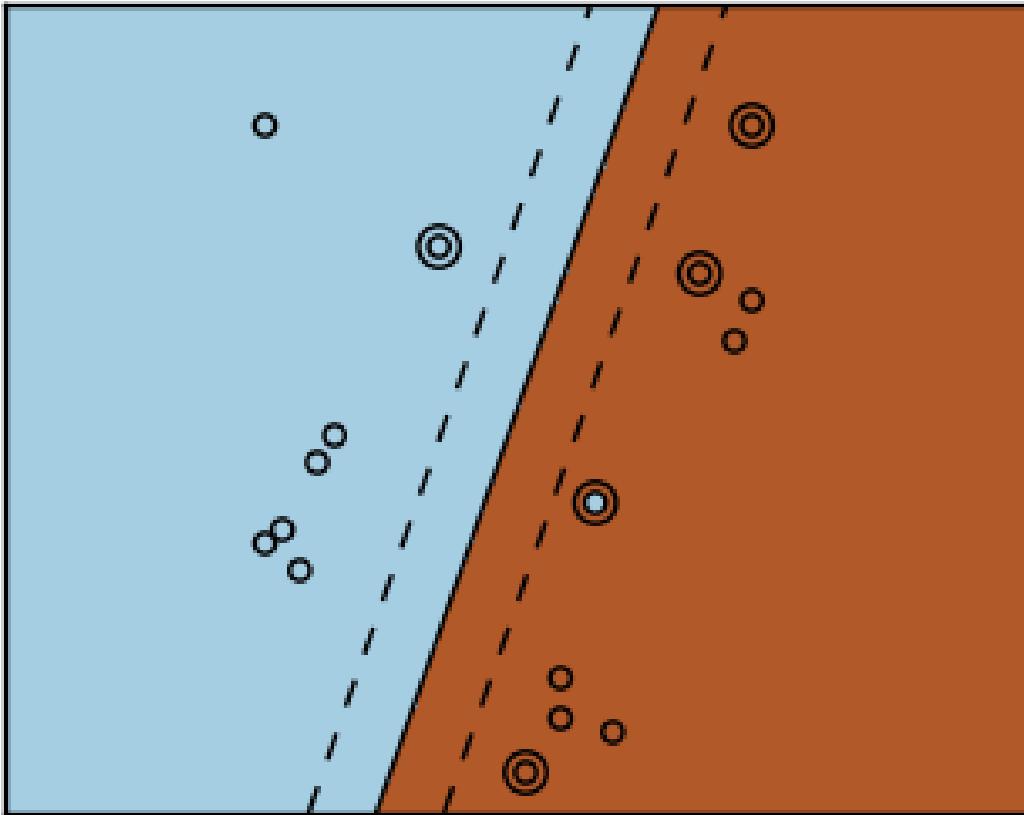


$C \uparrow$



$C \downarrow$

Более высокая размерность и ошибка



$$E_{gen} \leq \frac{\# \text{ of support vectors}}{\# \text{ of training vectors}}$$

Kernel trick (Ядерный трюк)

Заметим, что во всех расчетах мы используем только $x_i^T x_j$, тогда, для перехода к более высокой размерности, нужно лишь определить скалярное произведение векторов.

$K(x, x')$ – ядро, если $K(x, x') = \psi(x)^T \psi(x')$, при $\psi: X \rightarrow H$, где H – гильбертово пространство.

Пример:

$$K(x, x') = (1 + x^T x')^2 = 1 + x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2x_1 x'_1 + 2x_2 x'_2 + 2x_1 x'_1 x_2 x'_2$$

$$\psi(x) = (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2)$$

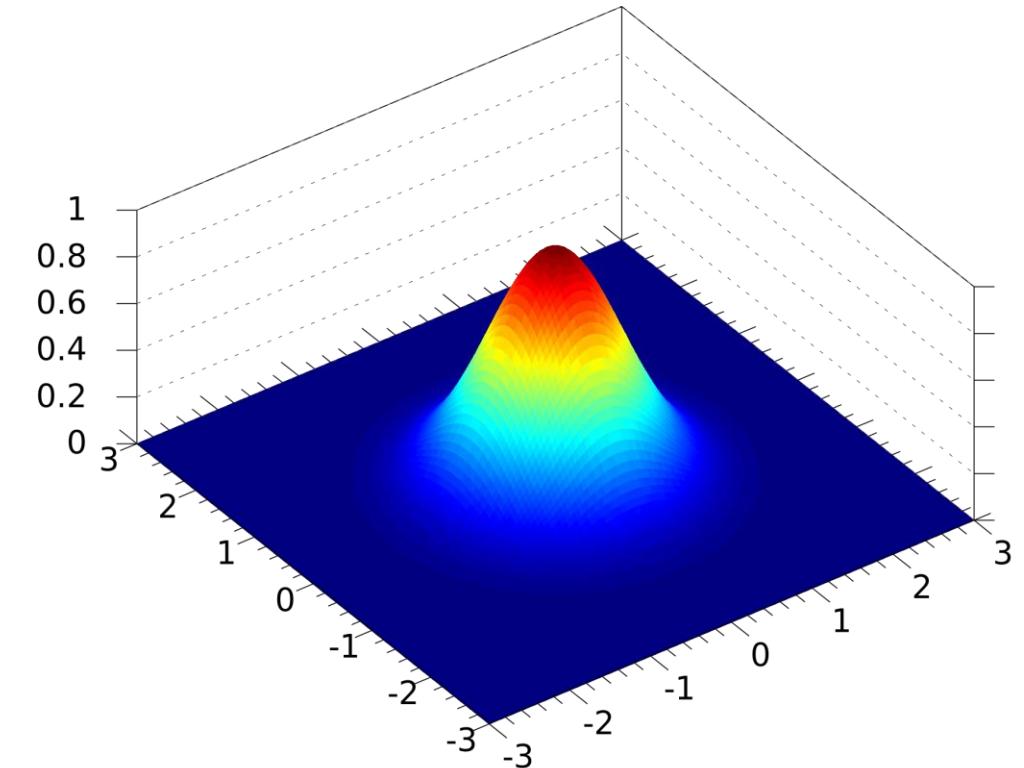
Разные ядра

Линейное ядро: $\langle \mathbf{x}, \mathbf{x}' \rangle$

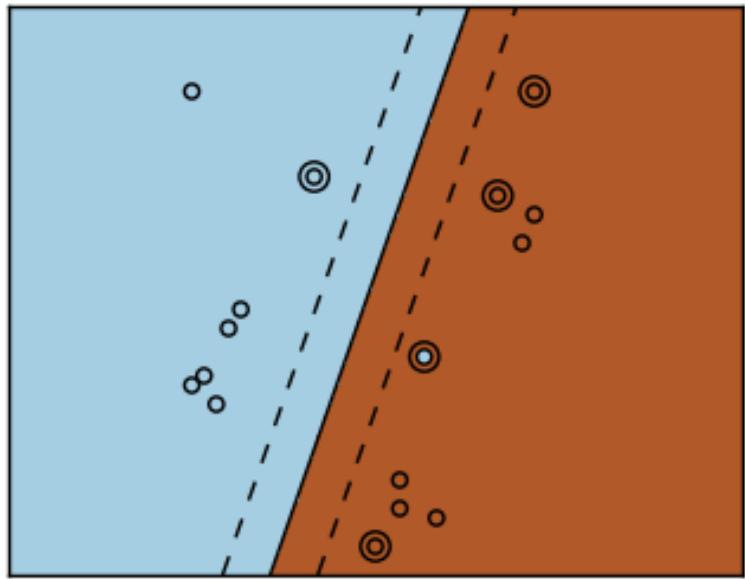
Полиномиальное ядро: $(r + \gamma \langle \mathbf{x}, \mathbf{x}' \rangle)^d$

Гауссианово ядро (RBF): $e^{-\gamma |\mathbf{x} - \mathbf{x}'|^2}$

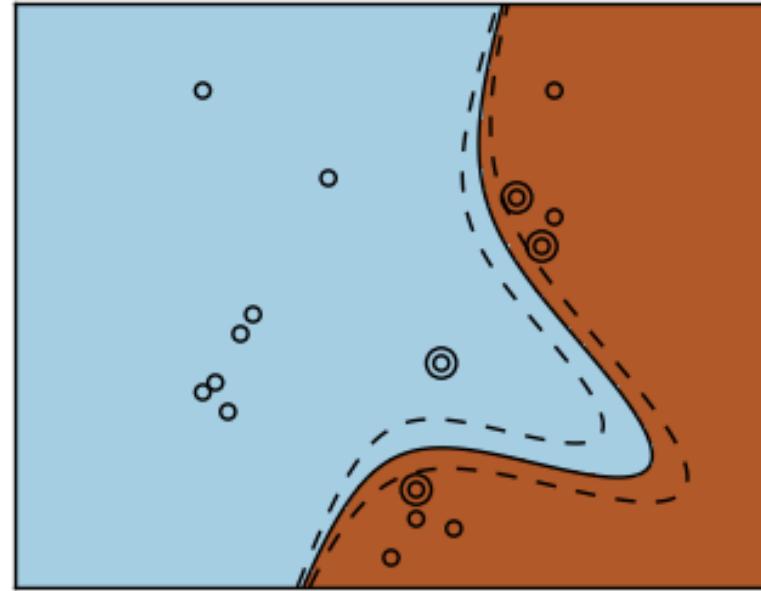
Сигмоид: $\text{th}(\gamma \langle \mathbf{x}, \mathbf{x}' \rangle + r)$



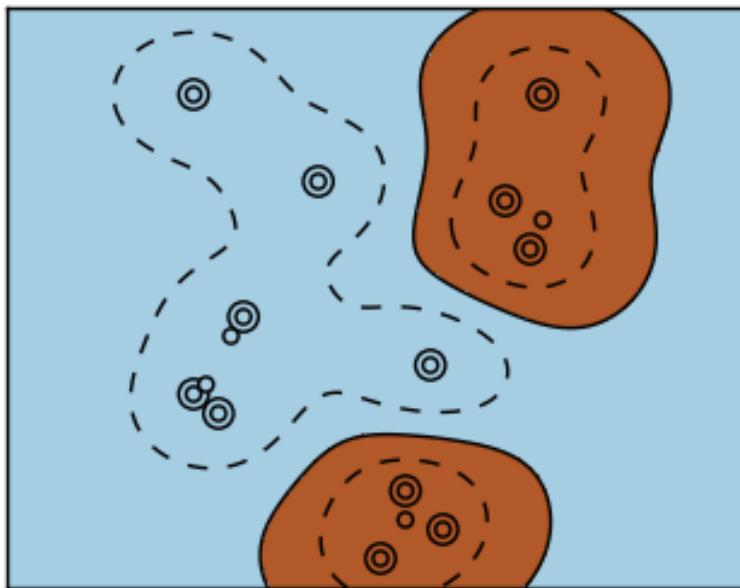
Radial Basis Function (RBF)



Линейное ядро



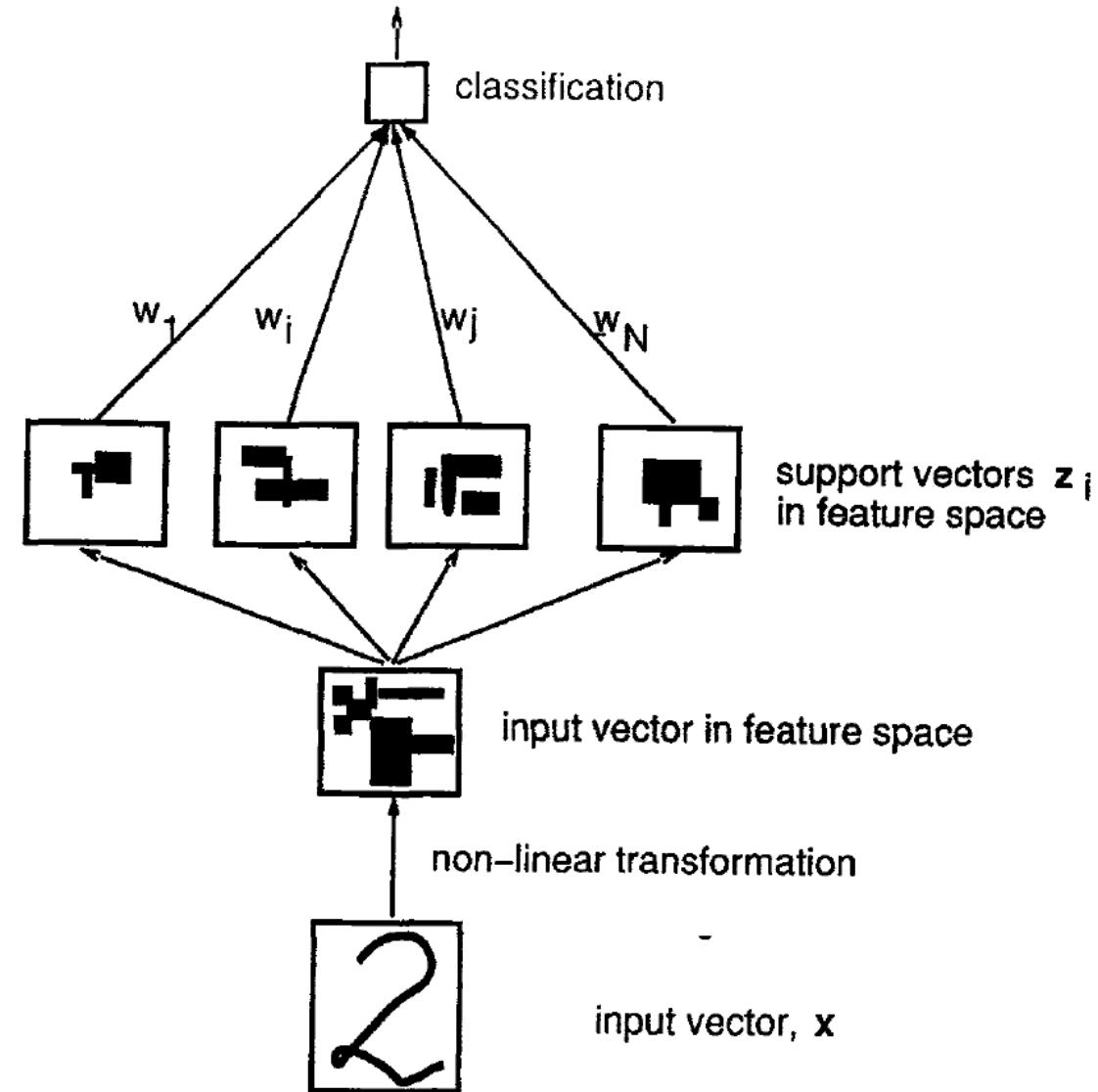
Полиномиальное ядро



Гауссианово ядро (RBF)

Support Vector Networks

Cortes and Vapnik, 1995



Байесовский классификатор

Текстовые задачи

- Классификация текста
- Определение авторства
- Определение эмоциональной окраски текста
- Парсинг научных статей

Байесовская постановка задачи

y – класс, x - документ

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)}$$

$$y_{MAP} = \arg \max_{y \in Y} P(y|x) = \arg \max_{y \in Y} \frac{P(y)P(x|y)}{P(x)} = \arg \max_{y \in Y} P(y)P(x|y)$$

$$\arg \max_{y \in Y} P(y)P(x|y) = \arg \max_{y \in Y} P(x_1, x_2, \dots, x_n|y)P(y)$$

Наивное предположение (о независимости):

$$P(x_1, x_2, \dots, x_n|y) = P(x_1|y) P(x_2|y) P(x_3|y) \dots P(x_n|y)$$

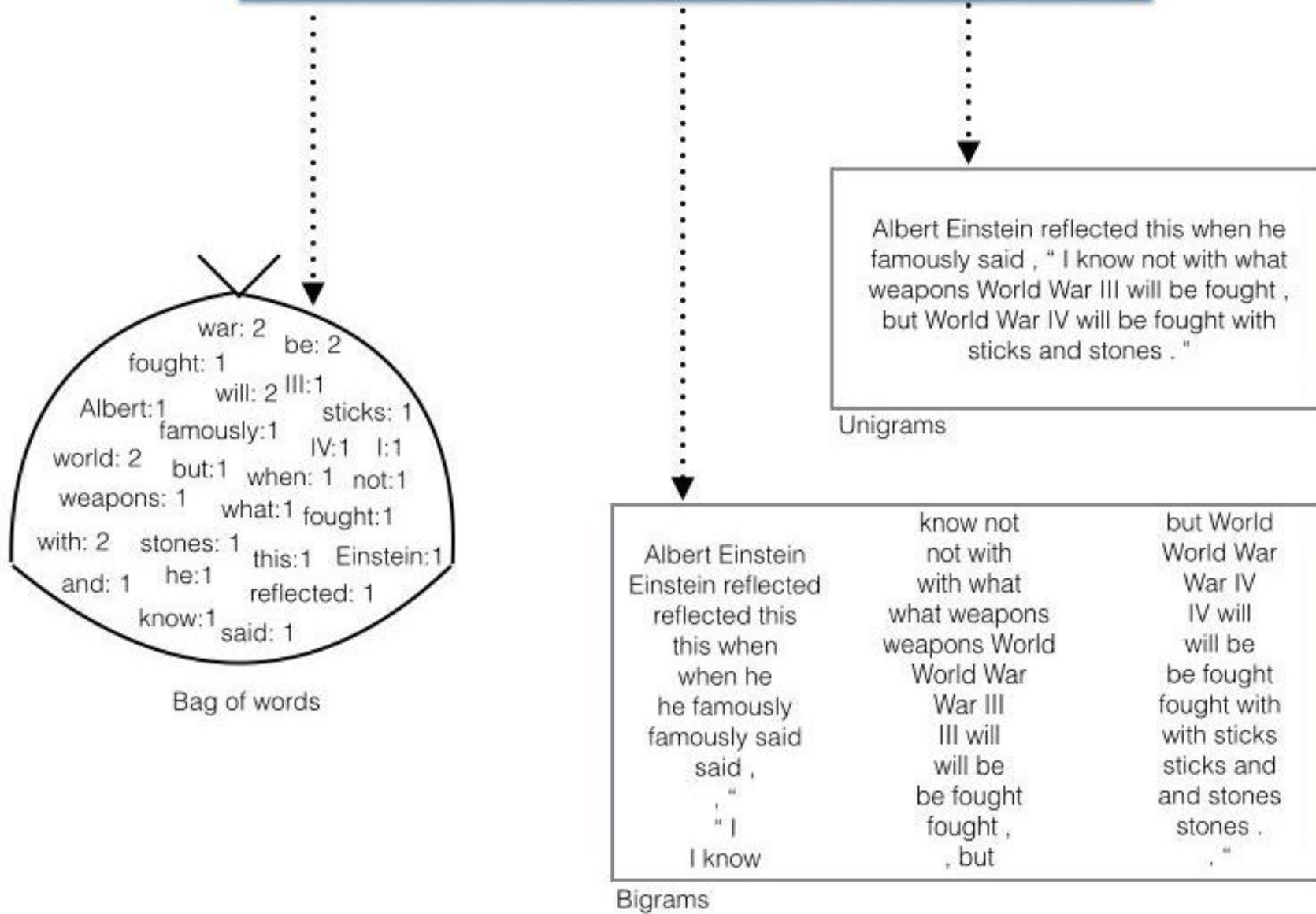
Bag-of-Words (BoW)

Важен только набор слов, не важен порядок.

x_1, x_2, \dots, x_n - количества слов из словаря длины n в документе.

Словарь может быть как полный, так и ограниченный.

Albert Einstein reflected this when he famously said, "I know not with what weapons World War III will be fought, but World War IV will be fought with sticks and stones."



Наивный байесовский классификатор

$$P(x_1, x_2, \dots, x_n | y) = P(x_1 | y) P(x_2 | y) P(x_3 | y) \dots P(x_n | y)$$

$$y_{MAP} = \arg \max_{y \in Y} P(y) P(\mathbf{x}|y)$$

$$y_{NB} = \arg \max_{y \in Y} P(y) \prod_i P(x_i | y)$$

Слова в документах

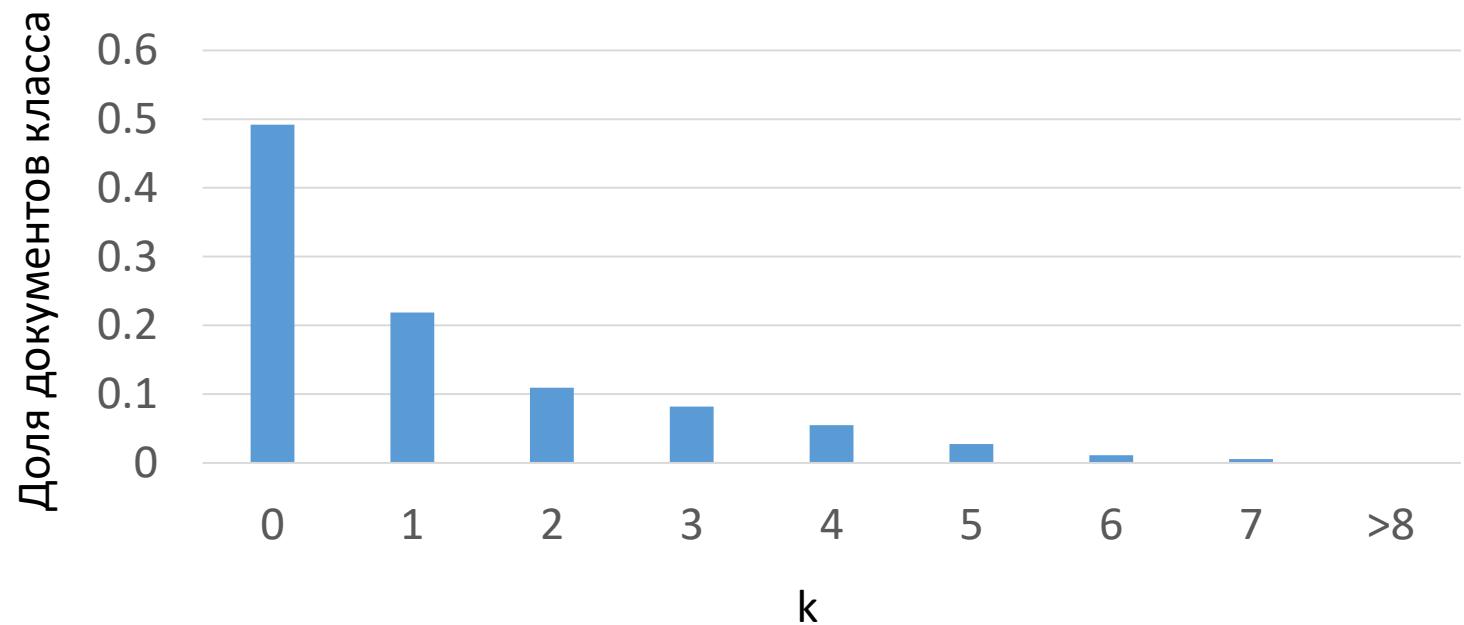
$P(y)$ – частота класса y .

$P(x_i|y)$ – вероятность значения признака x_i в классе y , например для документов в классе, в которых определенное слово встречается k раз.

$$P(x_i|y) = \frac{\text{count}(x_i,y)}{\text{count}(y)}$$

Проблема – $\text{count}(x_i,y) = 0$

$$\hat{P}(x_i|y) = \frac{\text{count}(x_i,y)+\alpha}{\text{count}(y)+\alpha K}$$



Обработка текста

- Уменьшение словаря:
 - 35, 535, 17, 200000 → \$number
 - $(5+3), \frac{1}{2}w^T w + C \rightarrow \$formula$
 - Stemming – приведение слова в инфинитивную форму (не всегда работает хорошо)
- Повышение веса:
 - Слова в названии документа (гиперссылка, подписях к картинкам)
 - Слова в предложениях, которые содержат слова из названия
 - Первое предложение в каждом абзаце

Байесовский классификатор с другими признаками

Бернуlli:

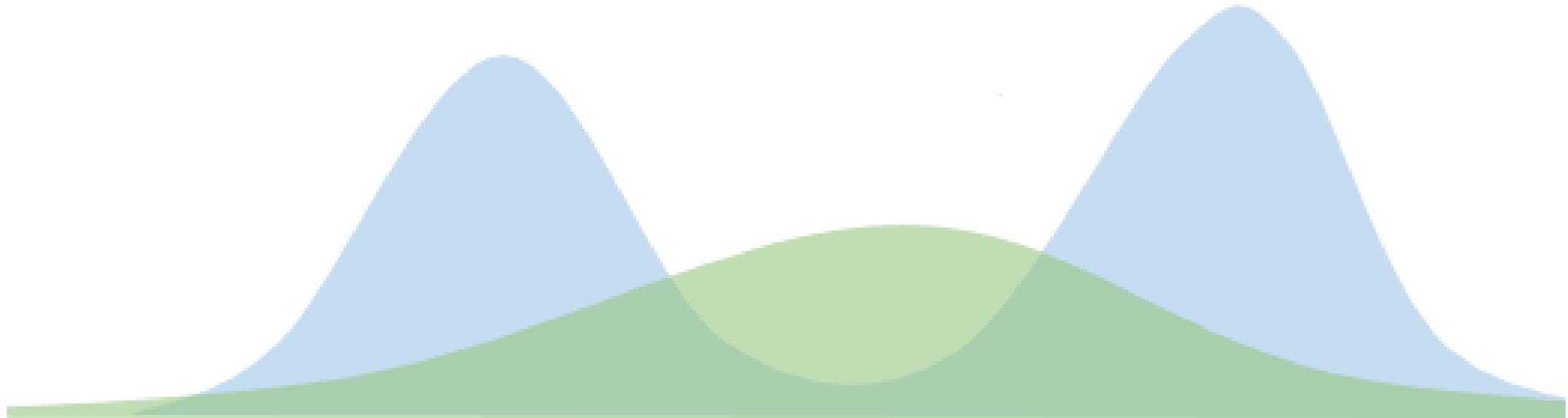
$$P(x_i|y) = P(x_i = 1|y)x_i + (1 - P(x_i = 1|y))(1 - x_i), \quad x_i \in \{0,1\}$$

Распределение Гаусса:

$$p(x_i|y) = \frac{1}{\sqrt{2\pi(\sigma_i^y)^2}} e^{-\frac{(x_i - \mu_i^y)^2}{2(\sigma_i^y)^2}}$$

Оценка распределения

Простой вариант (для домашки, например) – выборочное среднее и дисперсия для μ и σ .



Более сложный вариант – EM (Expectation-maximization) со смесью Гауссиан (Gaussian mixture).

Expectation-maximization (EM)

Смесь К Гауссиан задается параметрами:

μ_k – вектор среднего, Σ_k – матрица ковариаций

α_k – "вес" гауссианы, вероятность того, что случайная точка принадлежит к Гауссиане k

$$\sum \alpha_k = 1$$

Принадлежность объекта x_i к k -му распределению :

$$w_{ik} = p(\mu_k, \Sigma_k | x_i) = \frac{p(x_i | \mu_k, \Sigma_k) \cdot \alpha_k}{\sum_j p(x_i | \mu_j, \Sigma_j) \cdot \alpha_j}$$

$$\alpha_k^{new} = \frac{\sum_{i=1}^N w_{ik}}{N} = \frac{N_k}{N}$$

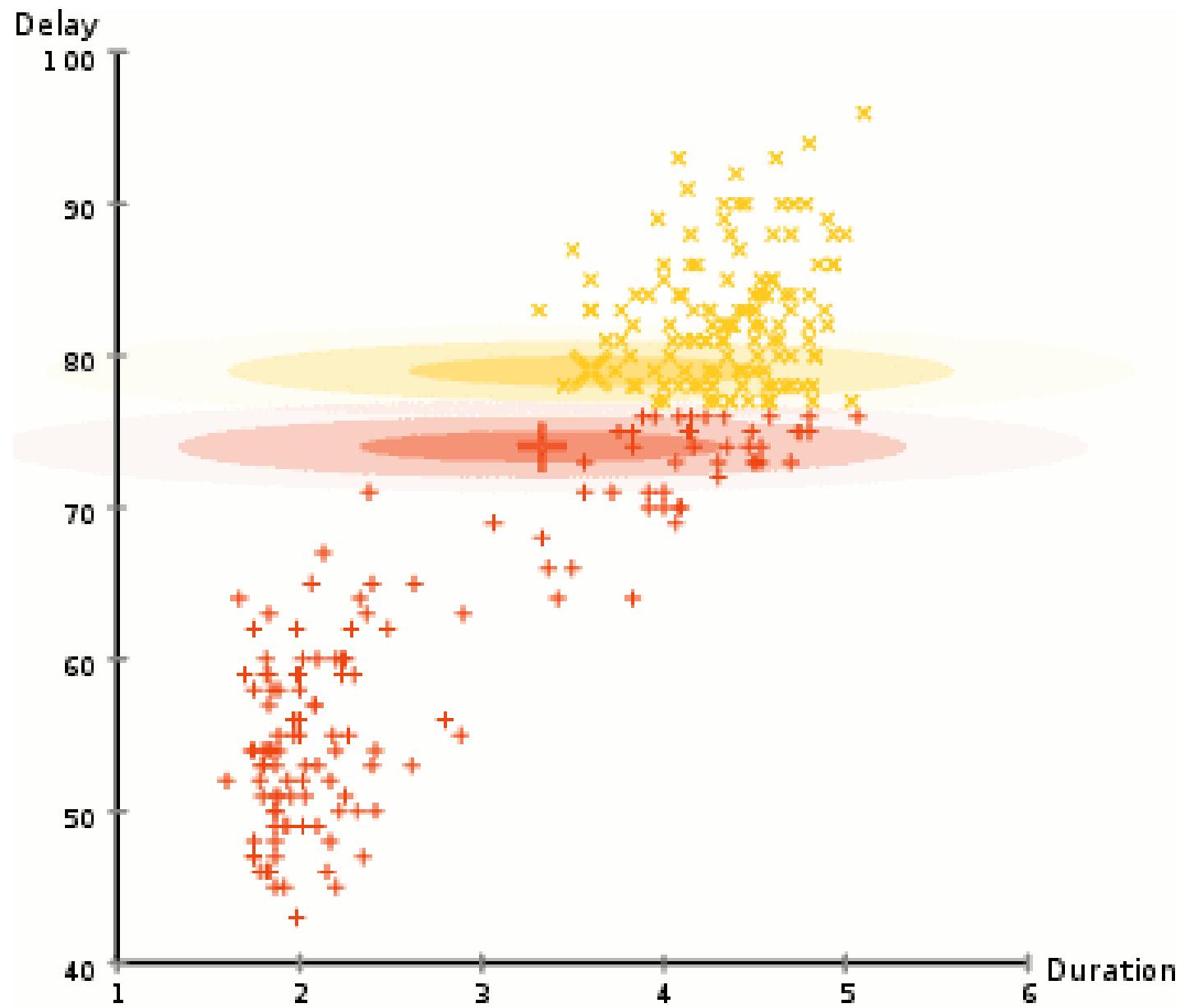
E-Step: считаем w_{ik}

M-Step:

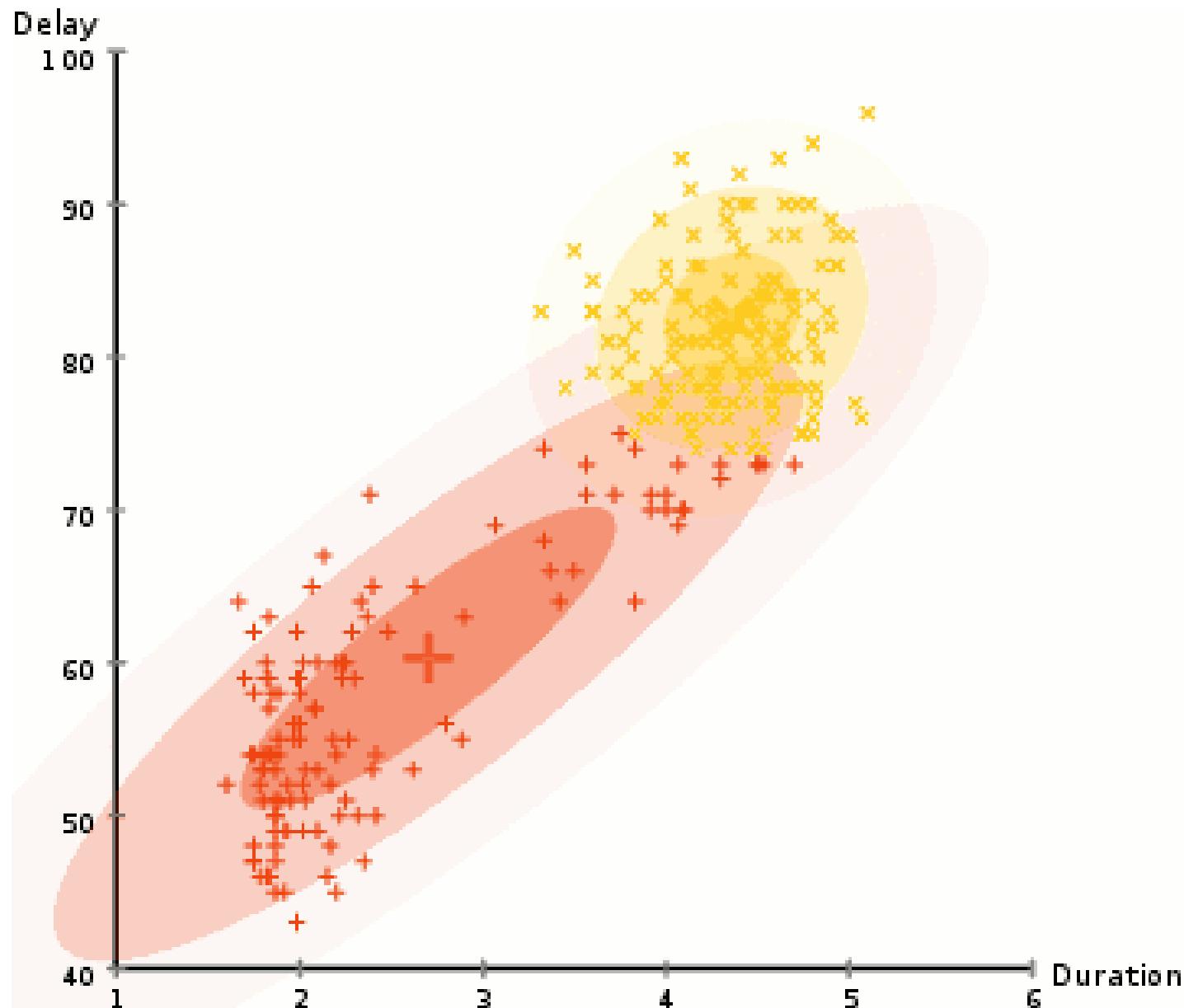
$$\mu_k^{new} = \left(\frac{1}{N_k} \right) \sum_{i=1}^N w_{ik} \cdot x_i$$

$$\Sigma_k^{new} = \left(\frac{1}{N_k} \right) \sum_{i=1}^N w_{ik} \cdot (x_i - \mu_k^{new})(x_i - \mu_k^{new})^T$$

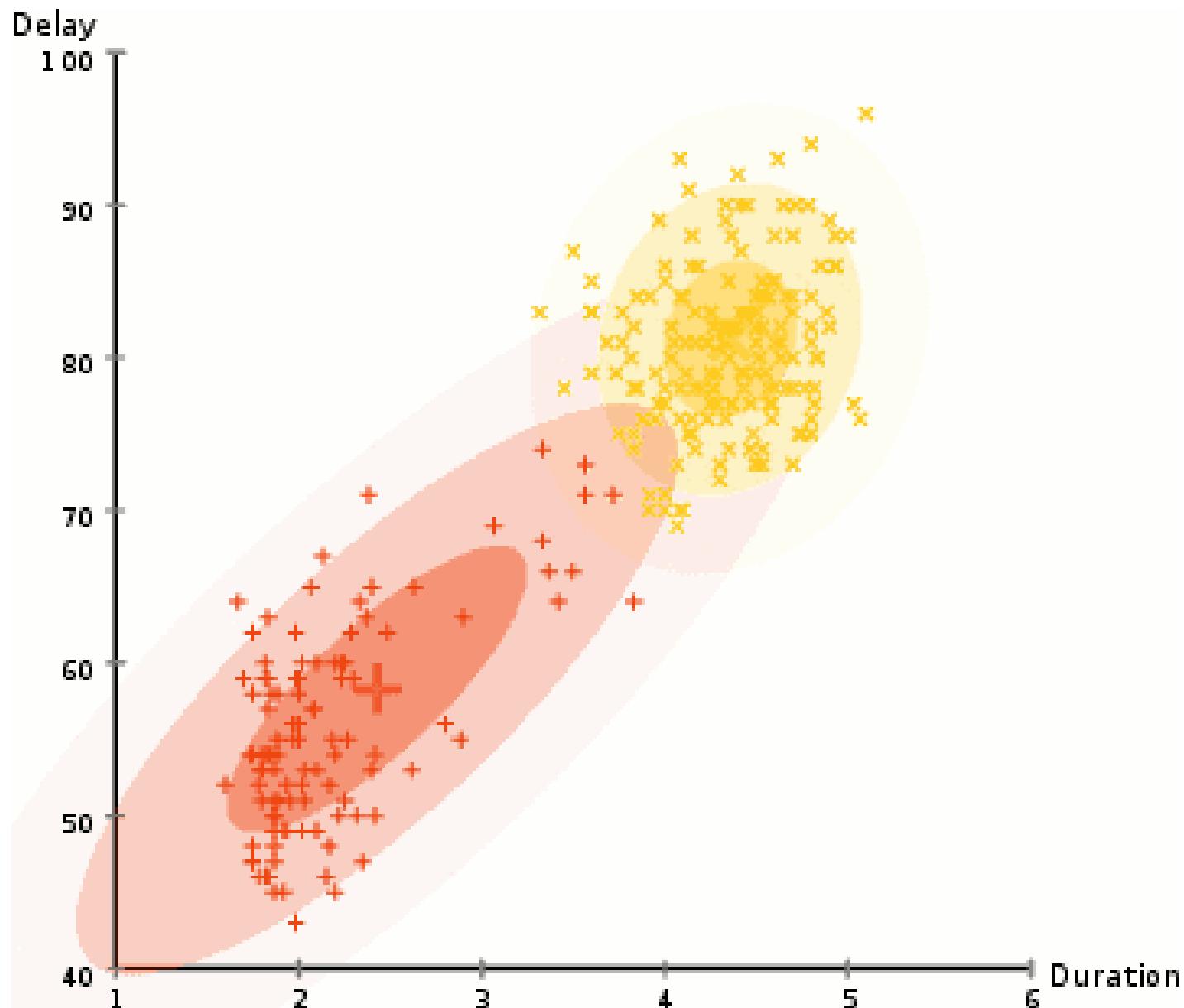
EM Clustering



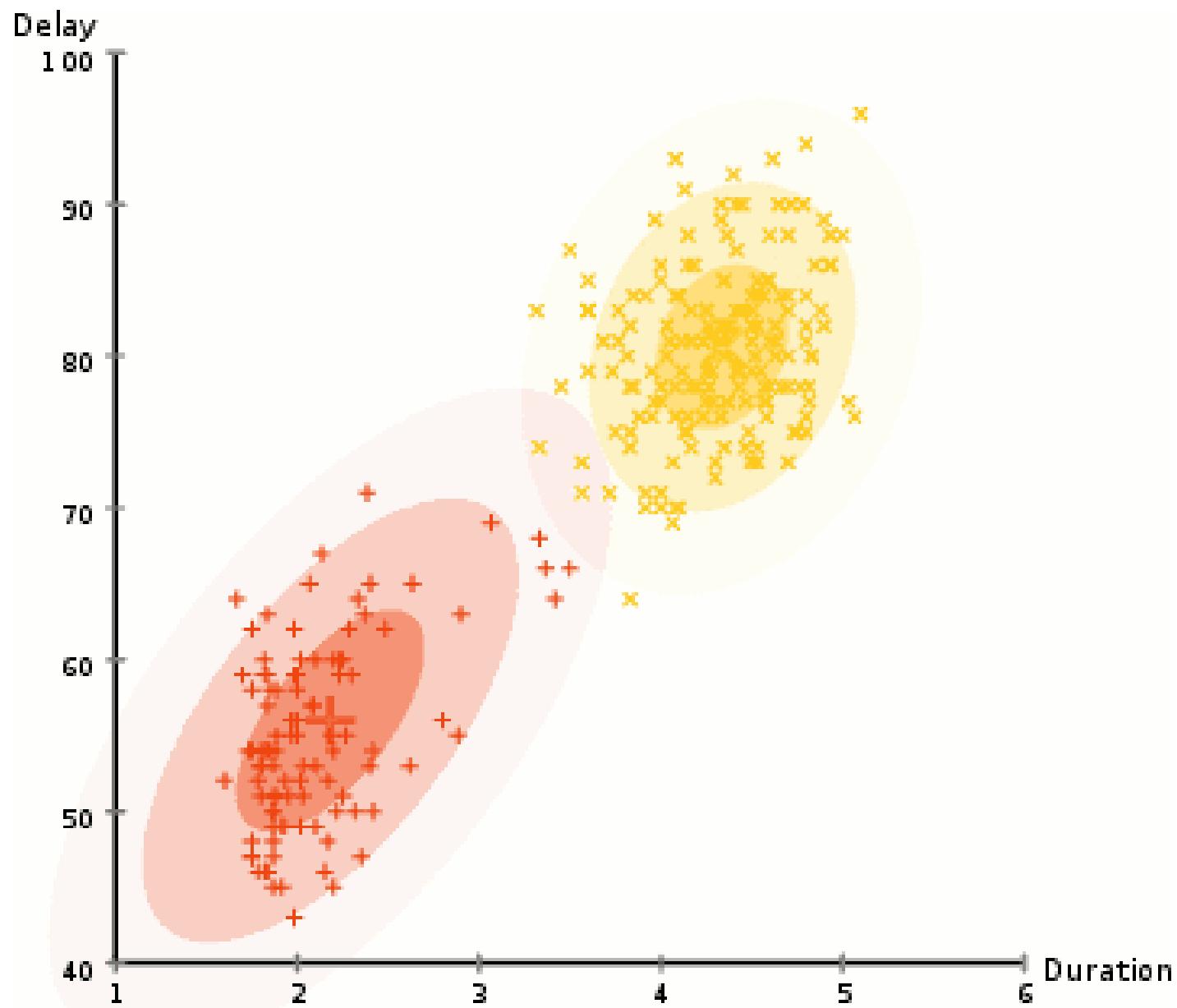
EM Clustering



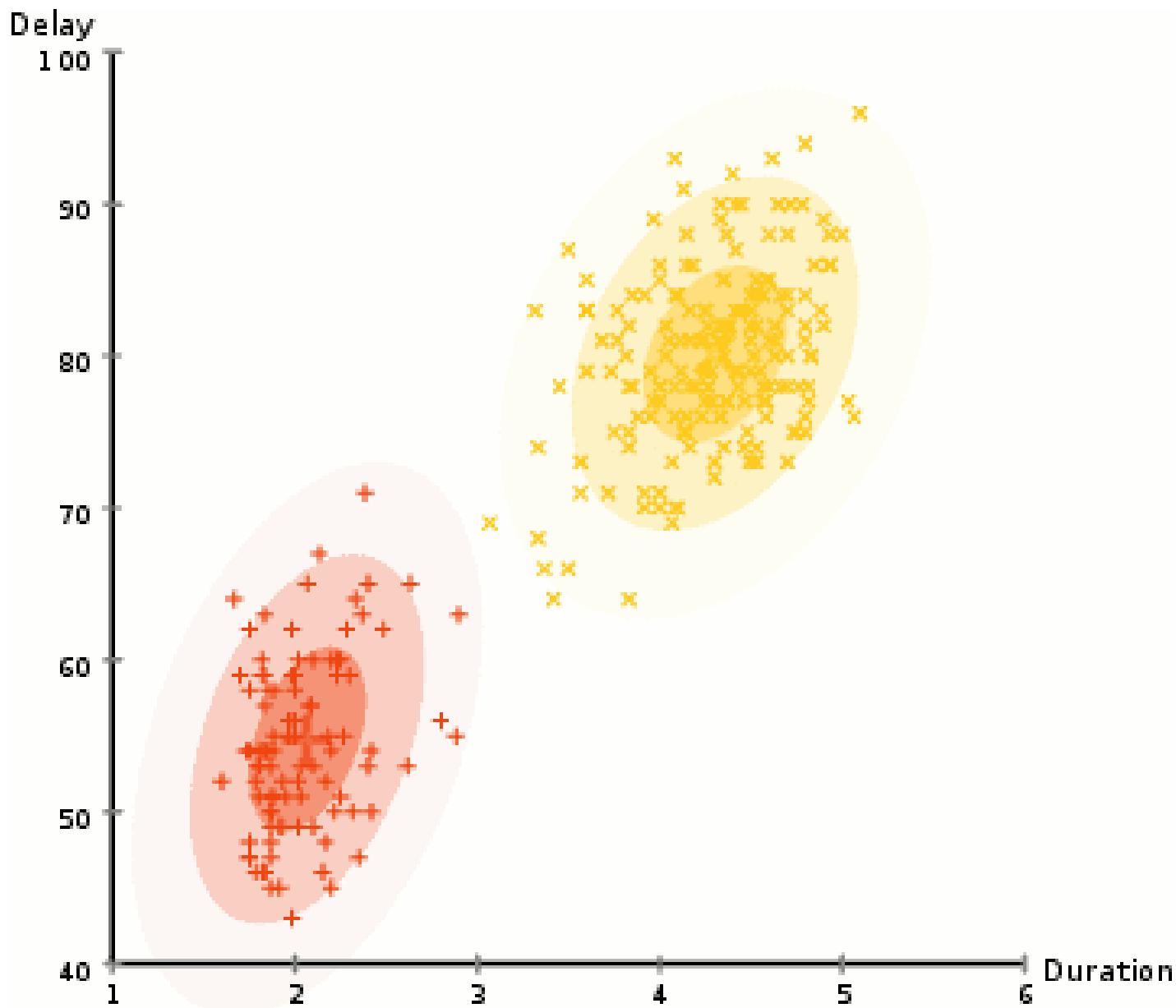
EM Clustering



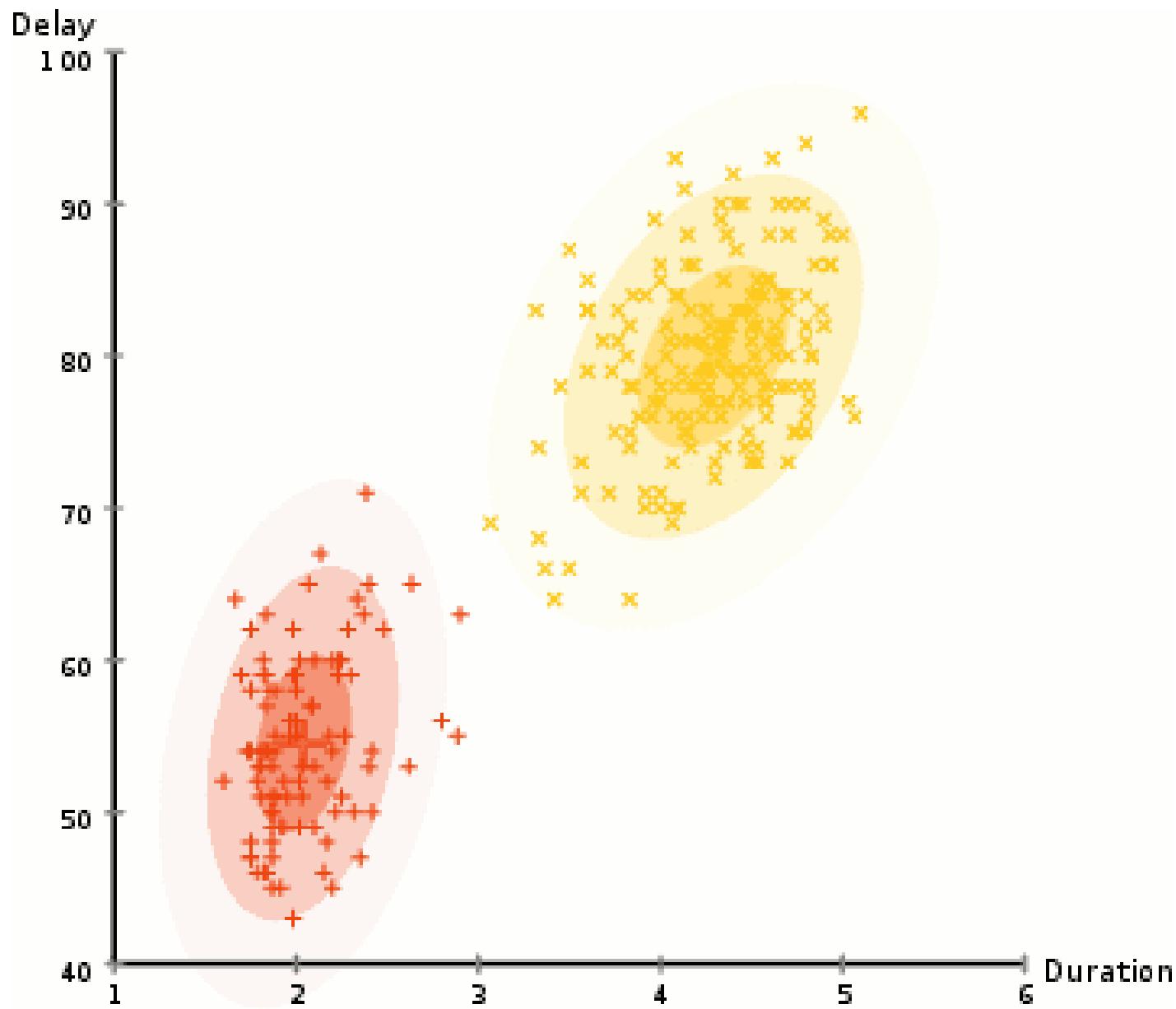
EM Clustering



EM Clustering



EM Clustering



Naïve Bayes – отличный Baseline!

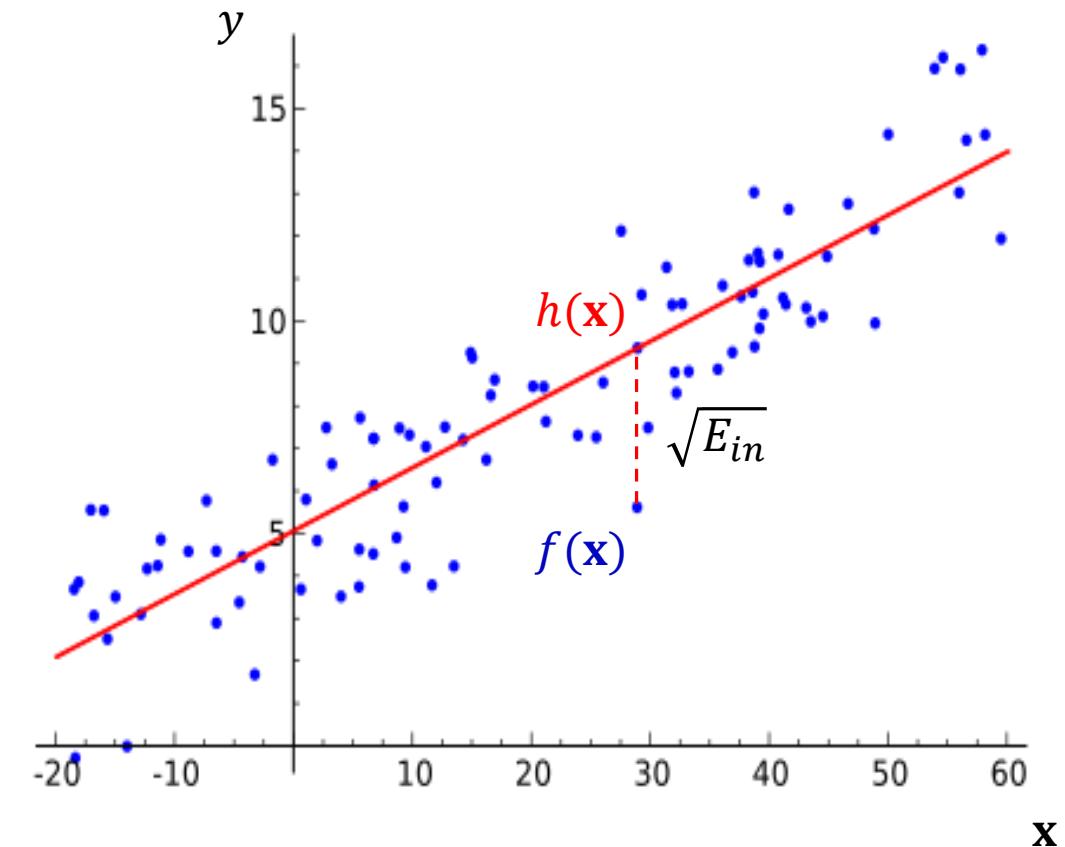
Регрессия

Линейная регрессия

$$E_{in}(h, \mathbf{x}) = (h(\mathbf{x}) - f(\mathbf{x}))^2$$

$$E_{in}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2 = \frac{1}{N} \| \mathbf{X}\mathbf{w} - \mathbf{y} \|_2^2$$

$$\mathbf{X} = \begin{bmatrix} -\mathbf{x}_1^T- \\ -\mathbf{x}_2^T- \\ \vdots \\ -\mathbf{x}_N^T- \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$



Линейная регрессия

$$E_{in}(\mathbf{w}) = \frac{1}{N} \left\| \mathbf{X}\mathbf{w} - \mathbf{y} \right\|_2^2$$

$$\nabla E_{in}(\mathbf{w}) = \frac{2}{N} \mathbf{X}^T (\mathbf{X}\mathbf{w} - \mathbf{y}) = 0$$

$$\mathbf{X}^T \mathbf{X}\mathbf{w} = \mathbf{X}^T \mathbf{y}$$

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}, \quad \mathbf{w} = \mathbf{X}^\dagger \mathbf{y}$$

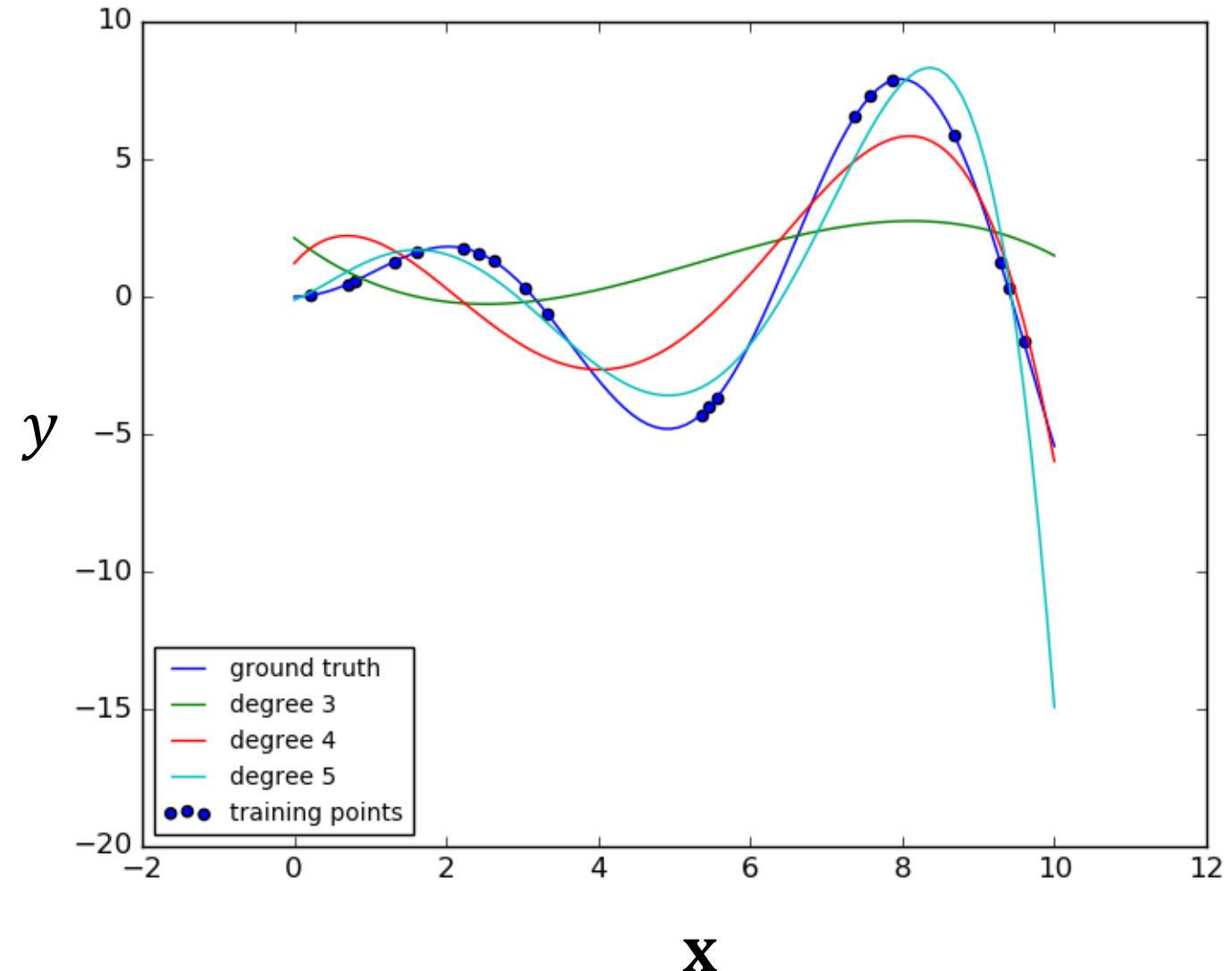
Полиномиальная регрессия

$X \rightarrow Z$

$x \rightarrow [1, x, x^2]$

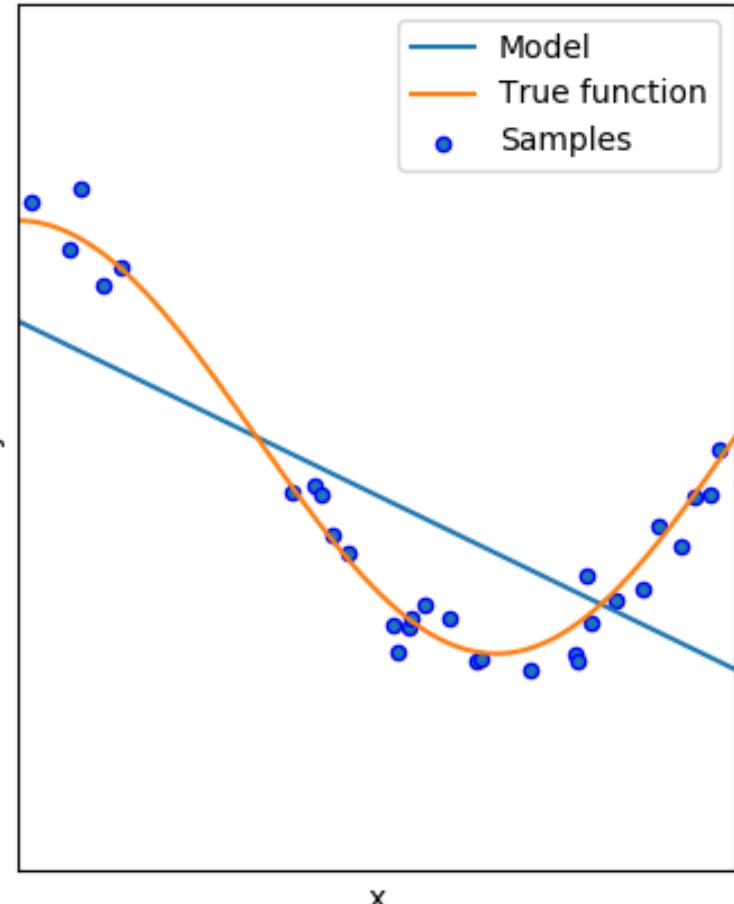
$[x_1, x_2] \rightarrow [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2]$

etc...

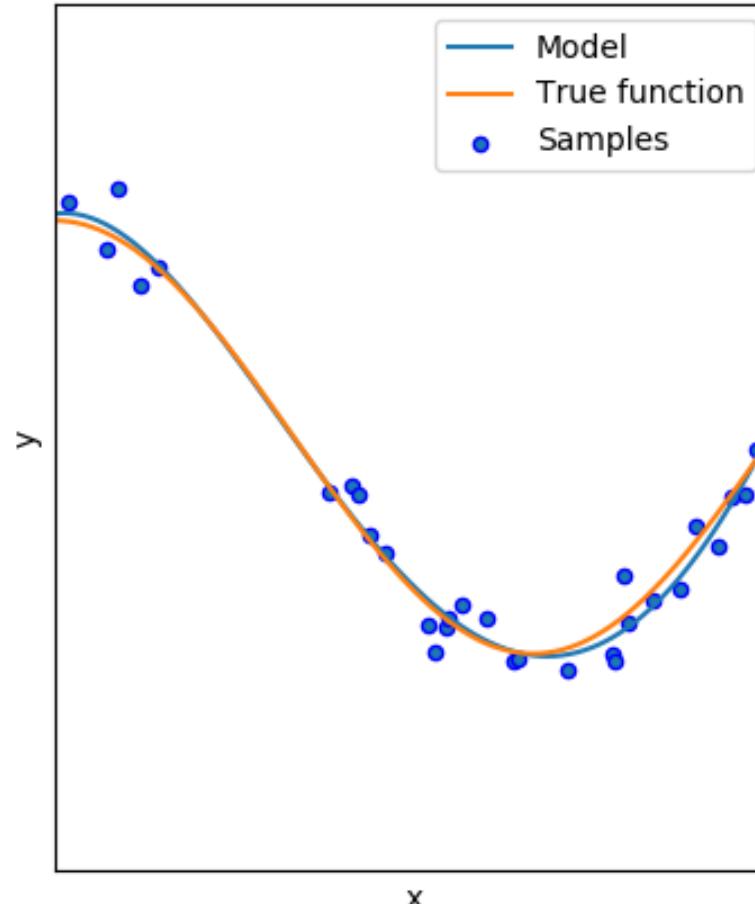


Полиномиальная регрессия

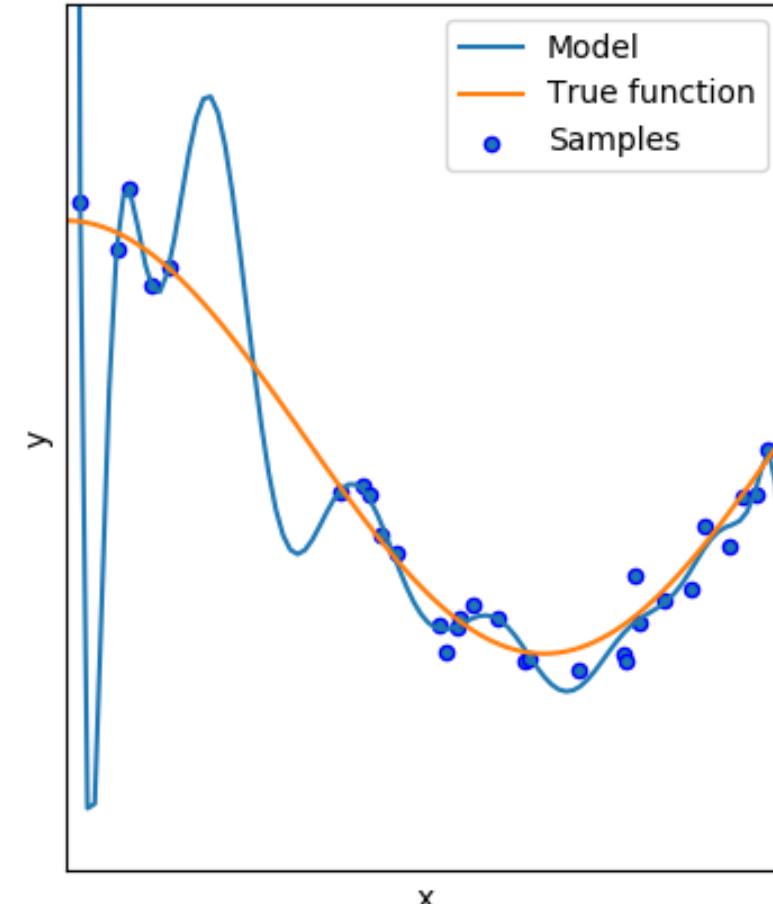
Degree 1
MSE = 4.08e-01(+/- 4.25e-01)



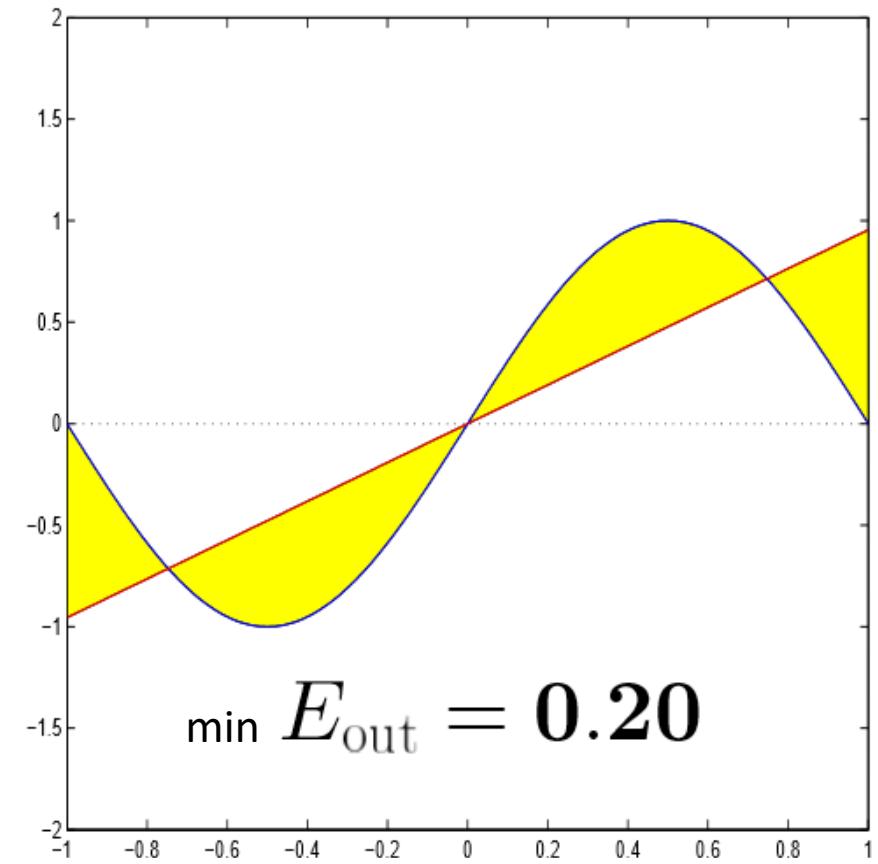
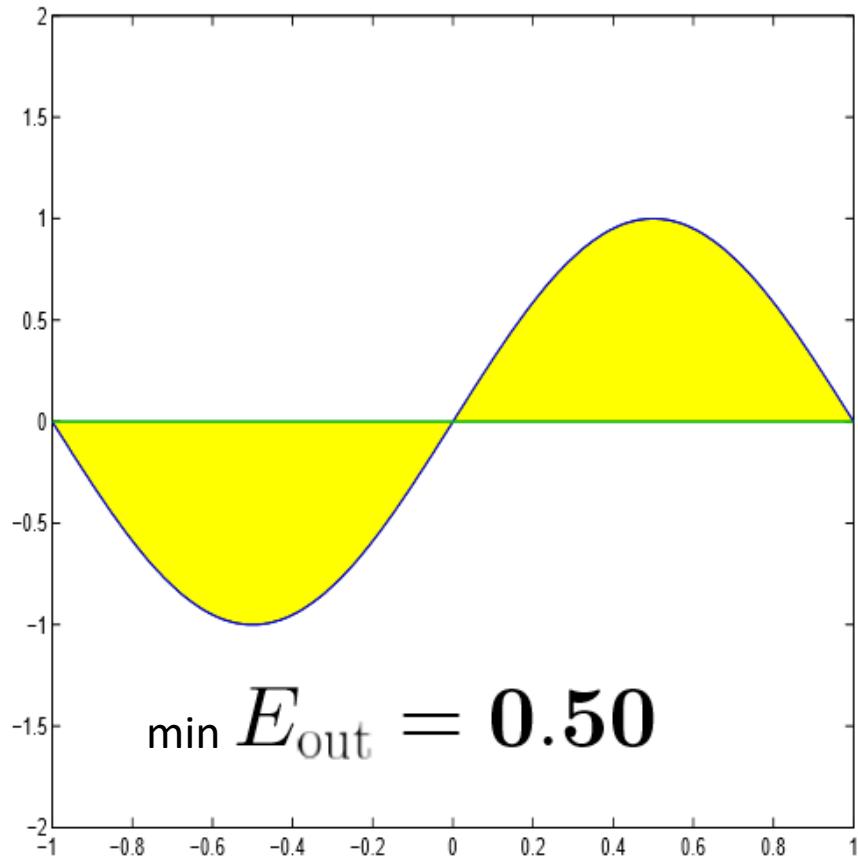
Degree 4
MSE = 4.32e-02(+/- 7.08e-02)



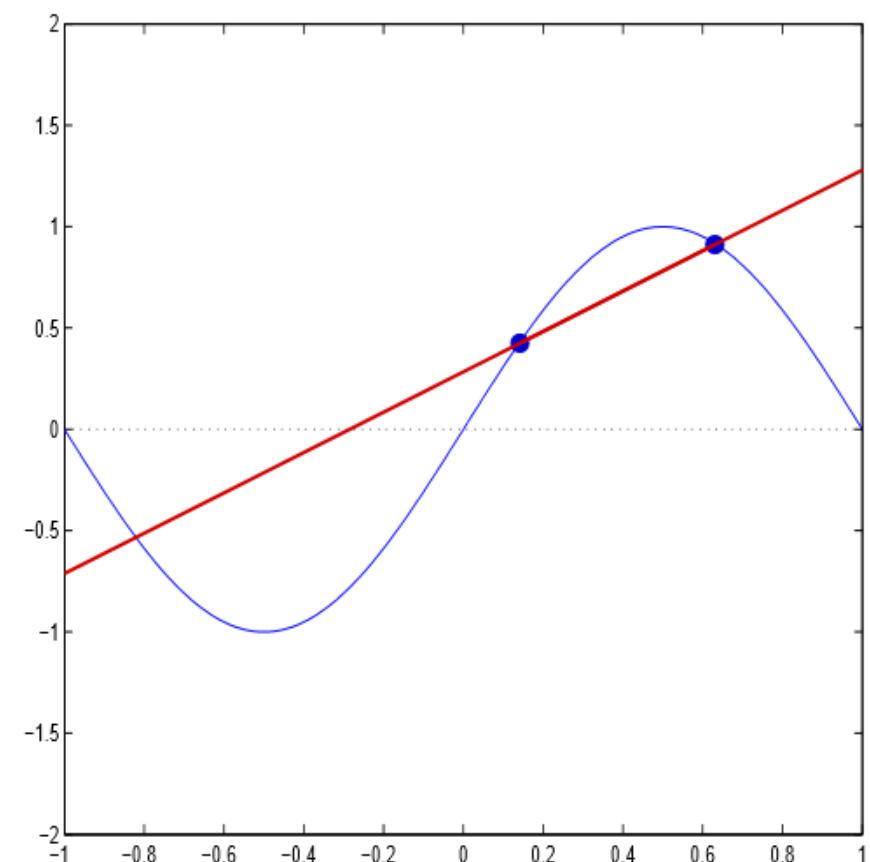
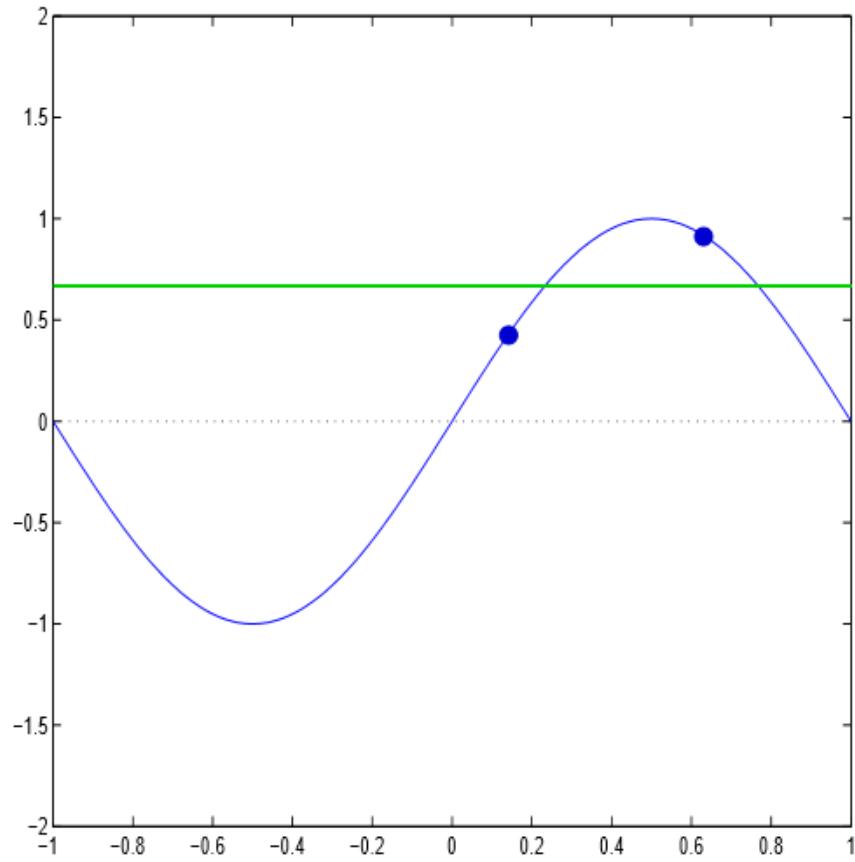
Degree 15
MSE = 1.82e+08(+/- 5.45e+08)



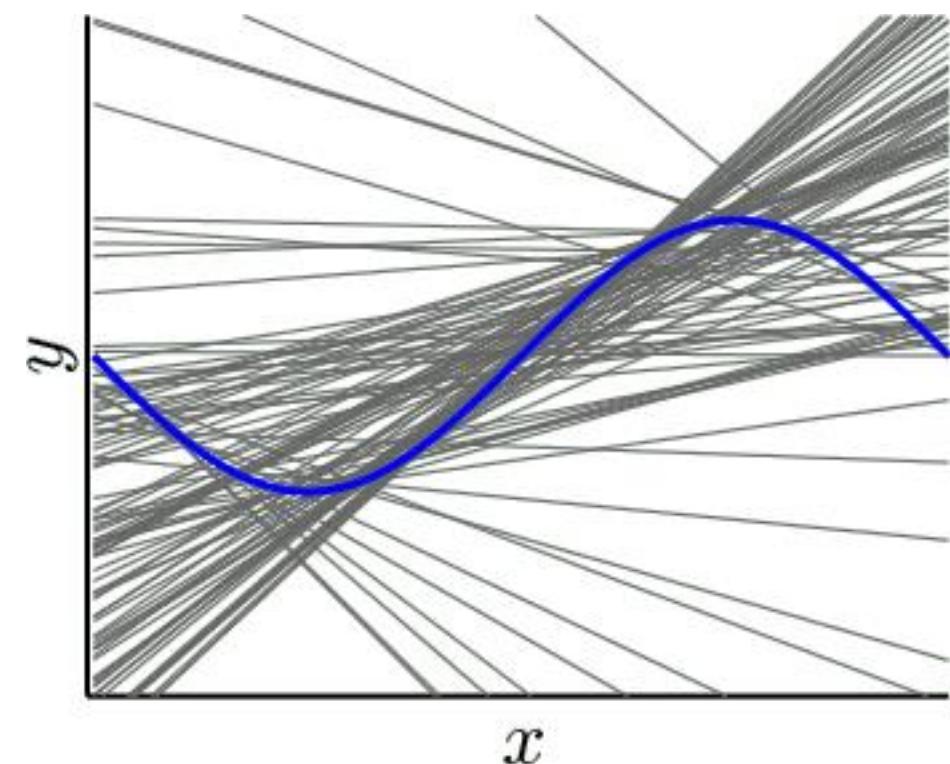
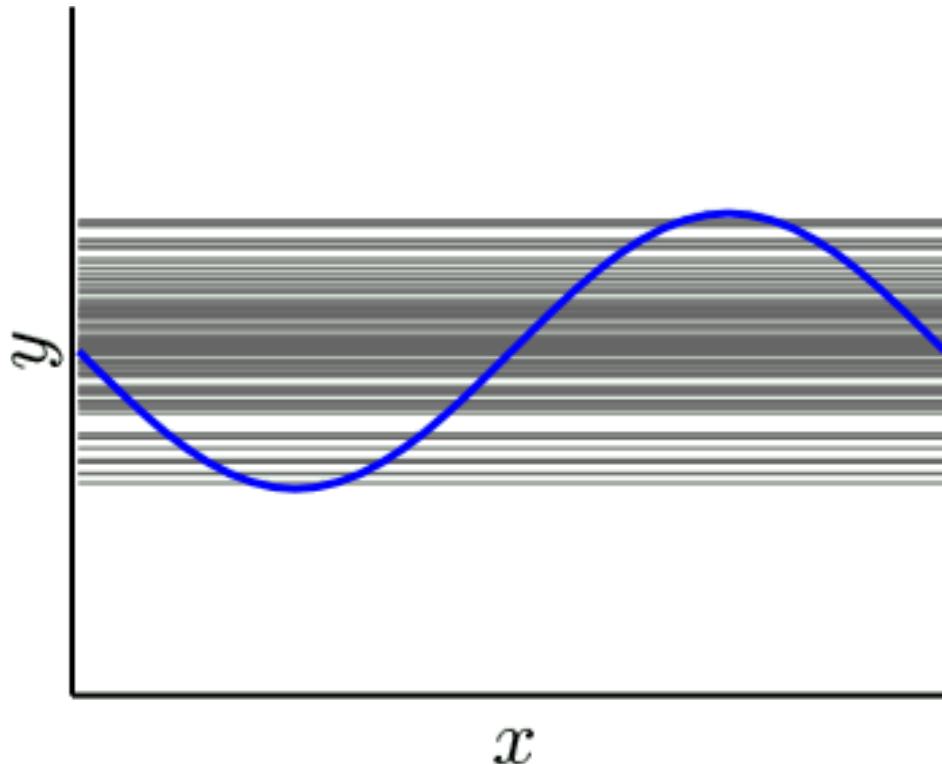
Цель - синус



Цель - синус



Цель - синус



Bias and Variance

$$E_{out}(h^D) = \mathbb{E}_{\mathbf{X}} \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] \quad (D \text{ -- датасет})$$

$$\mathbb{E}_D [E_{out}(h^D)] = \mathbb{E}_D \left[\mathbb{E}_{\mathbf{X}} \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] \right] = \mathbb{E}_{\mathbf{X}} \left[\mathbb{E}_D \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] \right]$$

$$\bar{h}(\mathbf{x}) = \mathbb{E}_D [h^D(\mathbf{x})] \quad (\text{средняя гипотеза})$$

$$\mathbb{E}_D \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] = \mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}) + \bar{h}(\mathbf{x}) - f(\mathbf{x}))^2 \right] =$$

$$= \mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 + (\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2 + 2(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))(\bar{h}(\mathbf{x}) - f(\mathbf{x})) \right] =$$

$$= \mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] + (\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2$$

Bias and Variance

$$\mathbb{E}_D \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] = \mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] + (\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2$$

$$\mathbb{E}_D [E_{out}(h^D)] = \mathbb{E}_{\mathbf{X}} \left[\mathbb{E}_D \left[(h^D(\mathbf{x}) - f(\mathbf{x}))^2 \right] \right] = \mathbb{E}_{\mathbf{X}} \left[\mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] + (\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2 \right] =$$

$$= \mathbb{E}_{\mathbf{X}} [variance(\mathbf{x}) + bias(\mathbf{x})] = bias + variance$$

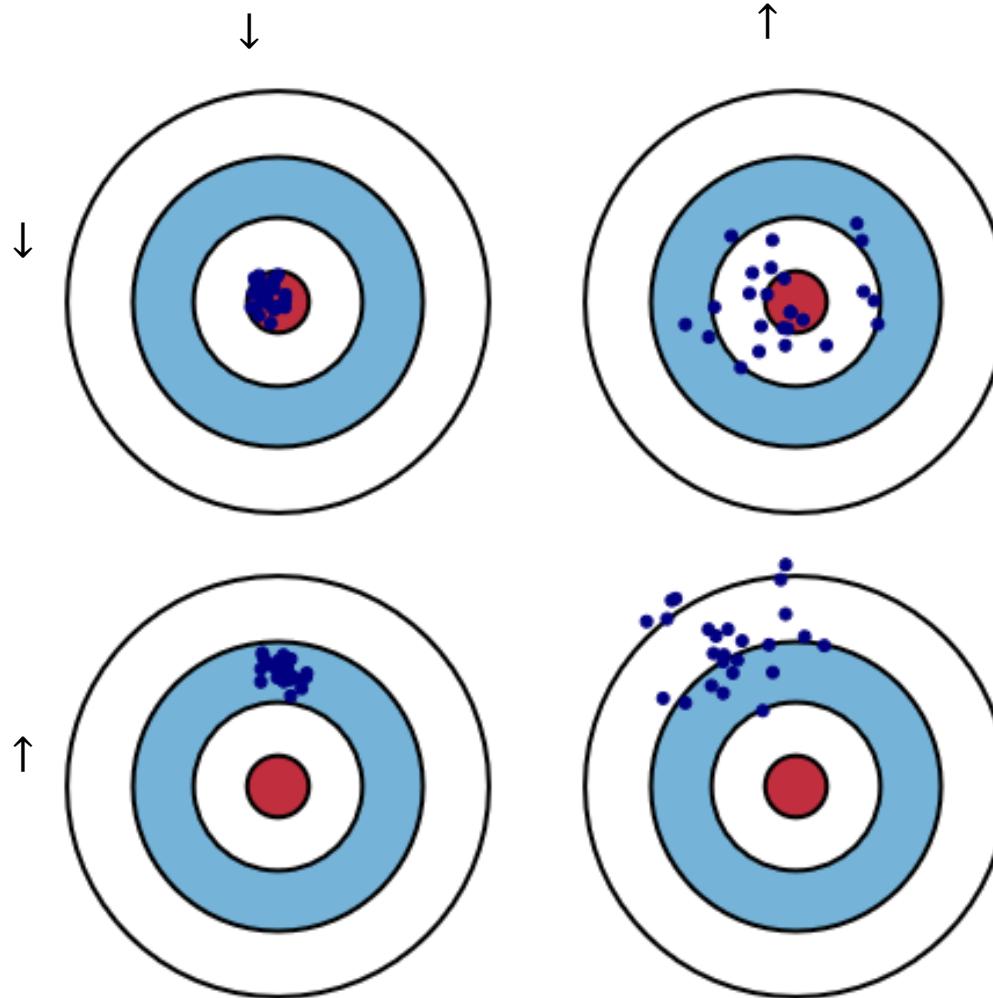
$$bias = \mathbb{E}_{\mathbf{X}} \left[(\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2 \right]$$

$$variance = \mathbb{E}_{\mathbf{X}} \left[\mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] \right]$$

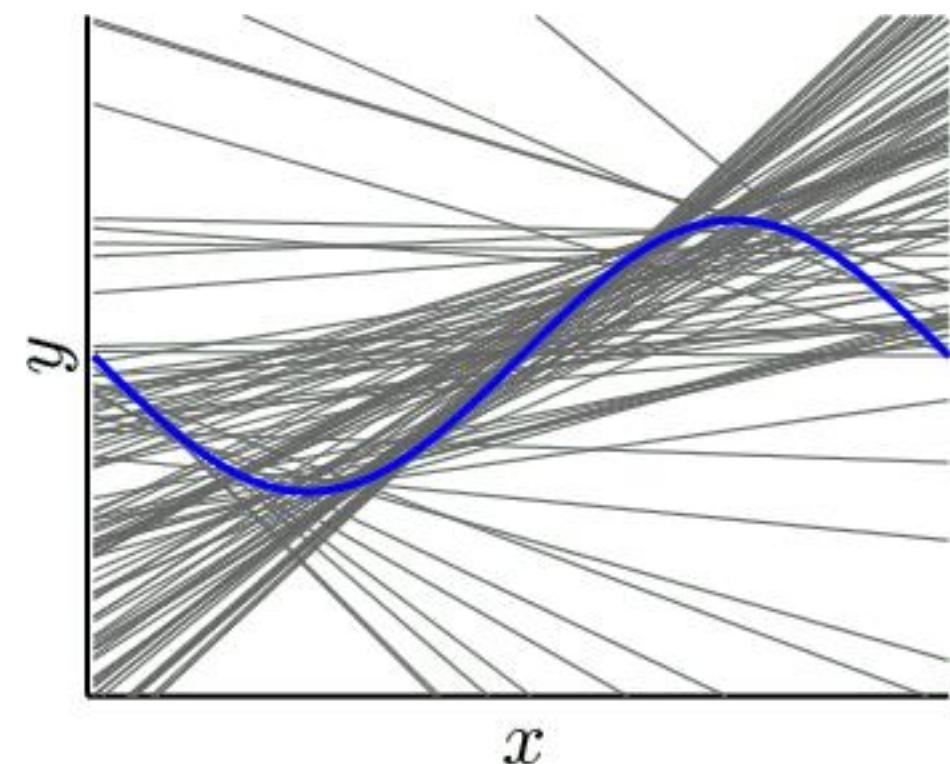
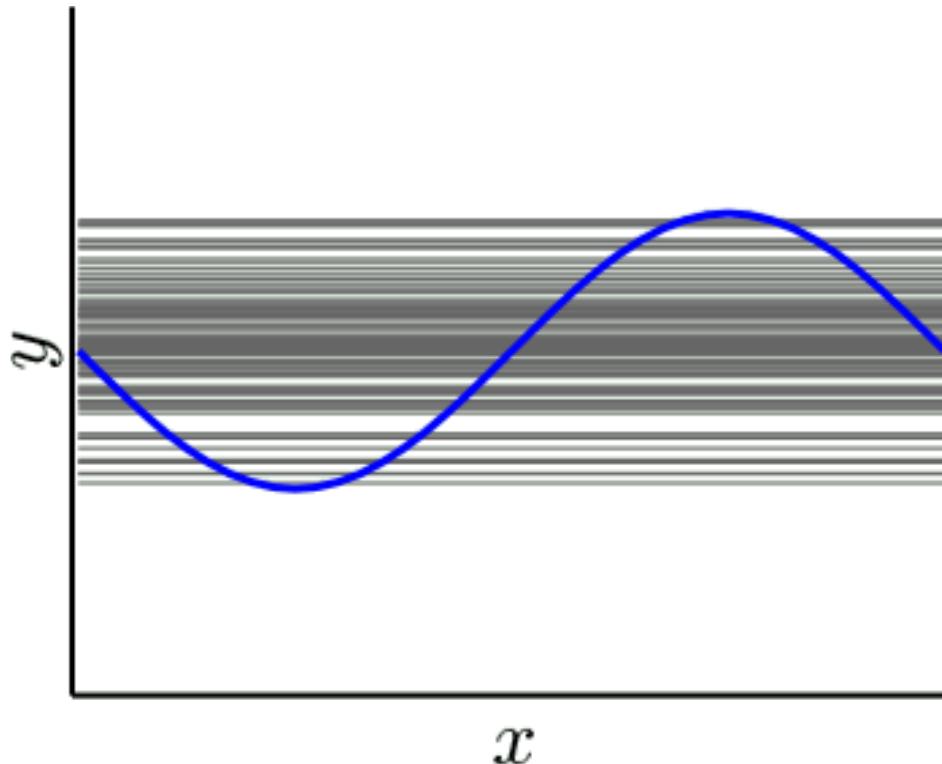
Bias and Variance

$$\text{variance} = \mathbb{E}_{\mathbf{X}} \left[\mathbb{E}_D \left[(h^D(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \right] \right]$$

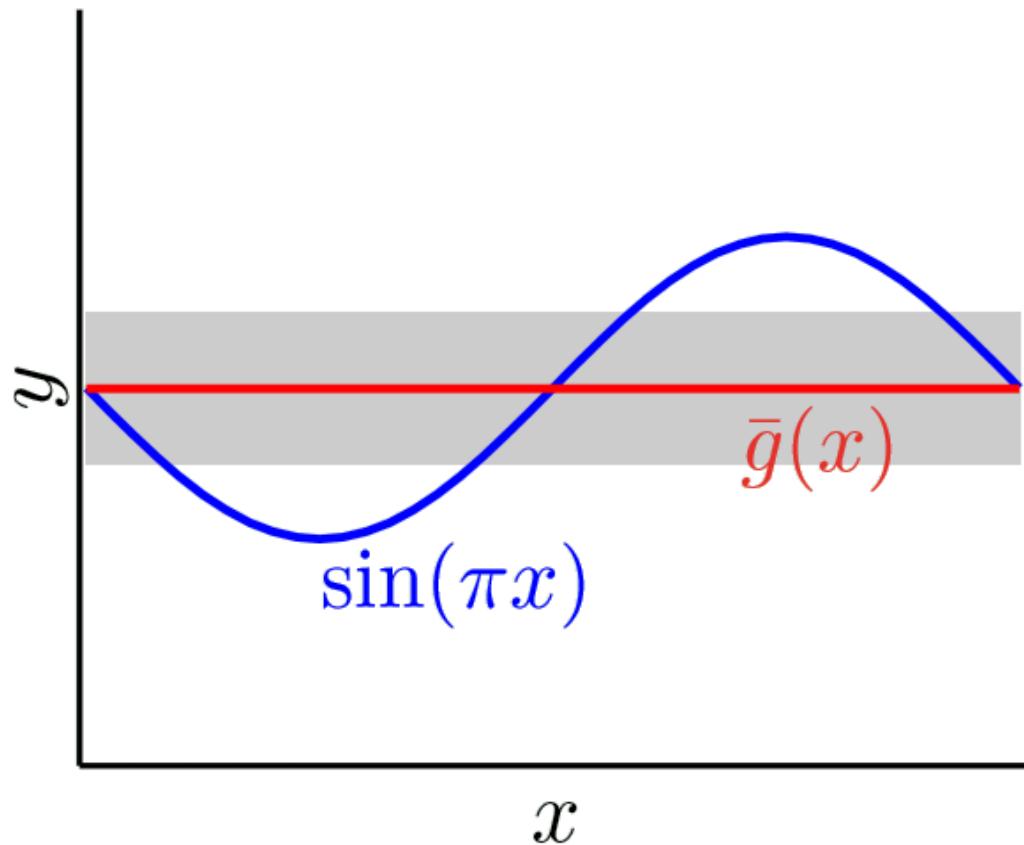
$$\text{bias} = \mathbb{E}_{\mathbf{X}} \left[(\bar{h}(\mathbf{x}) - f(\mathbf{x}))^2 \right]$$



Цель - синус

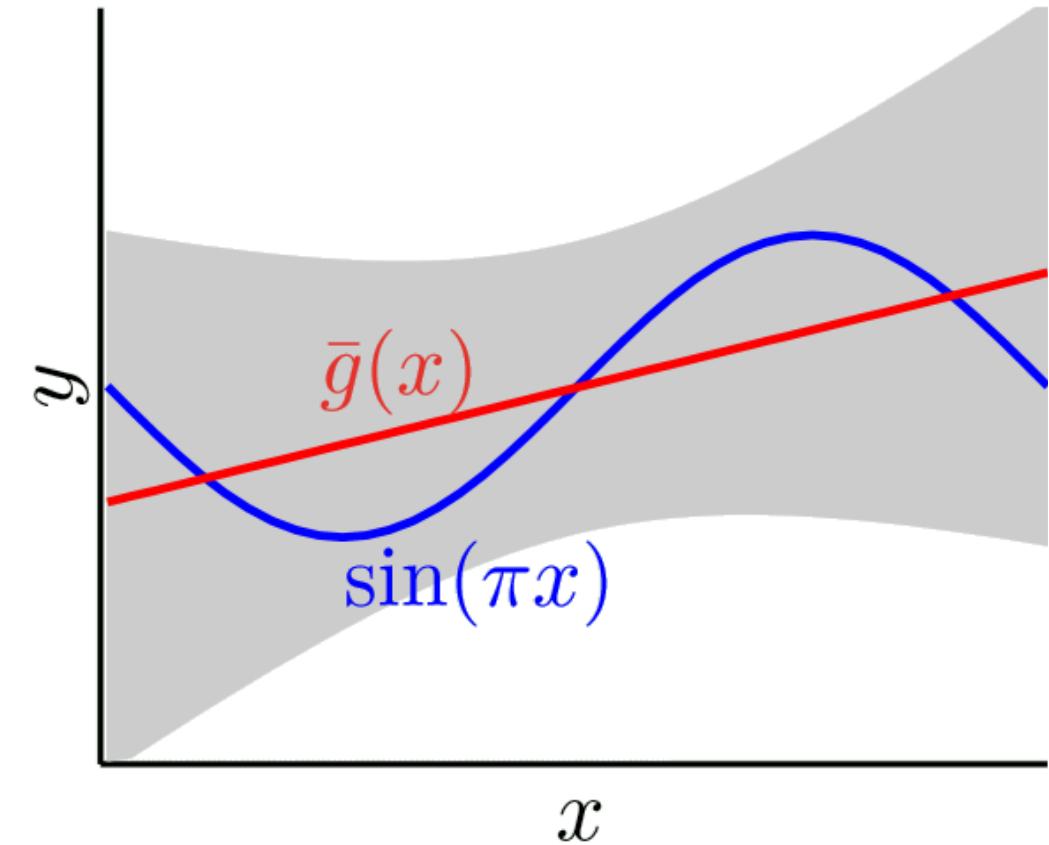


Цель - синус



bias = **0.50**

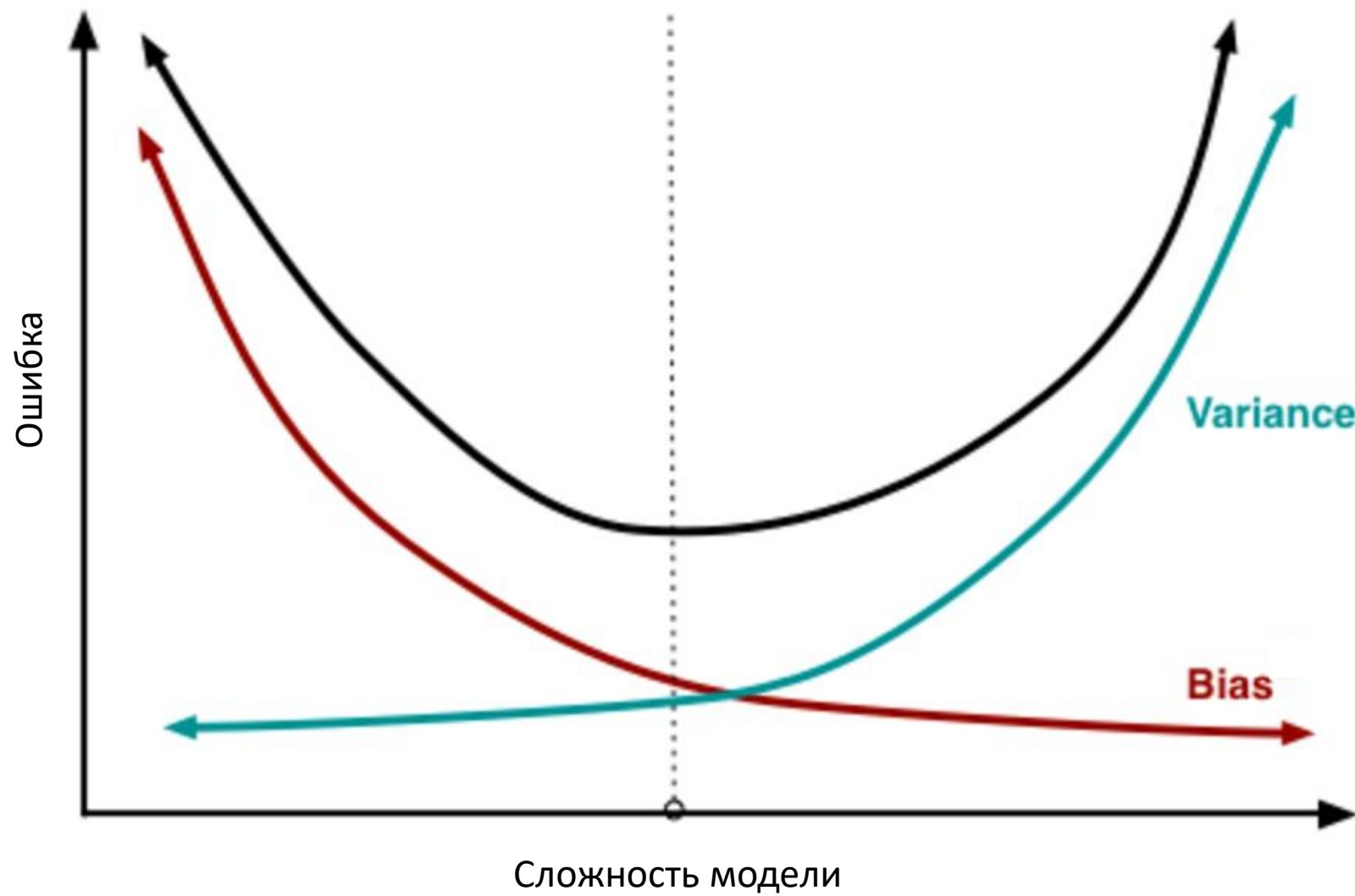
var = **0.25**



bias = **0.21**

var = **1.69**

Bias and Variance



Регуляризация регрессии

Регрессия. Регуляризация.

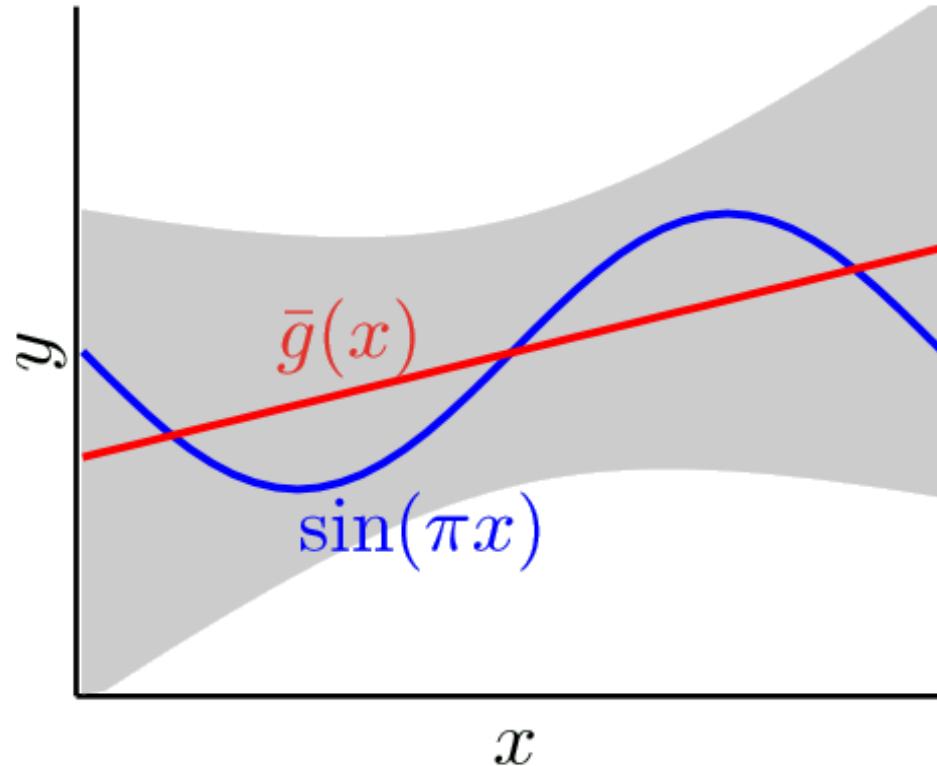
$$E_{in}(\mathbf{w}) = \frac{1}{N} \left\| \mathbf{X}\mathbf{w} - \mathbf{y} \right\|_2^2$$

$$E_{in}(\mathbf{w}) + \frac{\alpha}{N} \mathbf{w}^T \mathbf{w} = \frac{1}{N} \left((\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \alpha \mathbf{w}^T \mathbf{w} \right)$$

$$\nabla E_{in}(\mathbf{w}) = \frac{2}{N} \left(\mathbf{X}^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \alpha \mathbf{w} \right) = 0$$

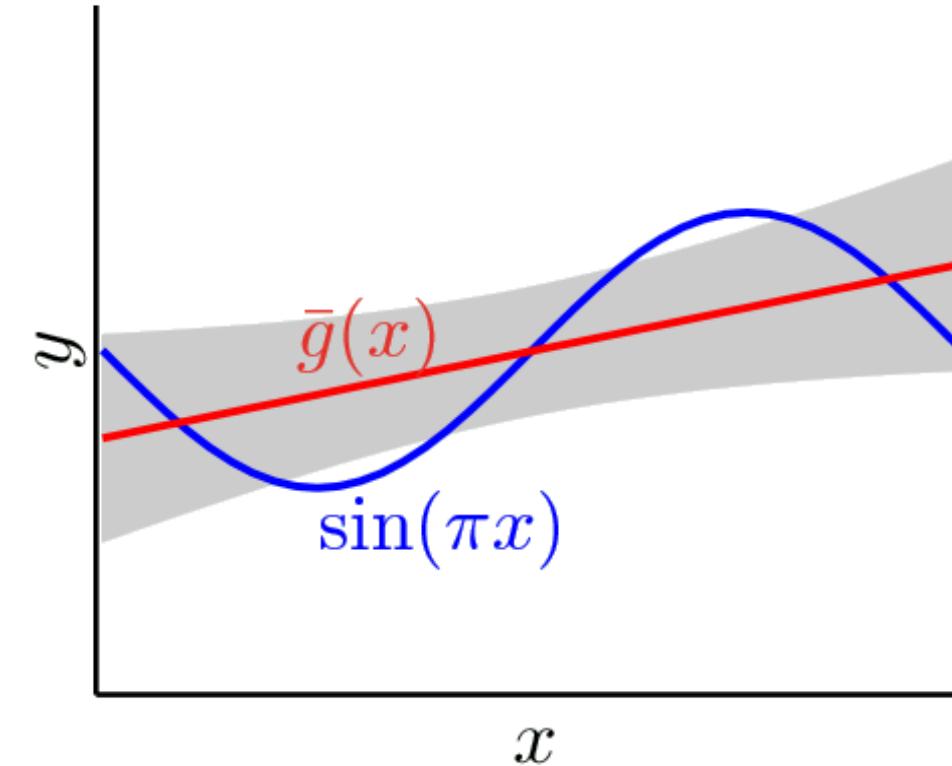
$$\mathbf{w} = (\mathbf{X}^T \mathbf{X} + \alpha \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

Регуляризация и синус



bias = **0.21**

var = **1.69**



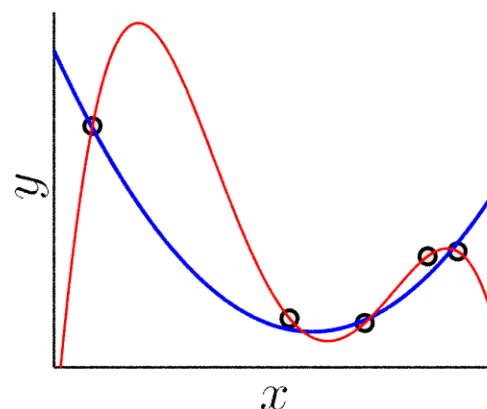
bias = **0.23**

var = **0.33**

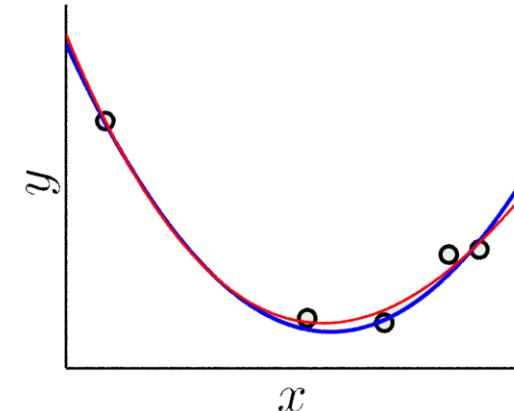
Переобучение и недообучение

$$E_{in}(\mathbf{w}) + \frac{\alpha}{N} \mathbf{w}^T \mathbf{w}$$

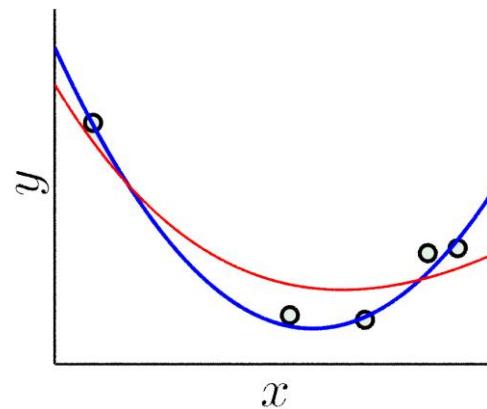
$$\alpha = 0$$



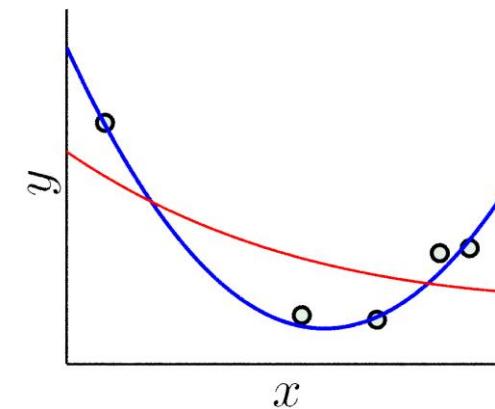
$$\alpha = 0.0001$$



$$\alpha = 0.01$$



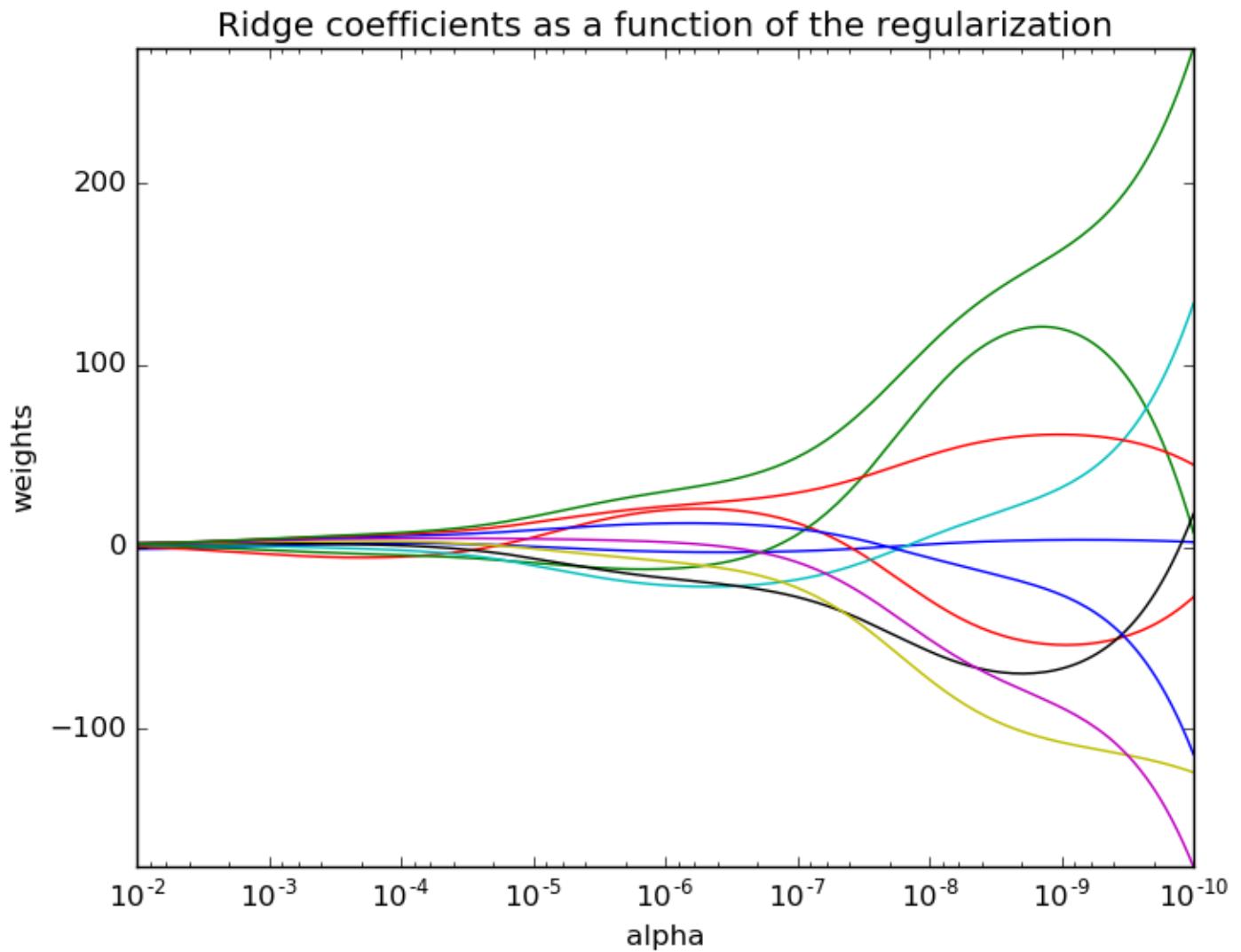
$$\alpha = 1$$



Гребневая (Ridge) регрессия

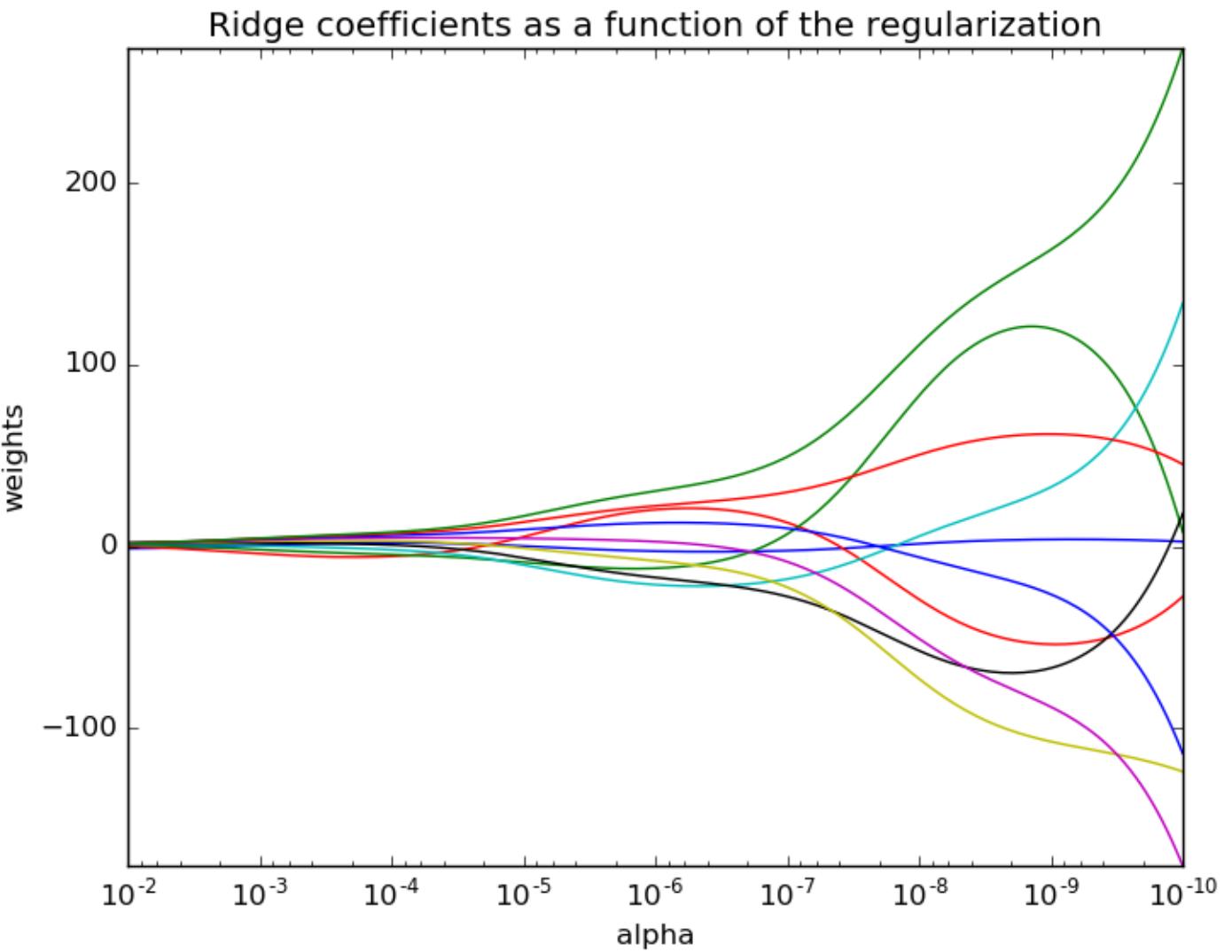
$$E_{in} = \frac{1}{N} \|Xw - Y\|_2^2 + \frac{\alpha}{N} \|w\|_2^2$$

$$w = (X^T X + \alpha I)^{-1} X^T Y$$



Гребневая (Ridge) регрессия

$$E_{in} = \frac{1}{2N} \|Xw - Y\|_2^2 + \alpha \|w\|_2^2$$

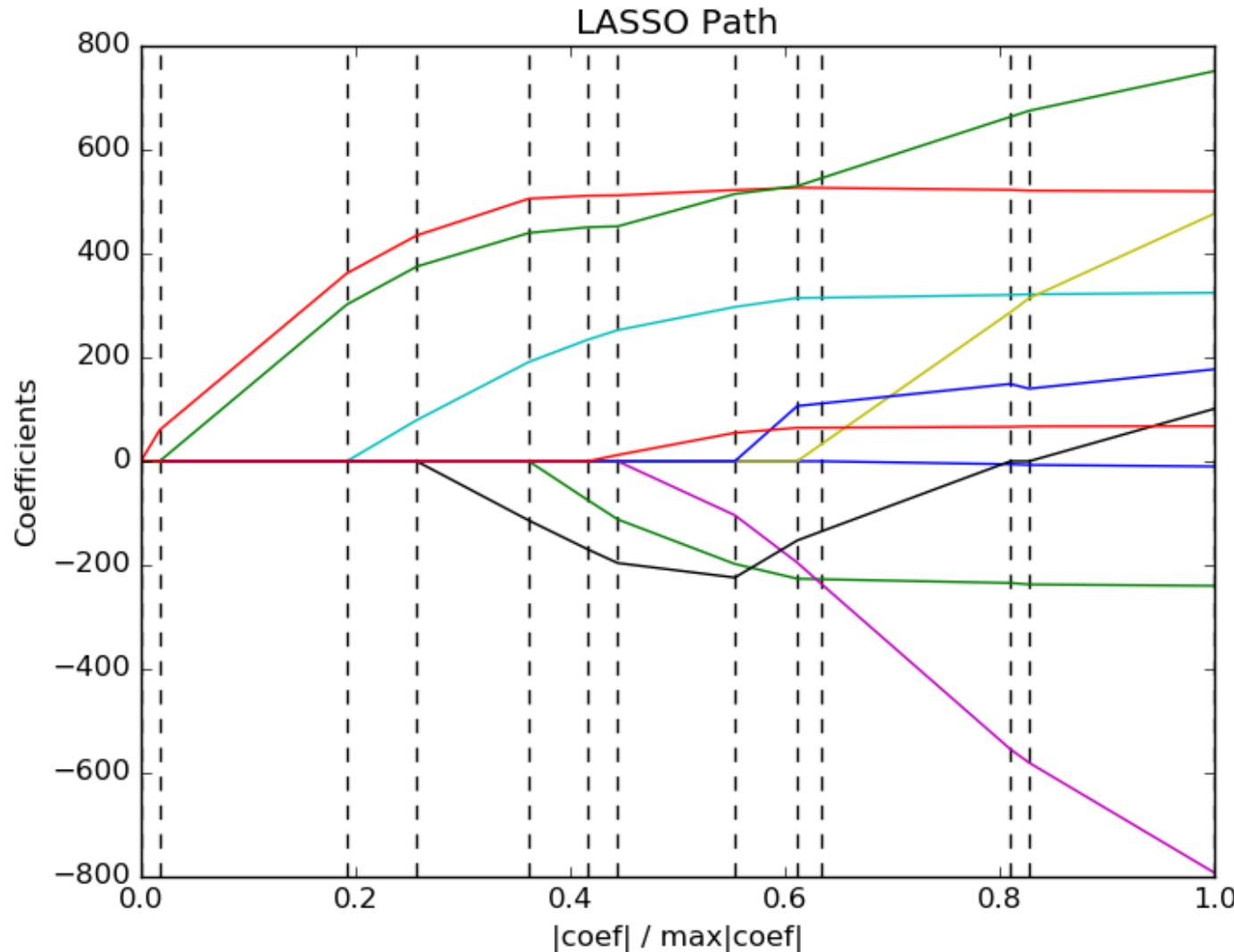


LASSO

(Least Absolute Shrinkage and Selection Operator)

$$E_{in} = \frac{1}{2N} \|Xw - Y\|_2^2 + \alpha \|w\|_1$$

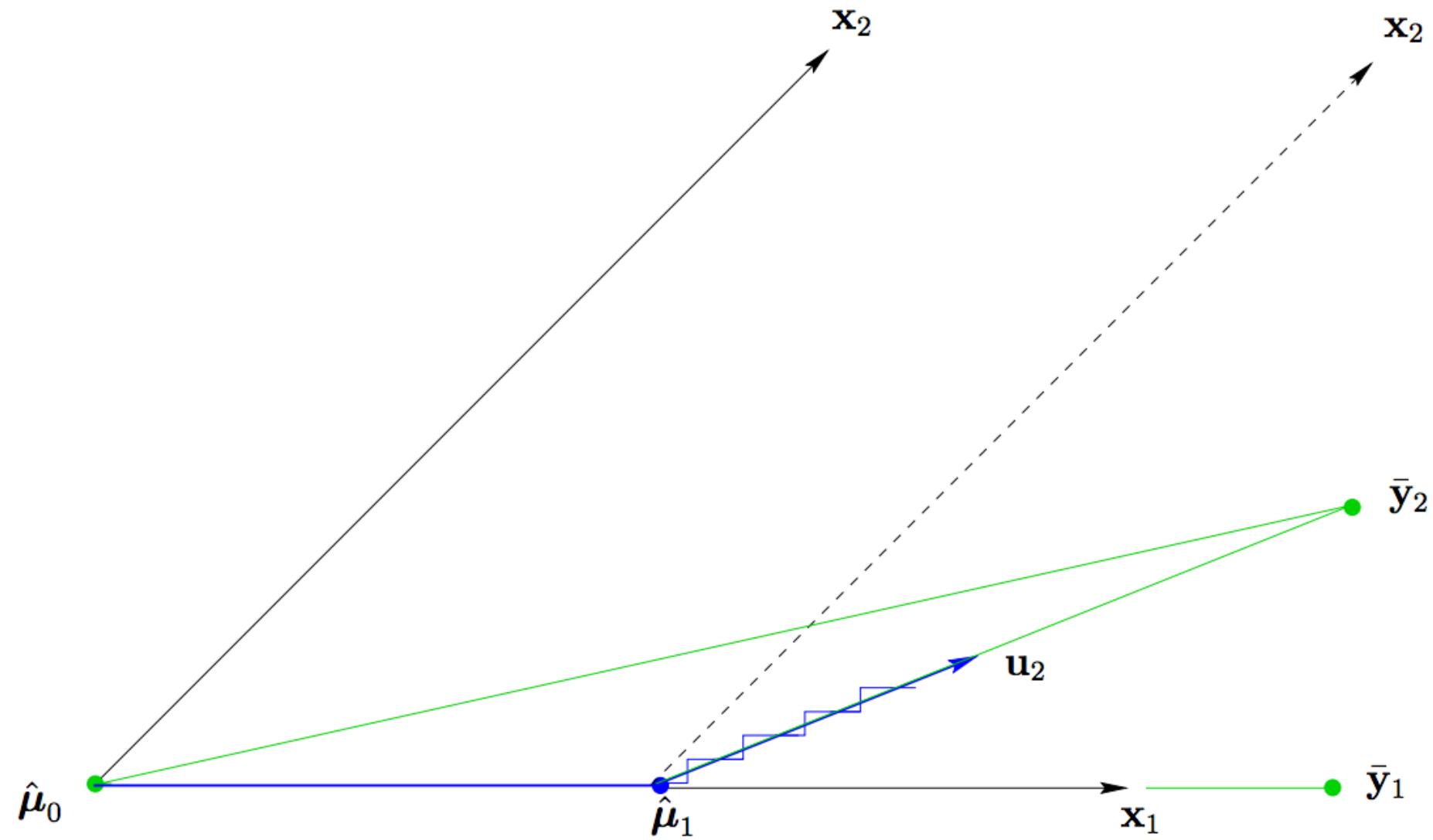
Применяется покоординатный спуск или LARS.



LARS (Least Angle Regression)

1. Возьмем x_i наиболее сильно коррелирующий с y .
2. Сдвигаем коэффициент β_1 при x_i в сторону корреляции пока корреляция x_i с остаточным значением $r = y - \hat{y}$ остается максимальной.
3. В тот момент, как корреляция перестает быть максимальной, появляется еще один (или несколько) признак x_j с такой же корреляцией.
4. Вводим коэффициент β_2 при $(x_i \pm x_j)$.
5. $\rightarrow 2$

LARS

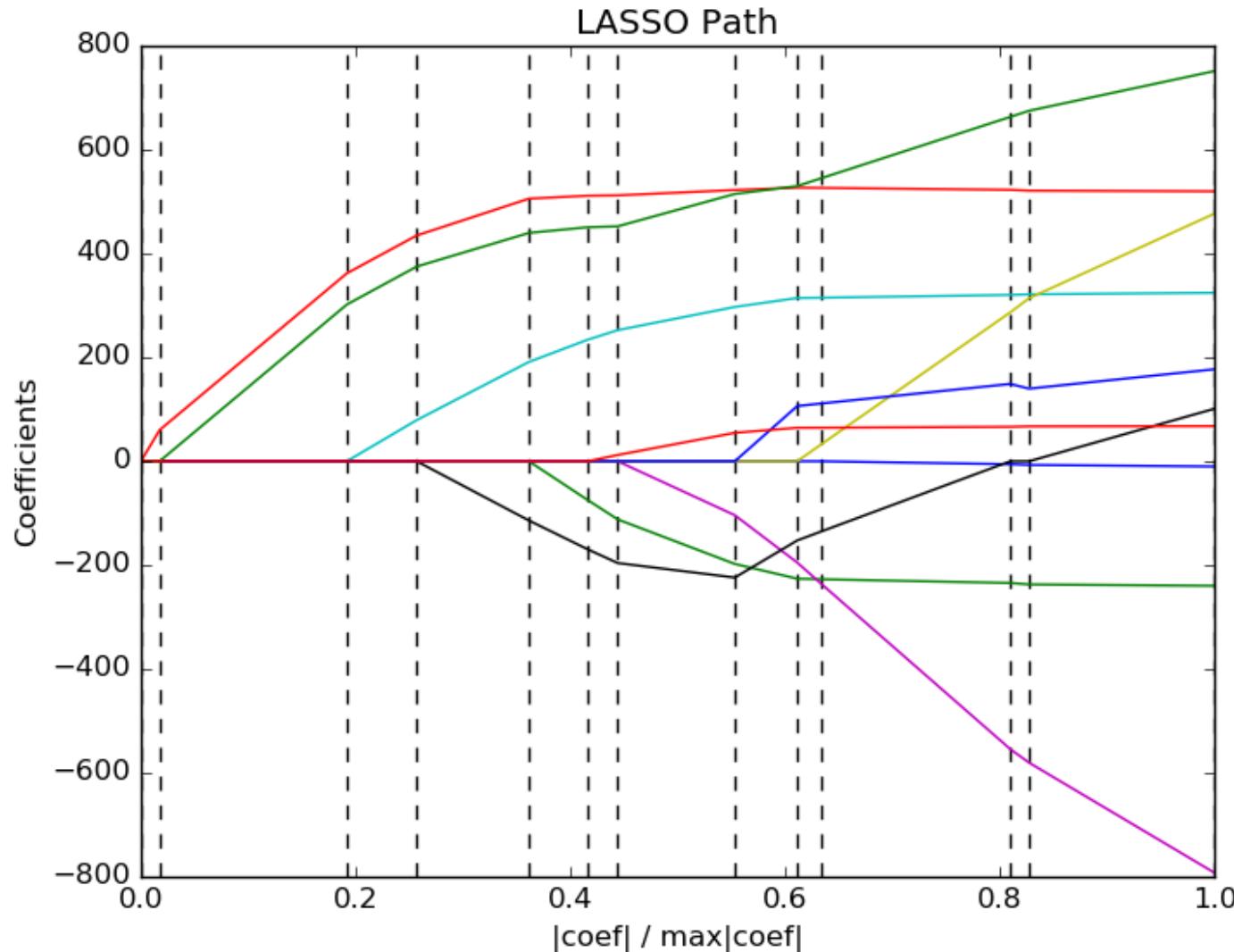


LASSO

(Least Absolute Shrinkage and Selection Operator)

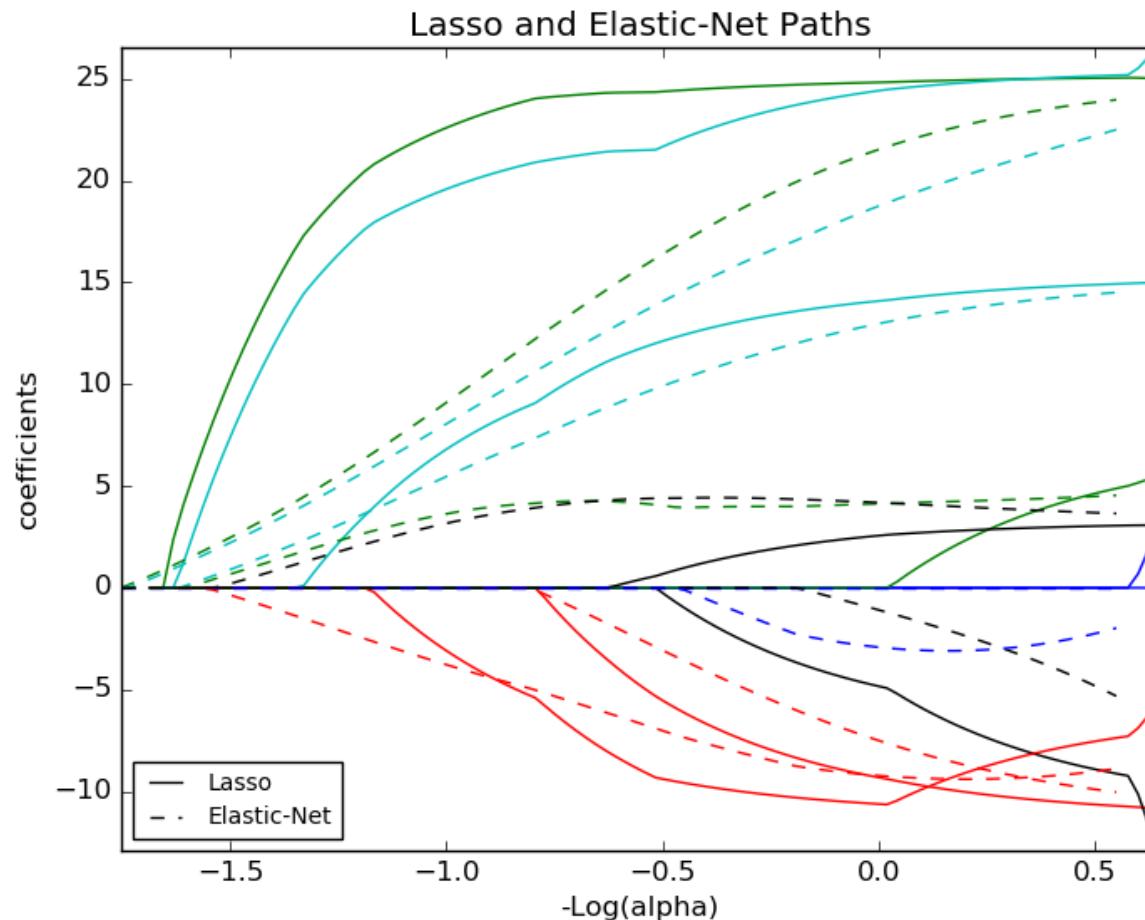
$$E_{in} = \frac{1}{2N} \|Xw - Y\|_2^2 + \alpha \|w\|_1$$

Применяется покоординатный спуск или LARS.



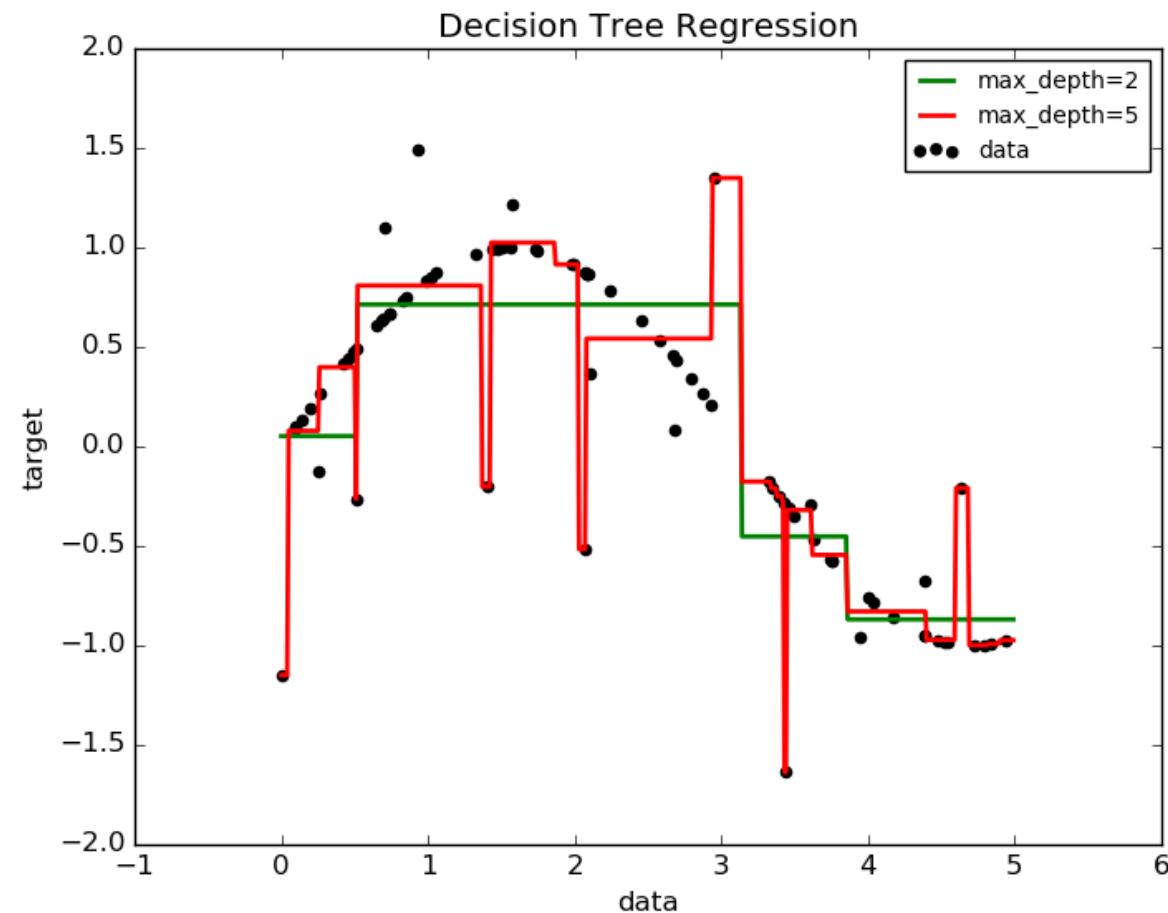
Elastic Net

$$E_{in} = \frac{1}{2N} \left\| Xw - Y \right\|_2^2 + \frac{1}{2} \alpha (1 - l1_{ratio}) \left\| w \right\|_2^2 + \alpha (l1_{ratio}) \left\| w \right\|_1$$



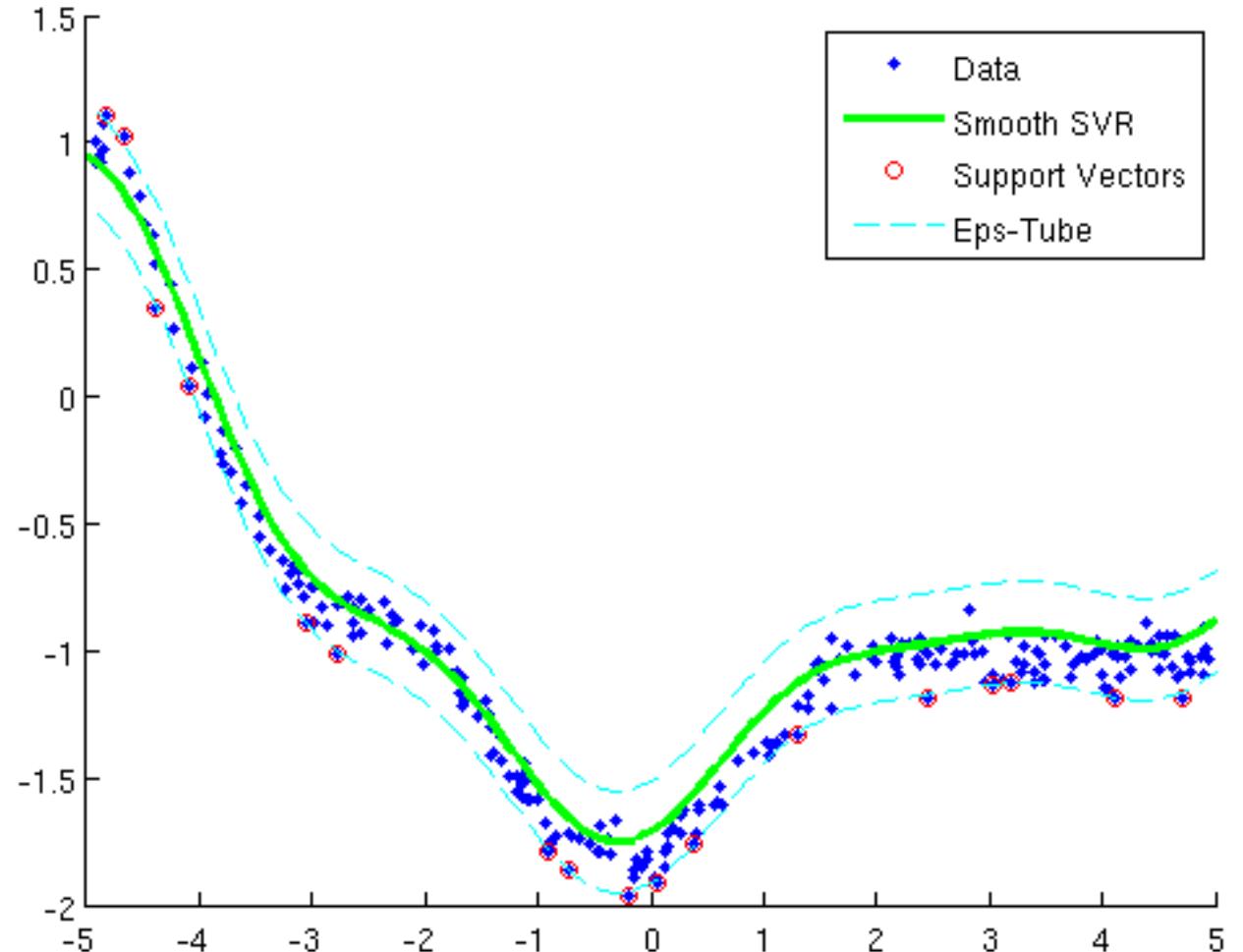
CART

$$\nabla V(R) = \frac{1}{N^2} \sum \sum \frac{1}{2} (y_i - y_j)^2 - \left(\frac{1}{N_1^2} \sum \sum \frac{1}{2} (y'_i - y'_j)^2 + \frac{1}{N_2^2} \sum \sum \frac{1}{2} (y''_i - y''_j)^2 \right)$$



Support Vector Regression Machine

$$\left\{ \begin{array}{l} \frac{1}{2} \|w\|^2 + C \sum (\xi_i + \xi_i^*) \rightarrow \min \\ y_i - w^T x_i - b \leq \epsilon + \xi_i \\ w^T x_i + b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq \epsilon \end{array} \right.$$



R^2 -score (коэффициент детерминации)

Абсолютное значение ошибки довольно неинформативно.

Удобная оценка результата регрессии:

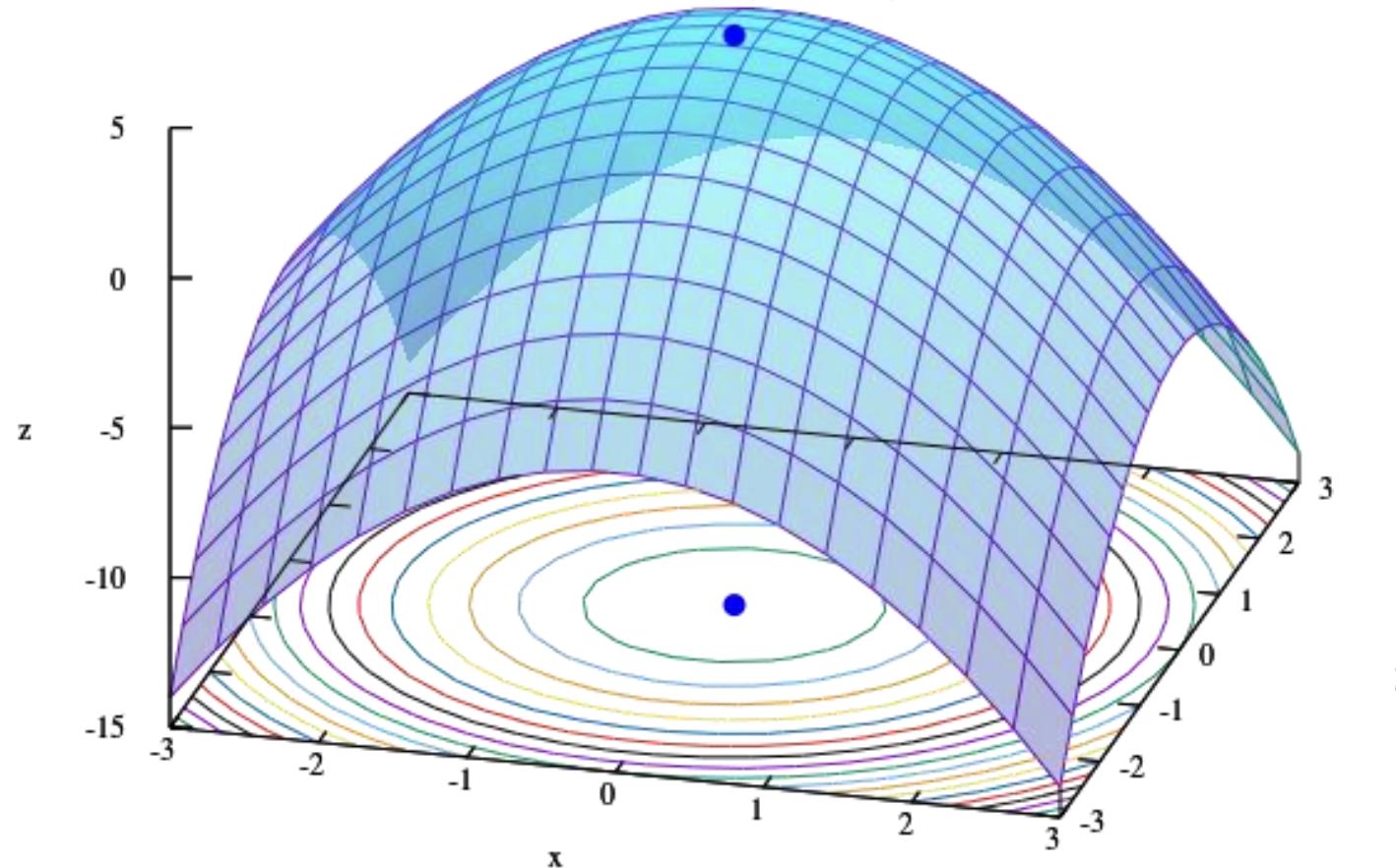
$$R^2 = 1 - \frac{u}{v}$$

$$u = \sum(h(\mathbf{x}_i) - y_i)^2 \quad v = \sum(\bar{y} - y_i)^2 \quad \bar{y} = \frac{1}{N} \sum y_i$$

Стохастическая оптимизация

Для дифференцируемых функций – градиентный спуск

Для недифференцируемых или неизвестных функций – локальный/глобальный поиск



Глобальный поиск vs локальный поиск

Глобальный поиск – пытаемся посмотреть на все пространство состояний и найти лучшее (Монте-Карло, случайное блуждание).

Локальный поиск – используем текущее состояние системы и двигаемся от него.

Гибридные алгоритмы – гибриды локального и глобального поиска.

Кросс-энтропийный метод

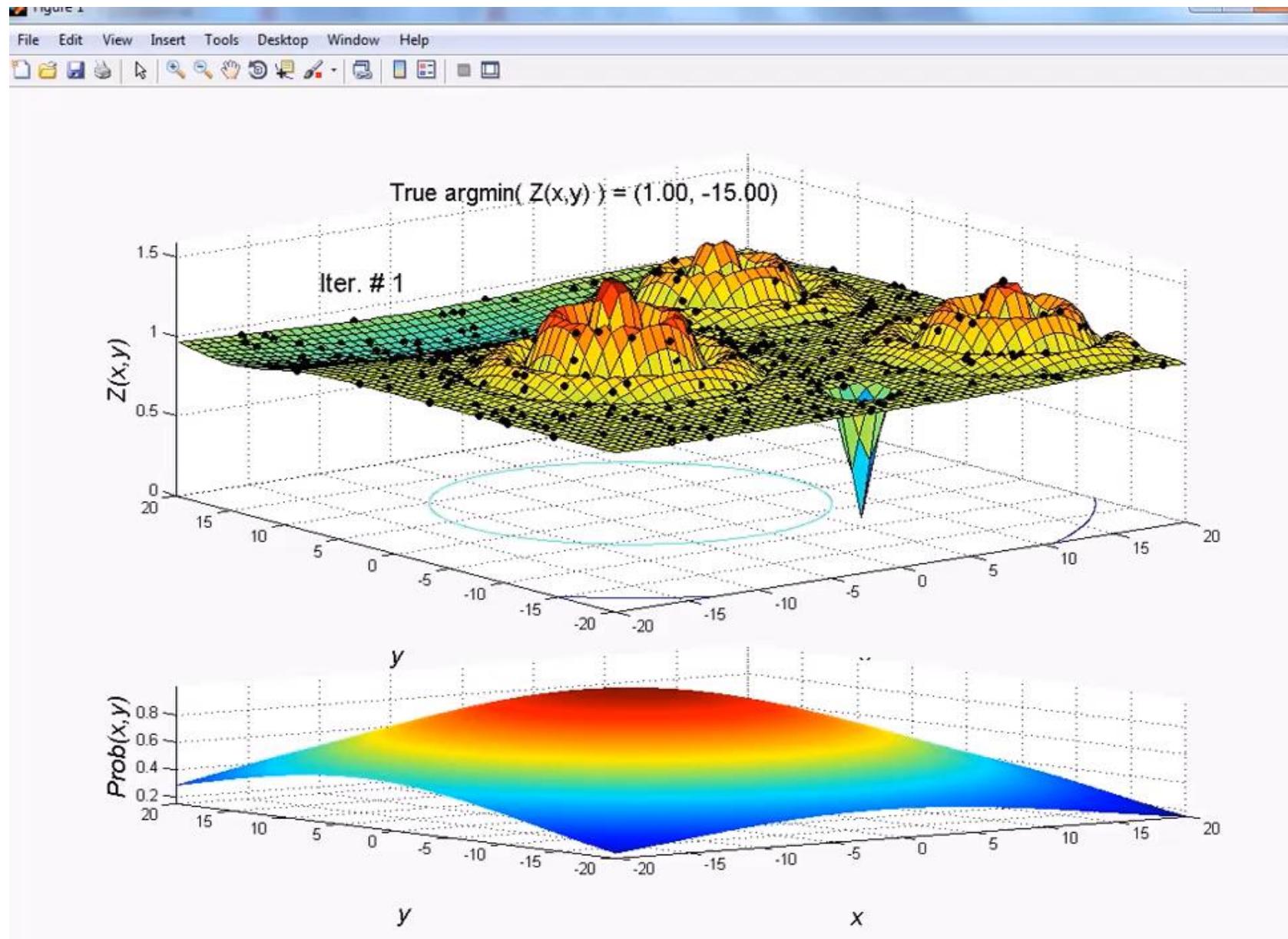
1. На шаге $t = 1$ выбираем начальный вектор параметров распределения ν_0 .
2. Сгенерируем случайную выборку $x_1 \dots x_N$ по распределению $f(x; \nu_{t-1})$.
3. Посчитаем новый вектор параметров основываясь на k лучших (по критерию H) примерах из выборки:

$$\nu_t = \arg \max_u \frac{1}{k} \sum_{x_i \in \text{best } k} H(x_i) \frac{f(x_i; u)}{f(x_i; \nu_{t-1})} \log f(X_i; \nu_{t-1})$$

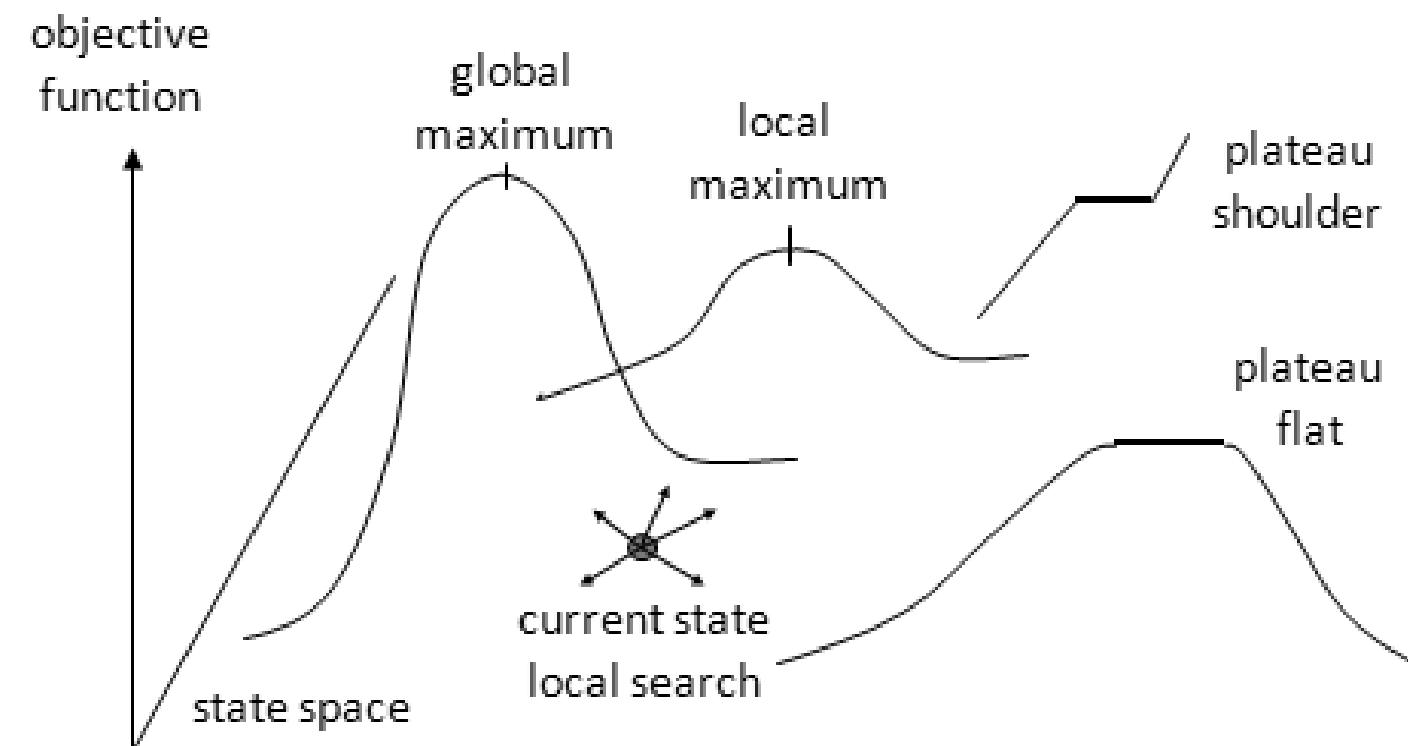
4. Если сошлось или достигли условия останавливаемся, иначе $\rightarrow 2$.

*для некоторых распределений $\arg\max_u$ можно получить аналитически.

Кросс энтропийный поиск



Hill Climbing



Аналог **Gradient Descent**.

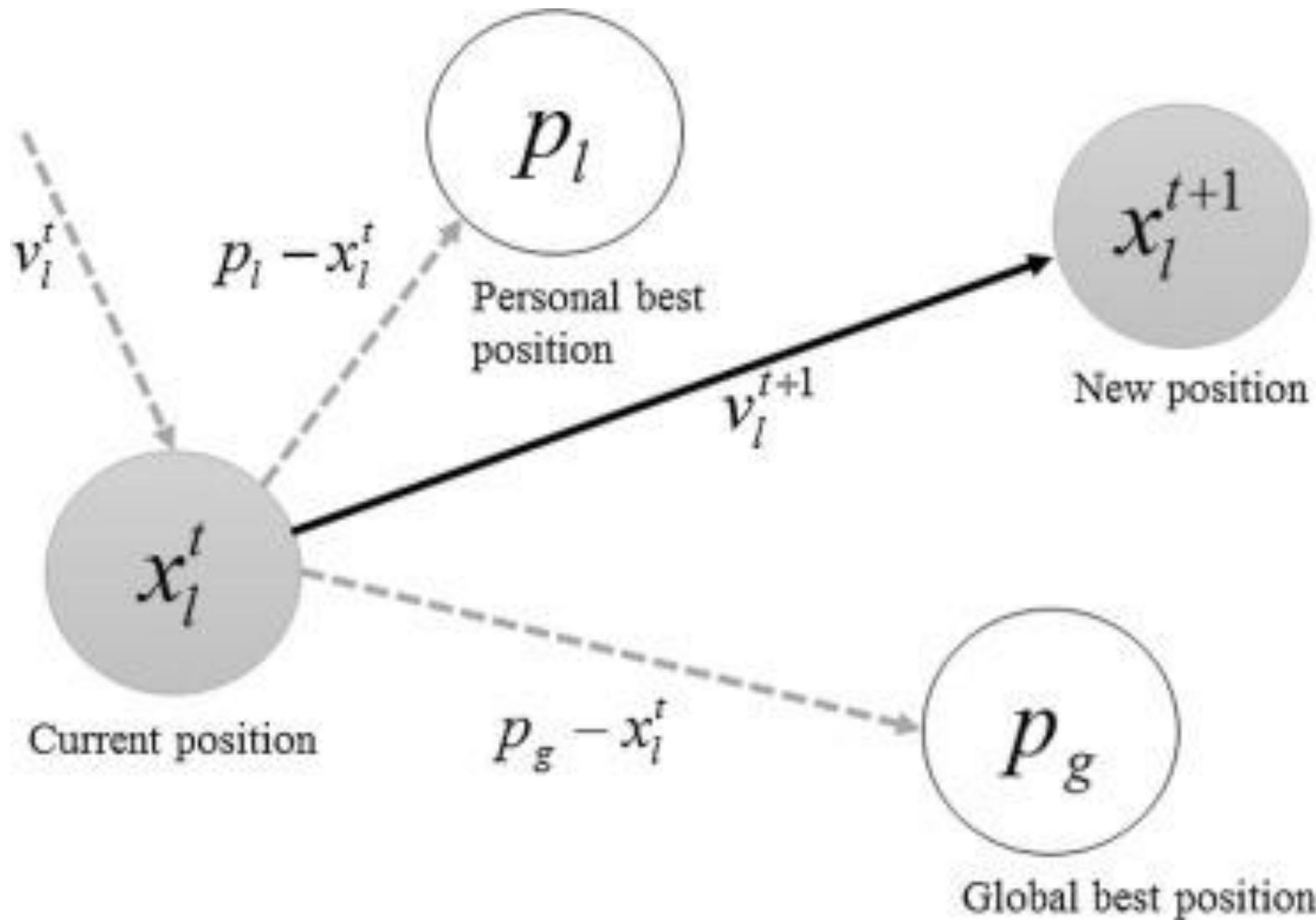
Считаем оптимизируемую функцию по нескольким возможным смещениям (например – изменения одной координаты/параметра) и идем в сторону наибольшего подъема.

Модификация – **Stochastic Hill Climbing**:
Идем с вероятностью, зависящей от значения функции (например – по softmax).

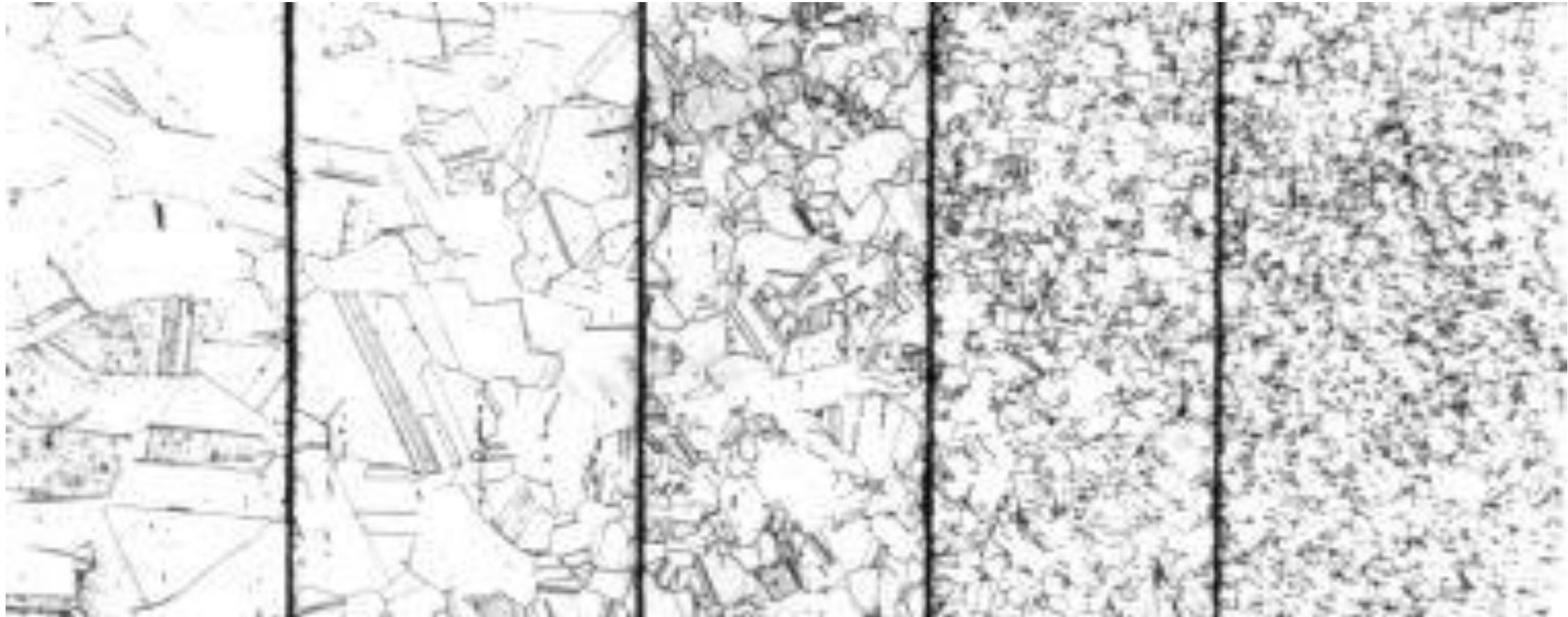
Модификация – **Tabu Search**:
Запоминаем последние несколько позиций и в них не возвращаемся.

Модификация – **Particle swarm optimization**:
Запускаем много агентов, которые могут обмениваться информацией.

Particle swarm optimization



Отжиг (Annealing)



Структура металла с отжигом и без

Отжиг (Simulated Annealing)

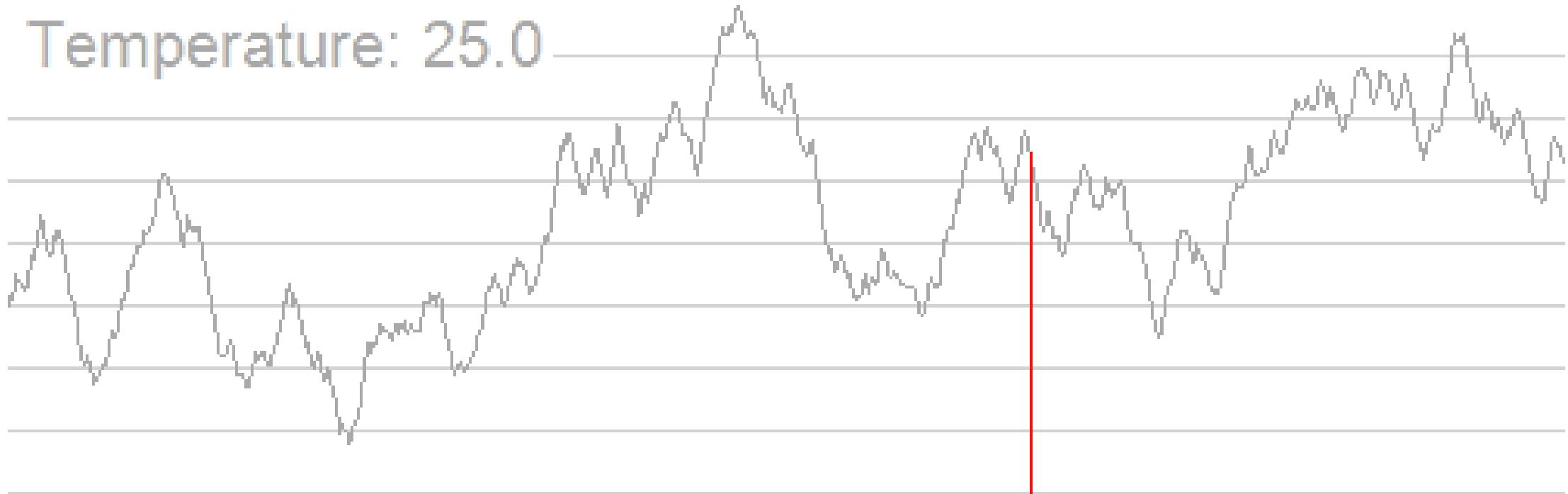
Введем в систему «температуру»:

Например, для Softmax:

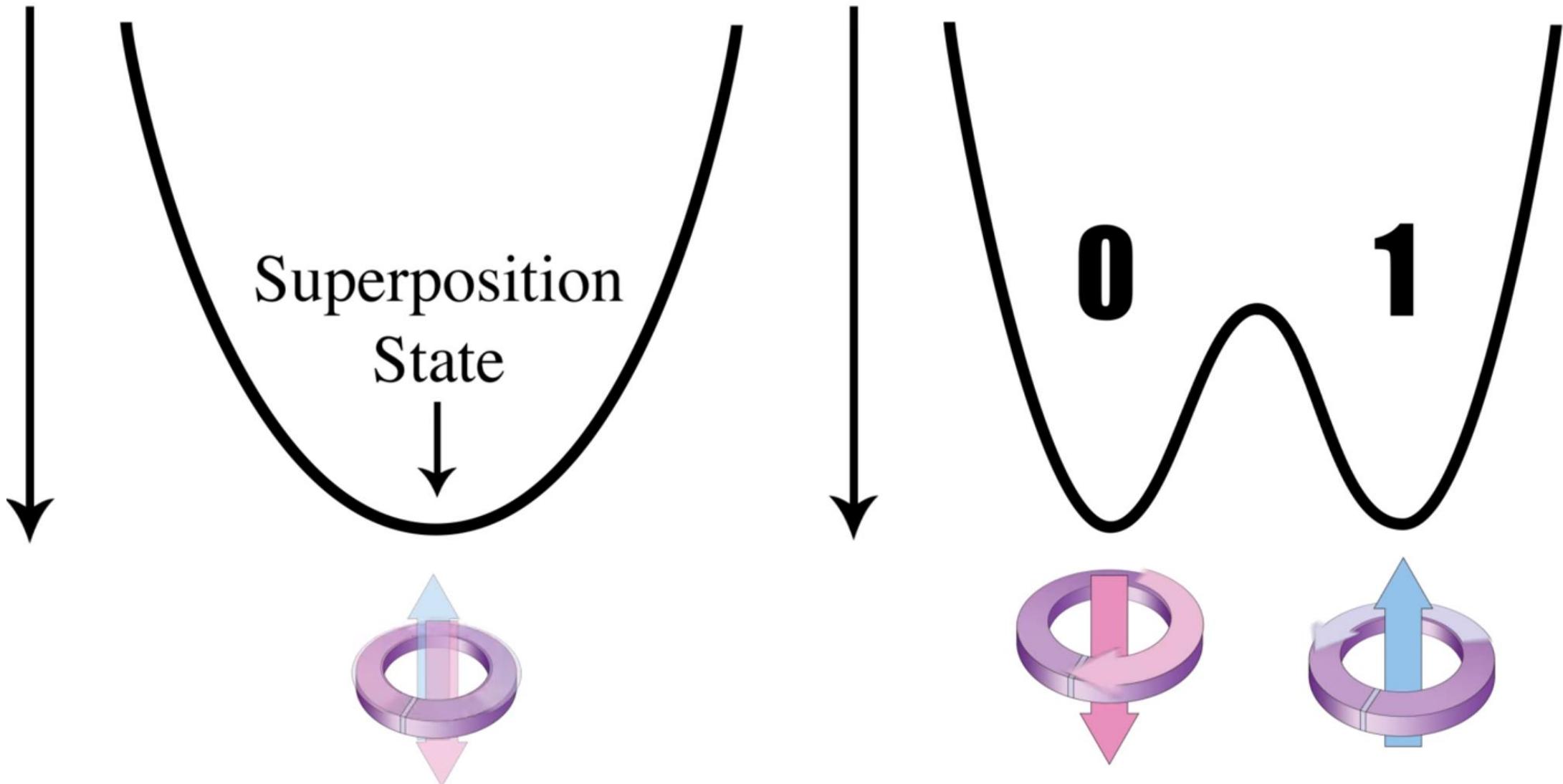
$$P(s_i) = \frac{e^{\frac{\Delta E(s_i)}{T}}}{\sum e^{\frac{\Delta E(s_j)}{T}}}$$

		T - Температура			
		10000	10	1	0.1
ΔE	P(ΔE)	P(ΔE)	P(ΔE)	P(ΔE)	
10	0.25016	0.43944	0.99325	1	
5	0.25004	0.26653	0.00669	1.9287E-22	
0	0.24991	0.16166	4.51E-05	3.7200E-44	
-2	0.24986	0.13235	6.1E-06	7.6676E-53	

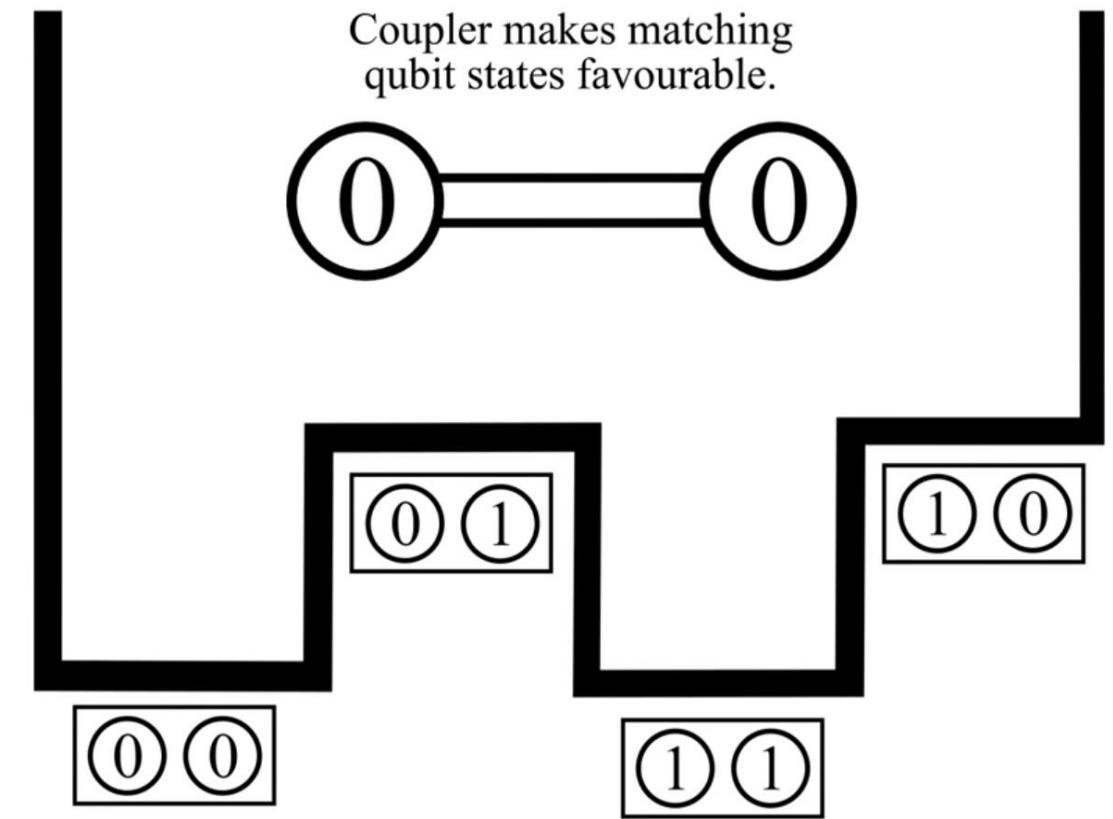
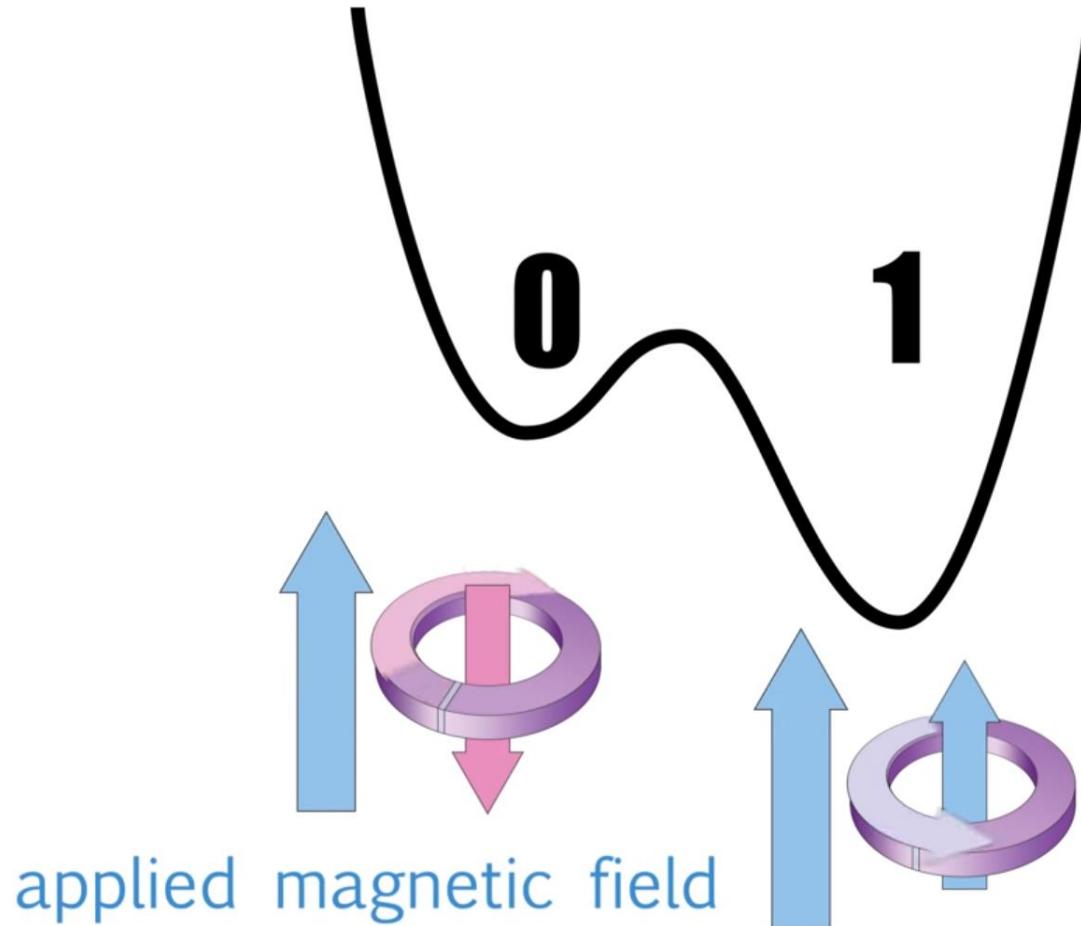
Отжиг (Simulated Annealing)



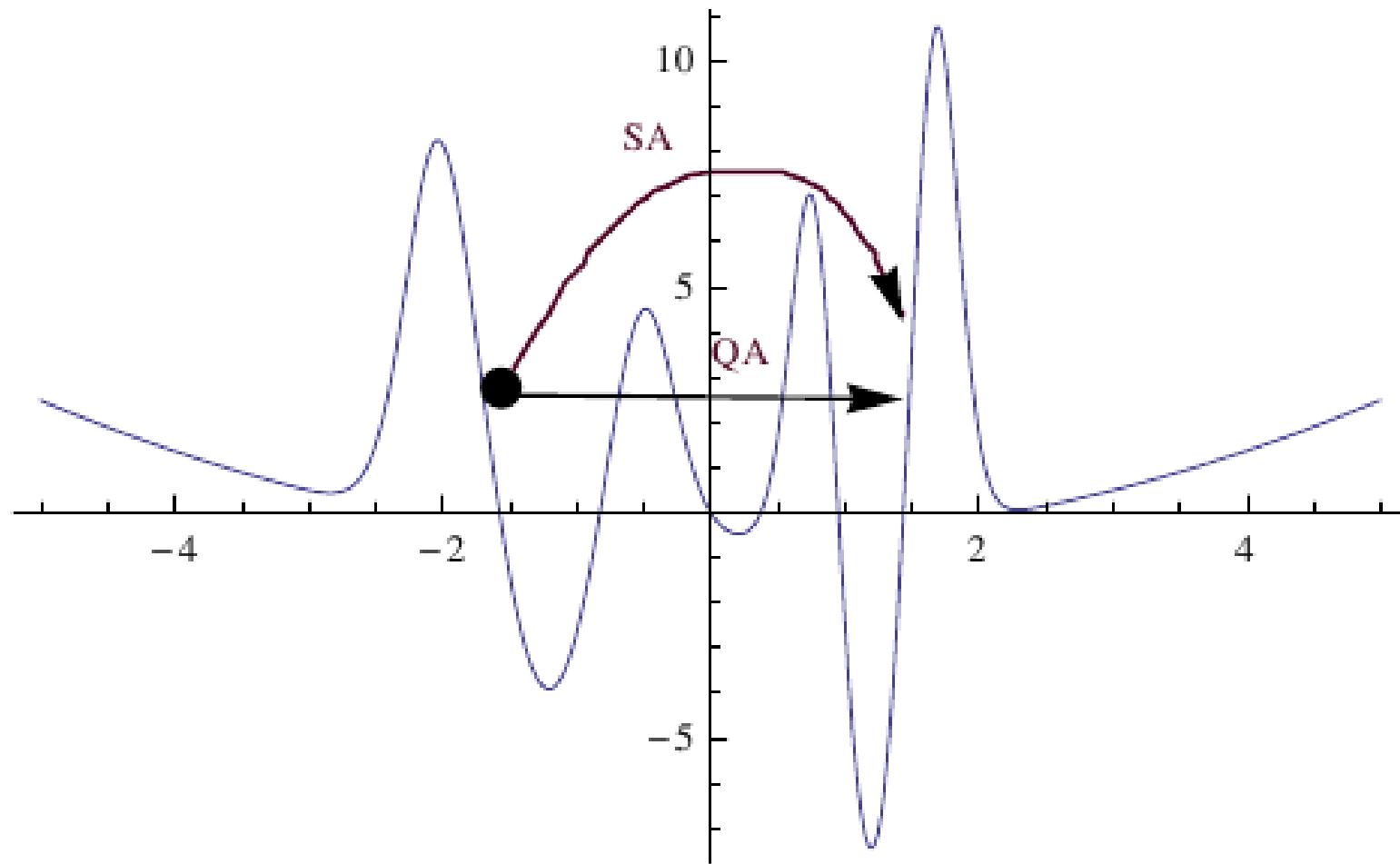
Квантовый отжиг



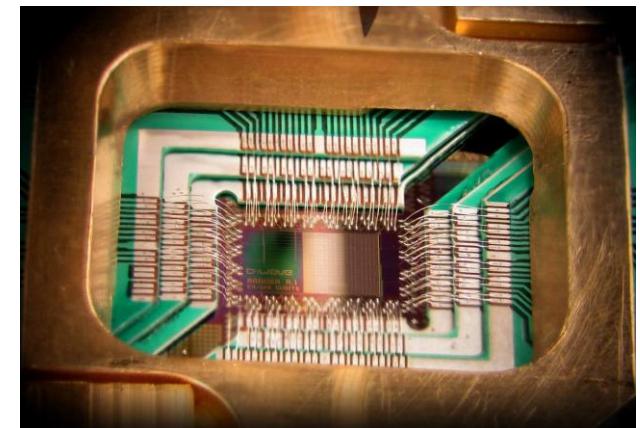
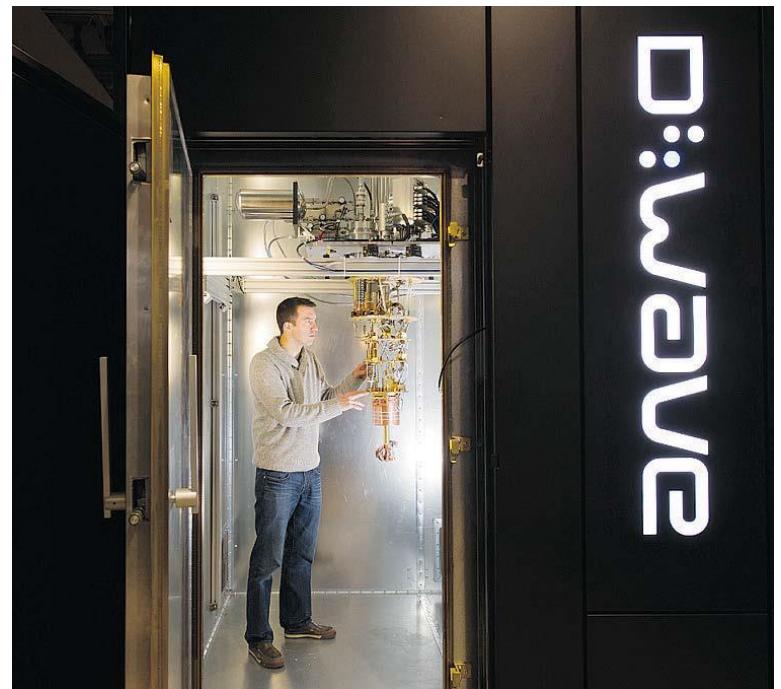
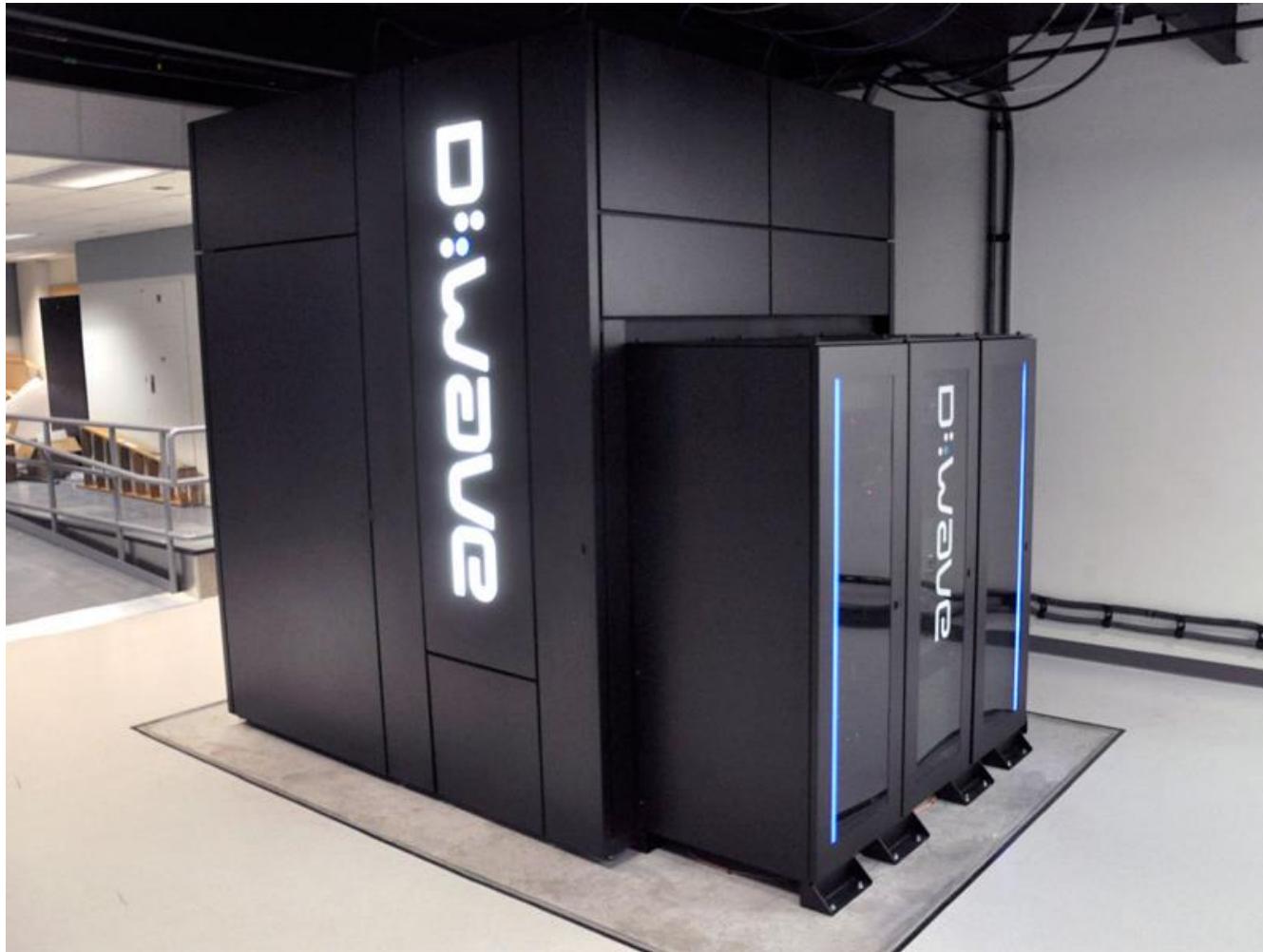
Magnets and couplers



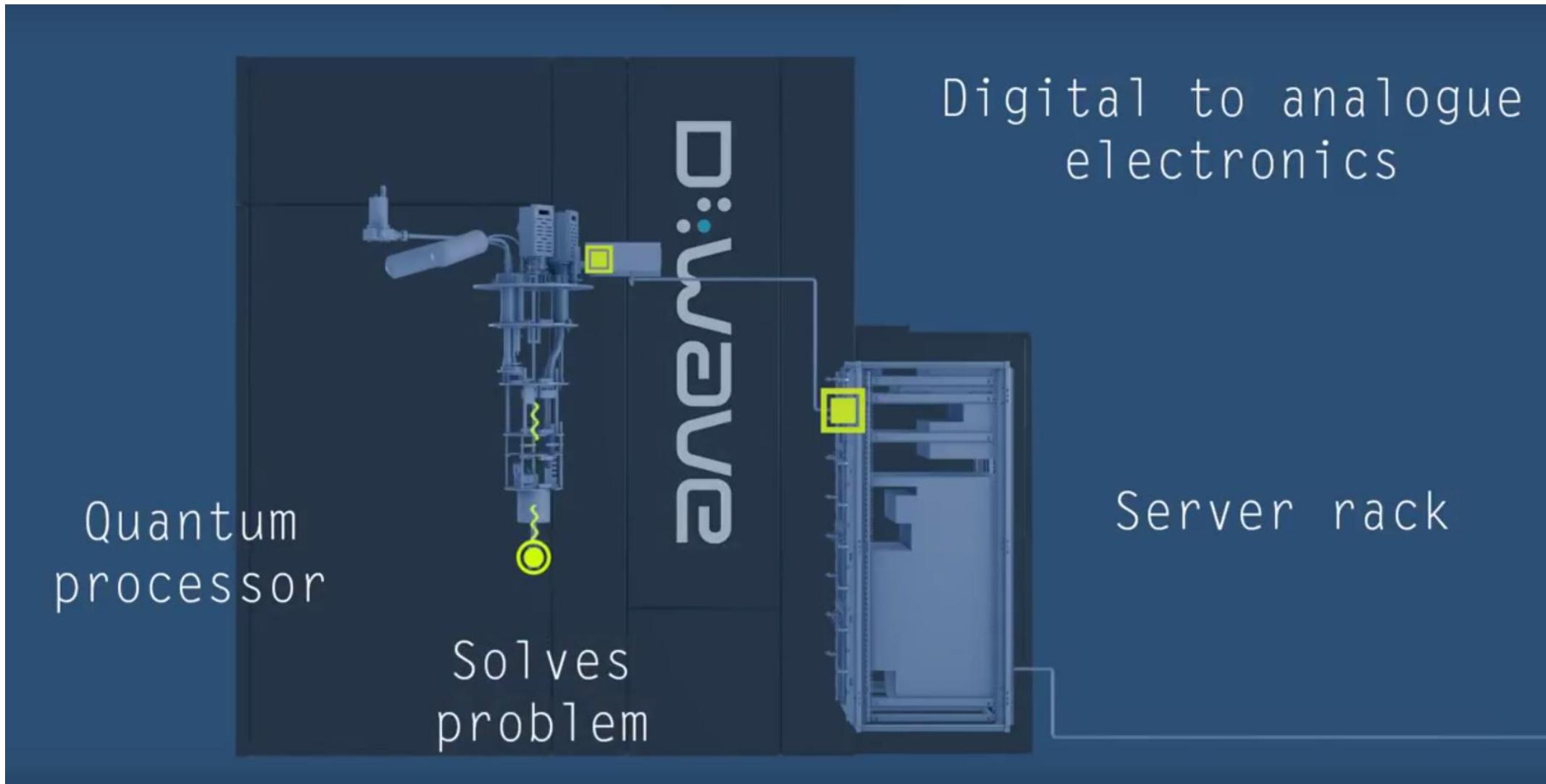
Туннельный эффект



D-Wave



D-Wave

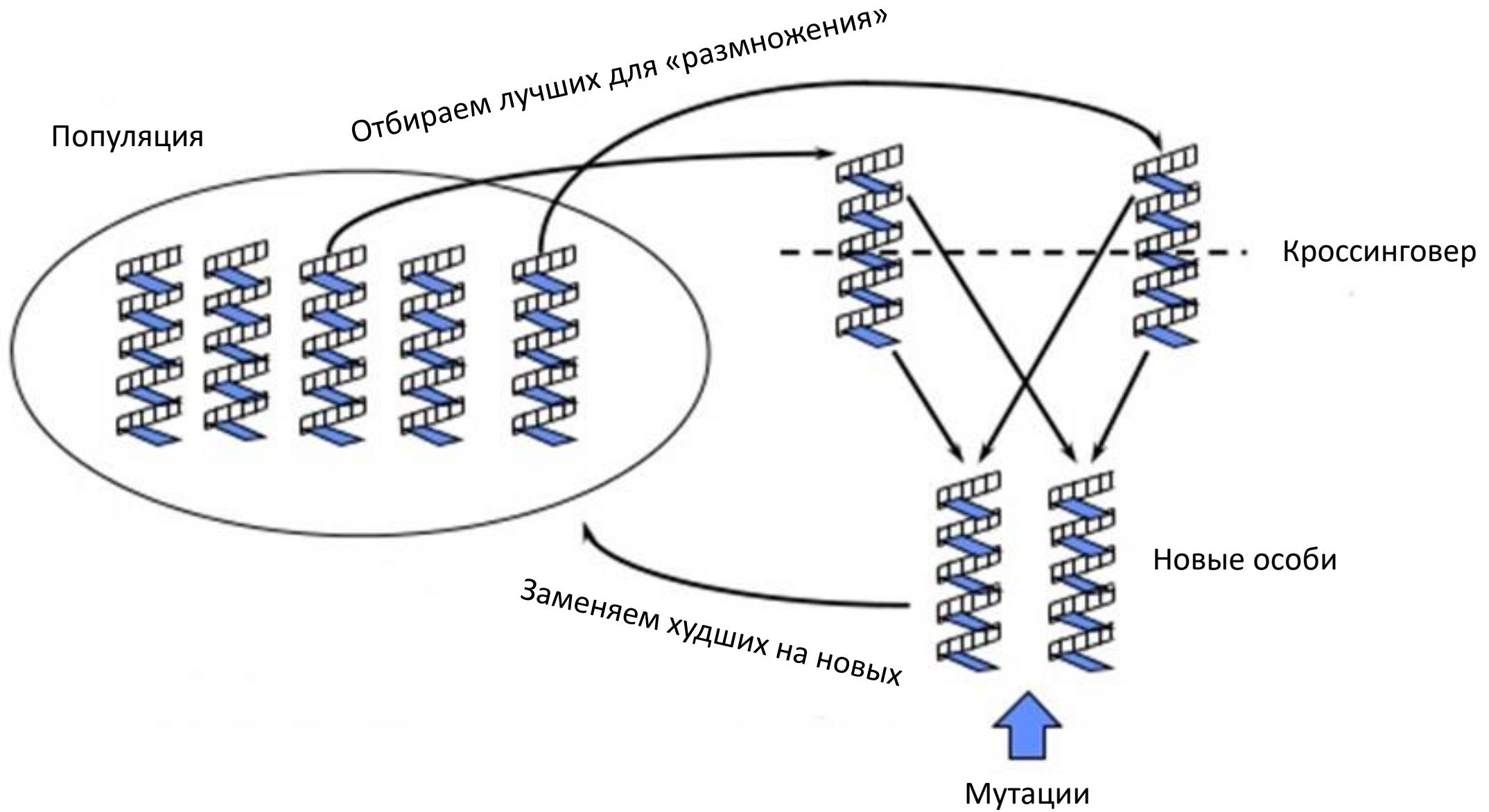


Генетический алгоритм



“Evolved antenna”:
полученная с помощью симуляции —
генетического алгоритма —
антенна, использующаяся на спутниках,
измеряющих магнитосферу земли,
запущенных в 2006 году – миссия Space Technology 5.

Генетический алгоритм



No Free Lunch Theorem

$$\sum_f P(d_m^y | f, m, a_1) = \sum_f P(d_m^y | f, m, a_2)$$

$d_m^y \{y_1, y_2 \dots y_m\}$ - последовательность полученных значений оптимизируемой функции.

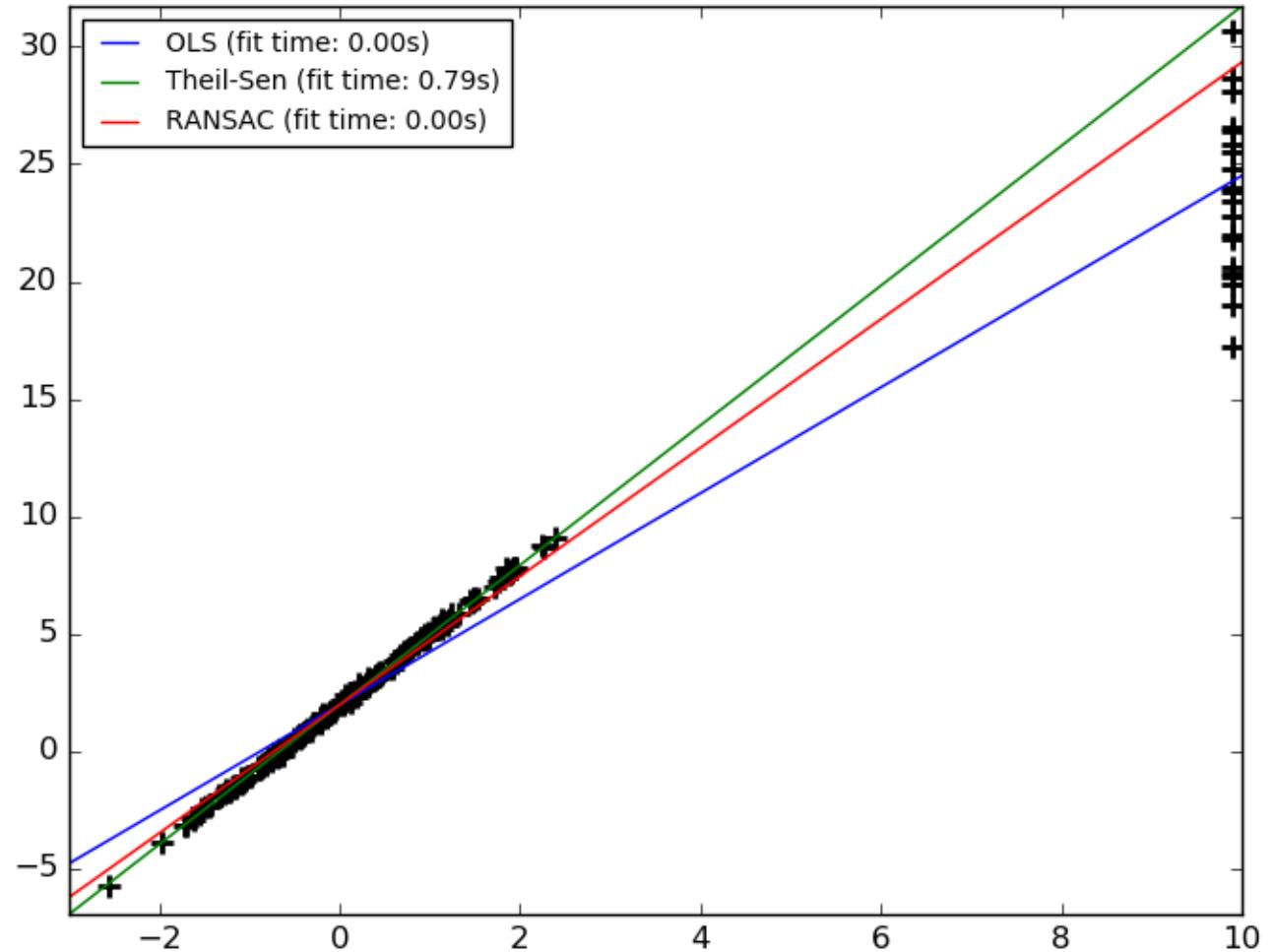
$$y = f(x)$$

a_1, a_2 – алгоритмы оптимизации.

Ансамблевые методы

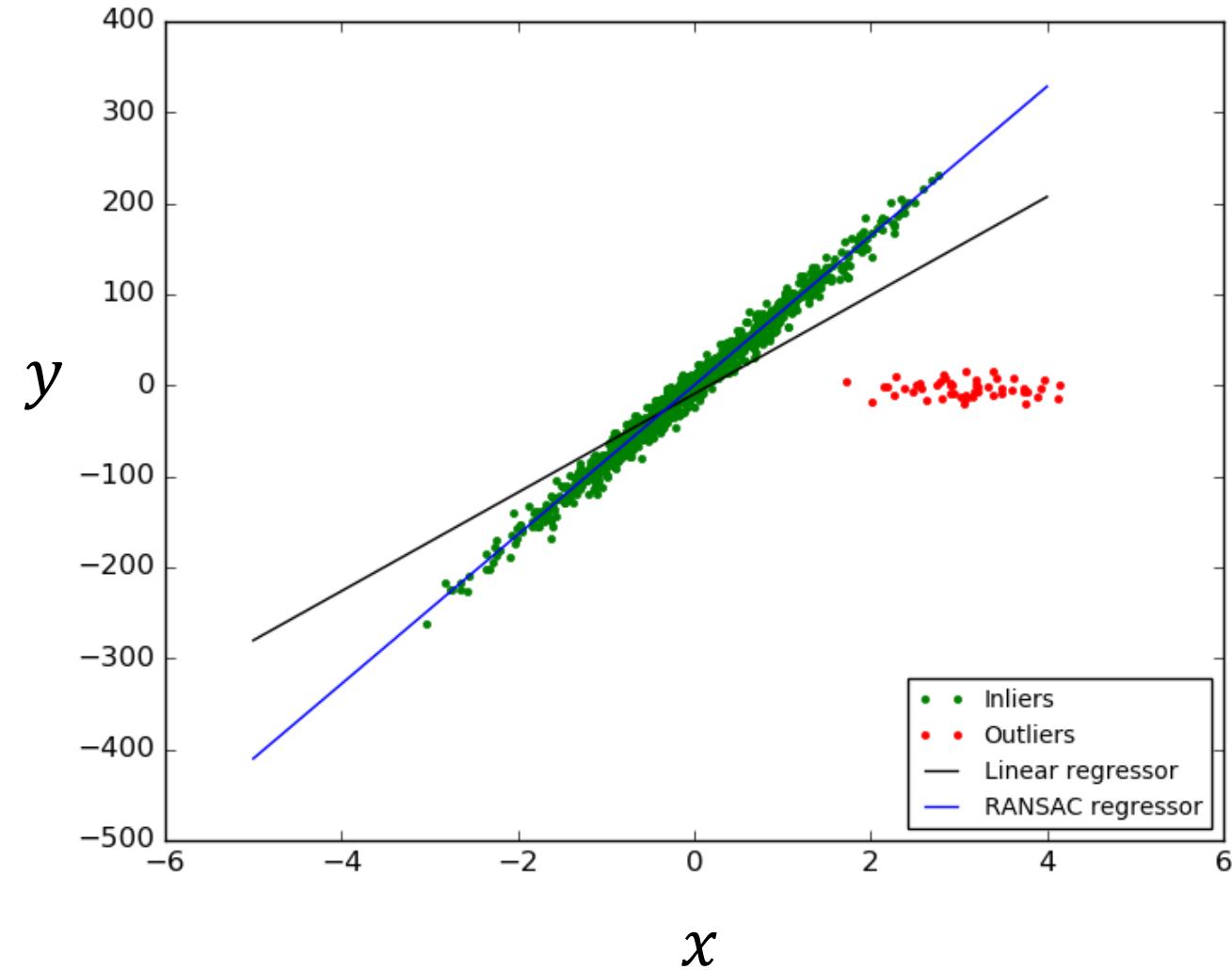
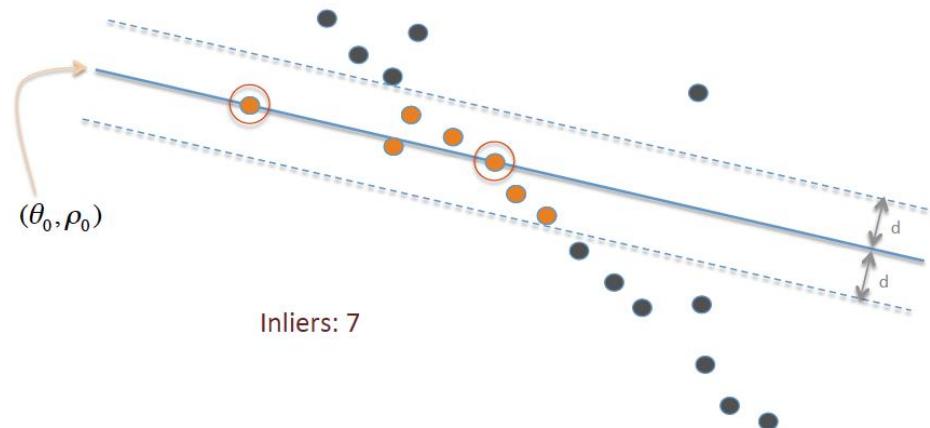
Theil-Sen Regressor

Считаем медианный вектор w по подмножествам X



RANSAC: RANdom SAmple Consensus

1. Строим модель по случайной подвыборке.
2. Принимаем модель в набор, если достаточное количество точек лежит внутри определенной полосы (inliers).
3. Повторяем 1-2 заданное количество раз, после чего выбираем лучшую модель из набора и дообучаем на всех inliers.



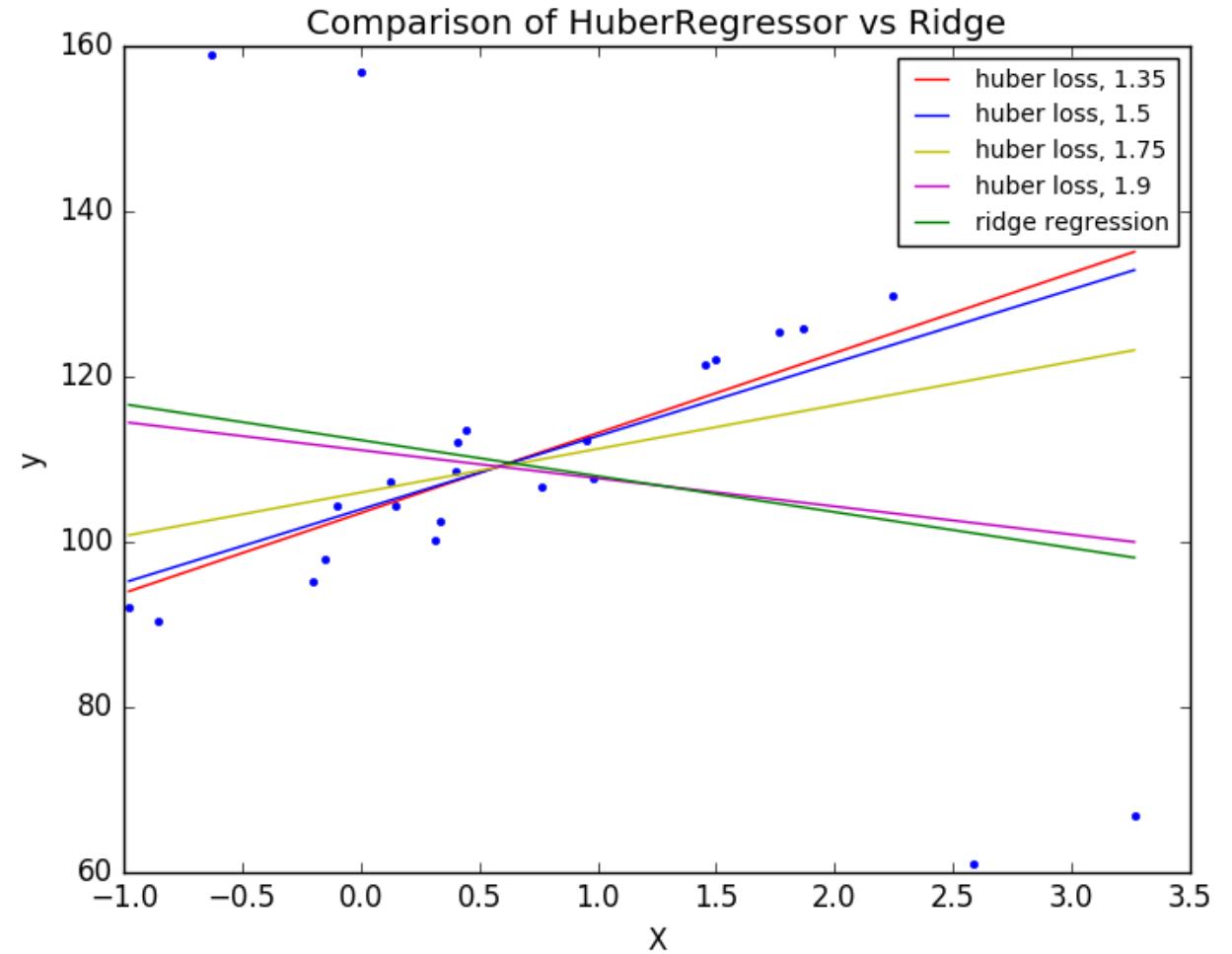
Huber Regressor

Квадратная ошибка для близких точек (inliers), линейная для дальних (outliers).

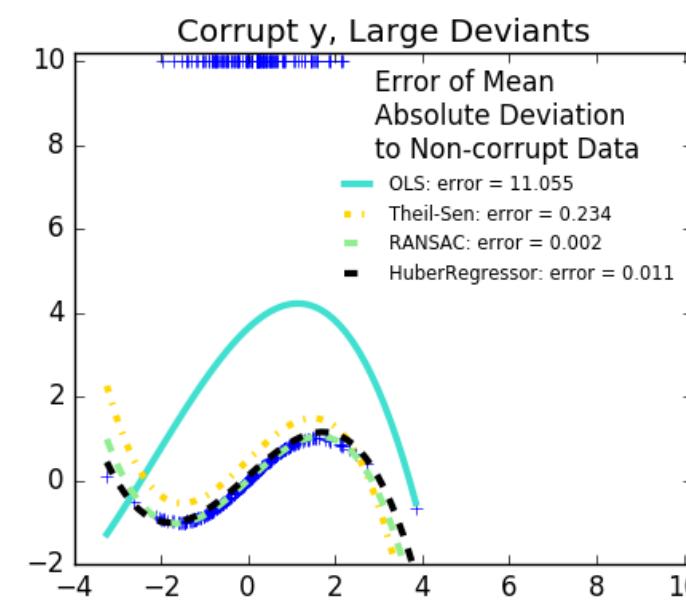
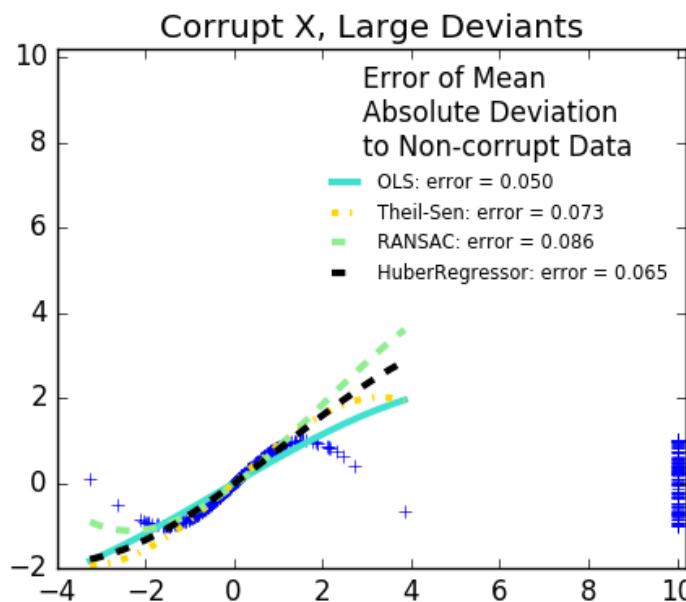
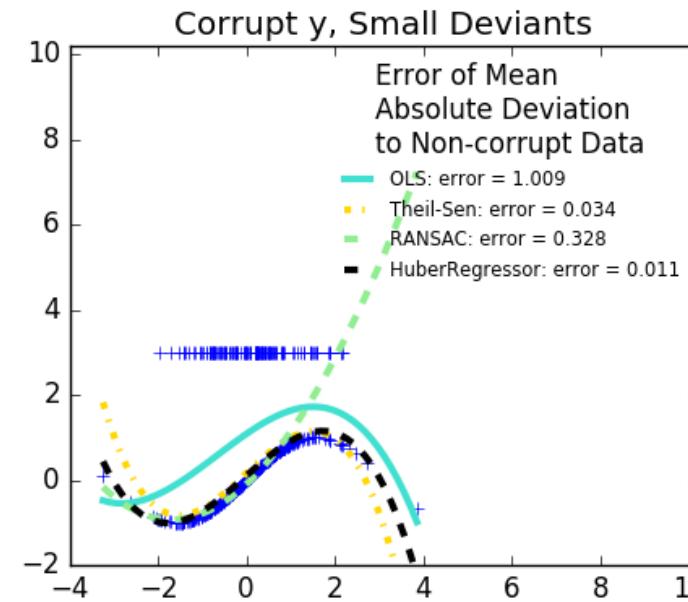
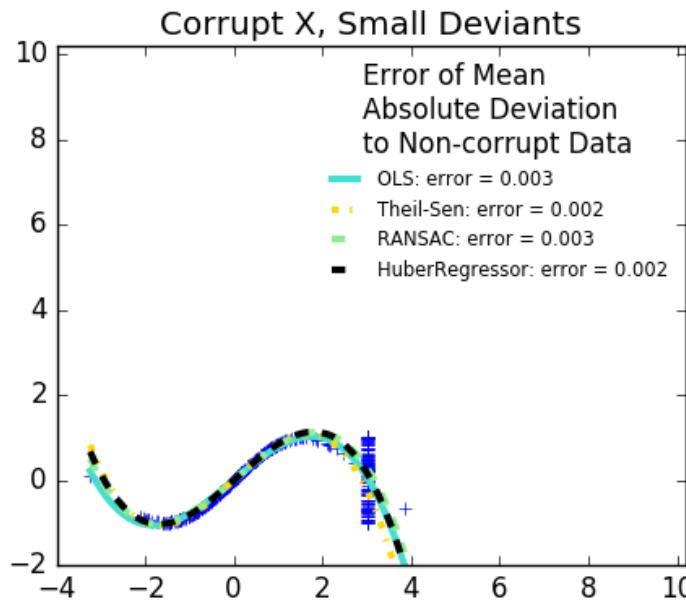
$$\min_{w,\sigma} \sum_{i=1}^N H_m \left(\frac{x_i w - y_i}{\sigma} \right) + \alpha \|w\|_2^2$$

$$H_m(z) = \begin{cases} z^2, & \text{если } |z| < \epsilon \\ 2\epsilon|z| - \epsilon^2, & \text{если } |z| \geq \epsilon \end{cases}$$

σ - константа масштабирования.



RANSAC vs. Theil-Sen vs. Huber



Voting

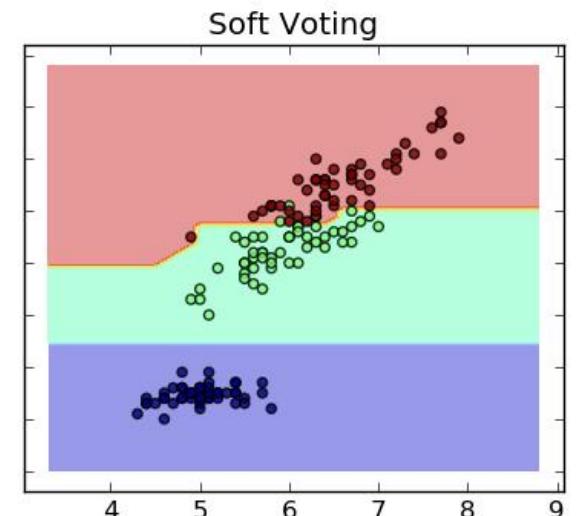
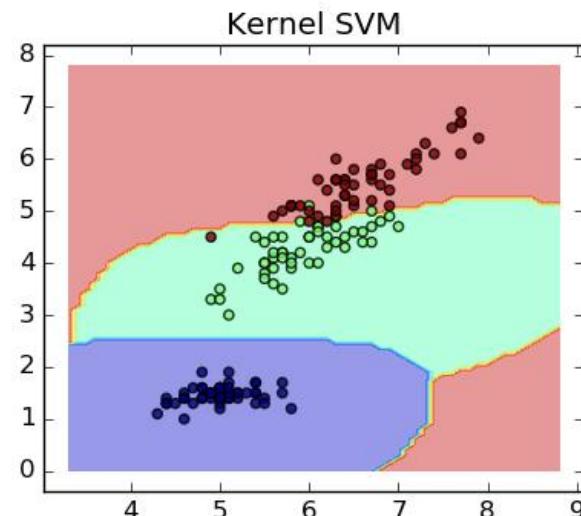
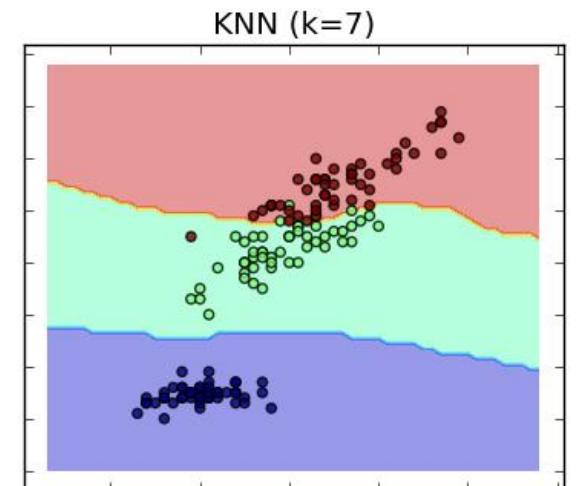
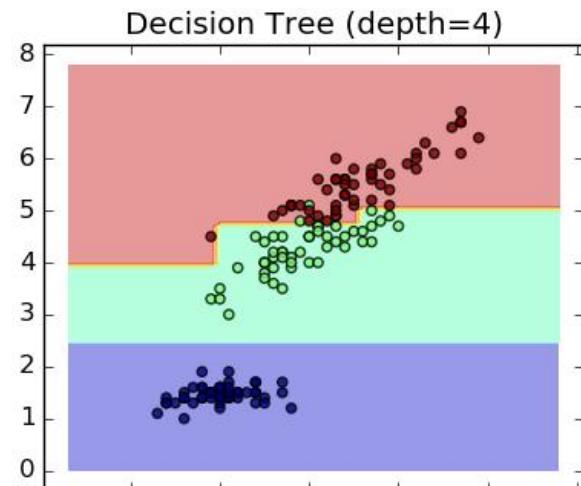
Hard voting:

Побеждает класс, за который голосует большее число голосов.

(При ничьей в sklearn победа отдается первому по алфавиту классу).

Soft voting:

Считается средняя вероятность.



Bagging

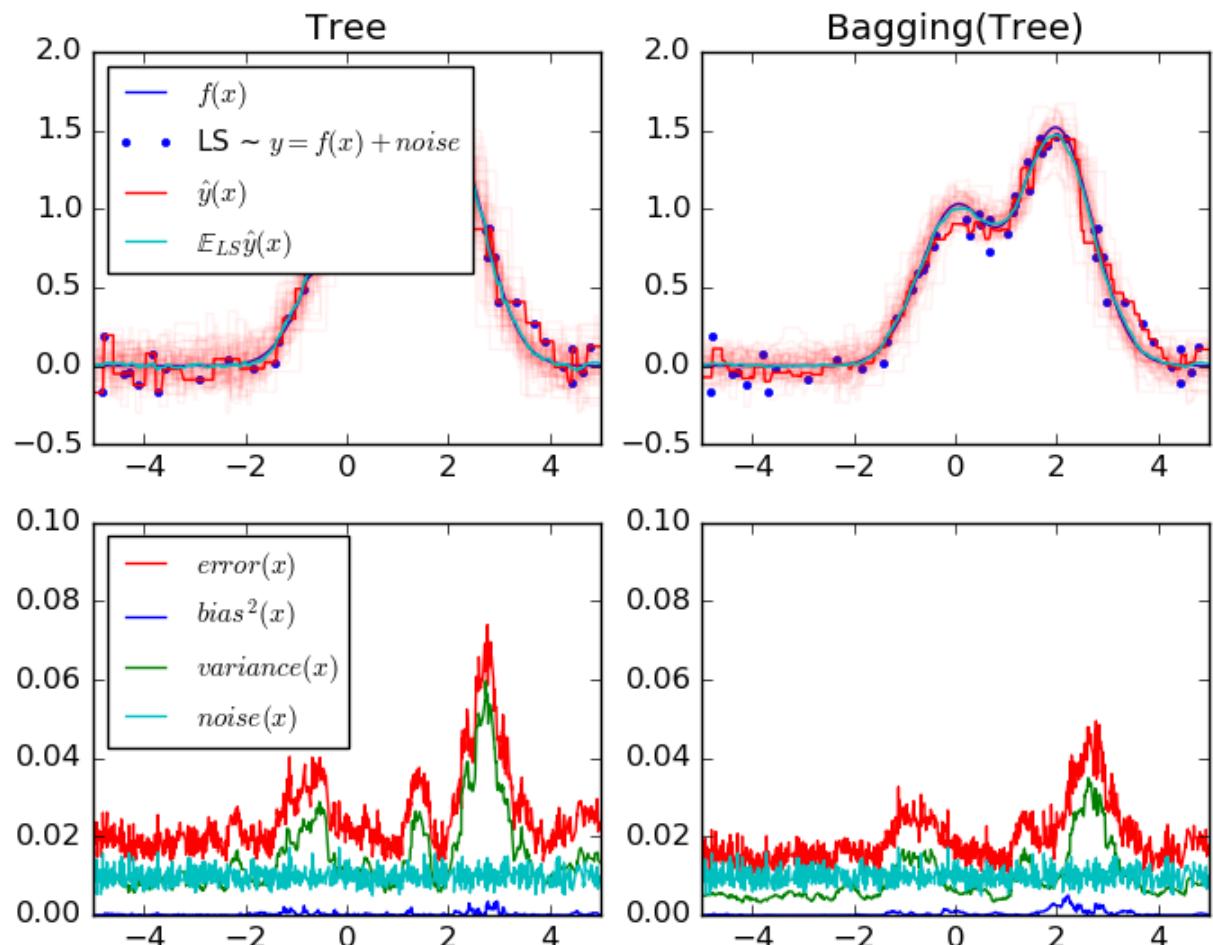
Типы набора выборок из изначальной:

Pasting – просто случайная подвыборка, без повторений.

Bagging – случайная подвыборка (может быть того же размера, что и вся выборка) с повторениями.

Random Subspaces – случайный набор признаков.

Random Patches – случайная подвыборка со случайным набором признаков.



Random Forests (Случайные леса)

Обучим много деревьев на «случайных данных»

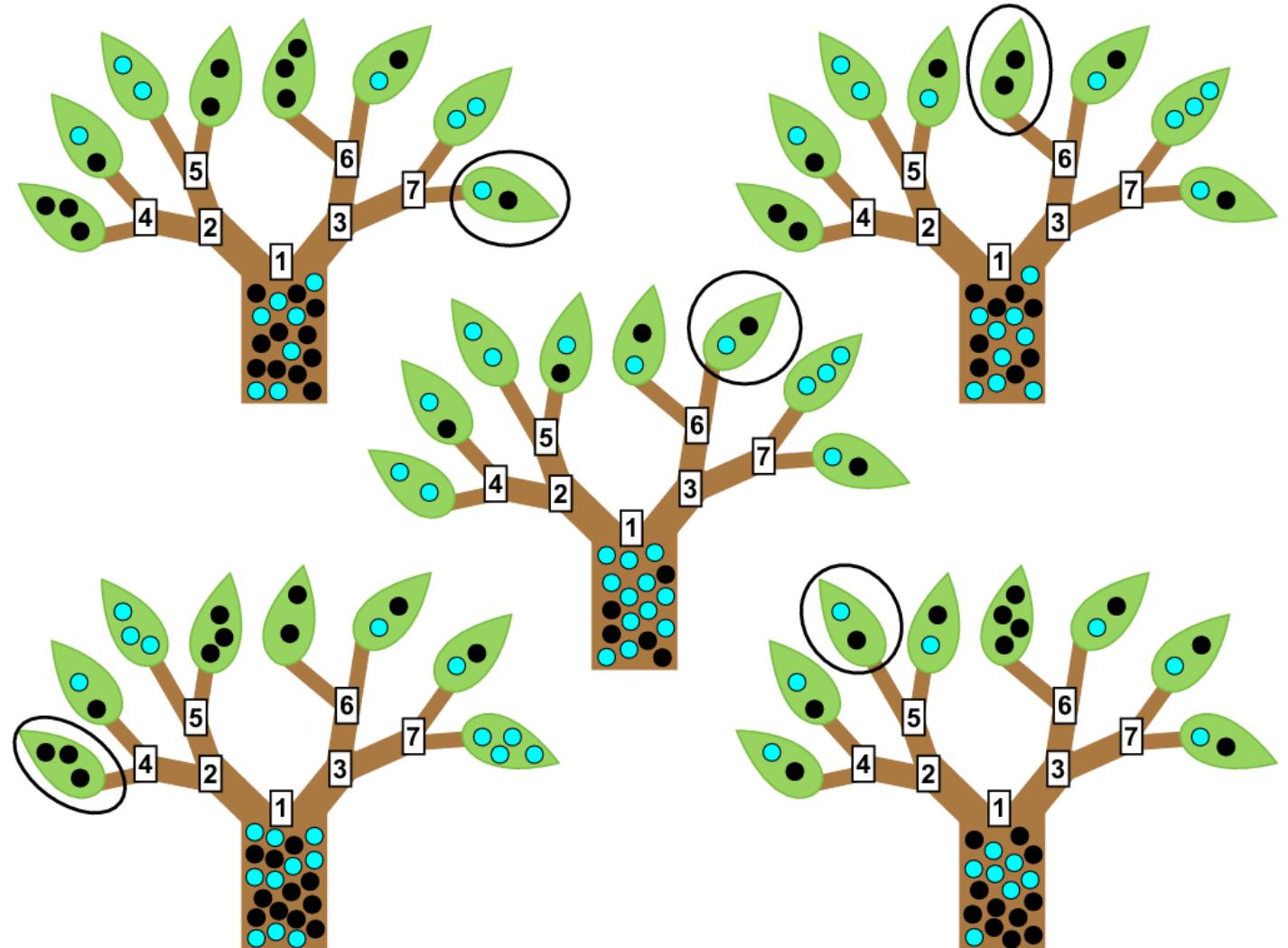
Случайные данные:

1) Датасеты с помощью **bagging (bootstrap aggregating)** – вытаскиваем из исходного датасета столько же данных, но с повторениями.

2) Берем случайное подмножество признаков.

Результат – голосование, вероятность, логарифм вероятностей.

Параметры – количество используемых примеров и признаков для построения каждого дерева.



Adaptive Boosting (AdaBoost)

Начнем с равномерного распределения: $D_1(i) = \frac{1}{N}$, для $(x_1, y_1), \dots, (x_N, y_N)$.

На каждом шаге:

Обучим слабую гипотезу $h_t: X \rightarrow \{-1, +1\}$, такую что взвешенная ошибка минимальна:

$$E_t = \sum_{i=1}^N D_t(i) [h_t(x_i) \neq y_i], \quad \text{вес гипотезы: } \alpha_t = \frac{1}{2} \ln \left(\frac{1 - E_t}{E_t} \right)$$

Меняем веса данных в зависимости текущих ошибок на них:

$$D_{t+1}(i) = D_t(i) e^{-\alpha_t y_i h_t(x_i)}$$

Итоговая гипотеза – сумма (с коэффициентами) слабых гипотез:

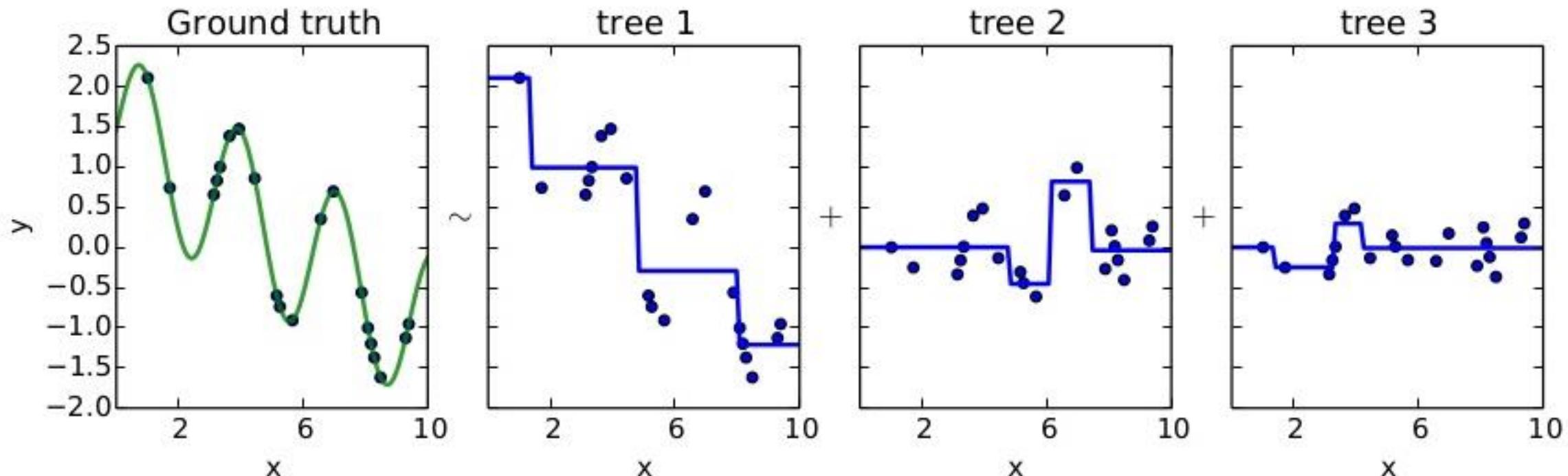
$$H(x) = \operatorname{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

Gradient Boosting

В общем случае:

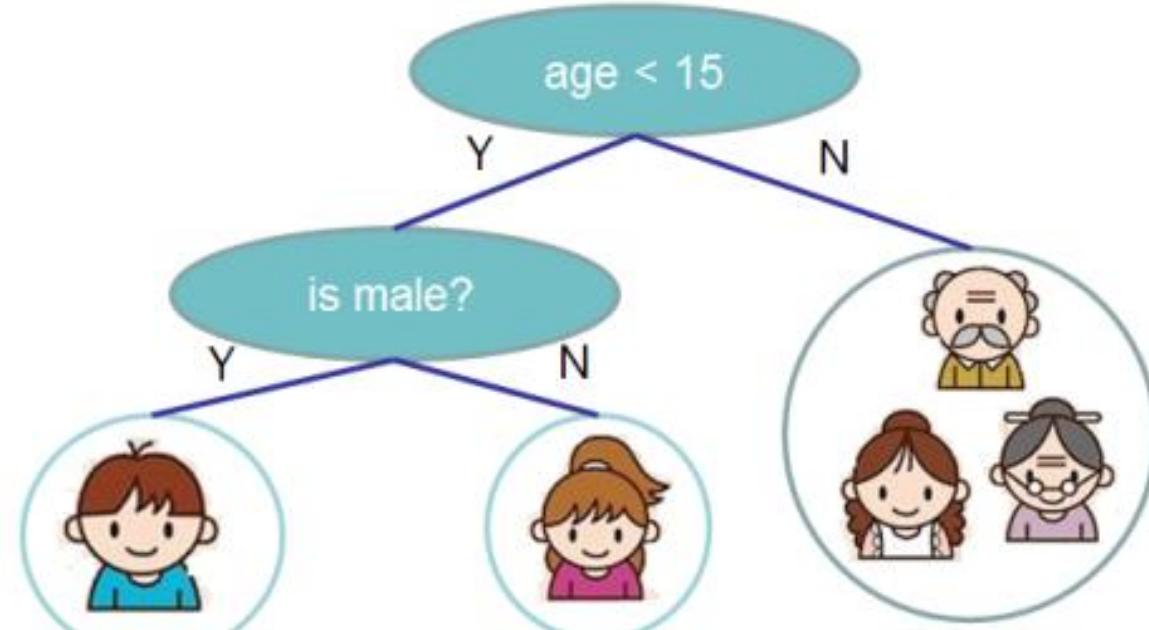
$$H_{t+1}(x) = H_t(x) + h_{t+1}(x) \rightarrow y \Rightarrow h_{t+1}(x) \rightarrow y - H_t(x)$$

Пример для деревьев:



Input: age, gender, occupation, ...

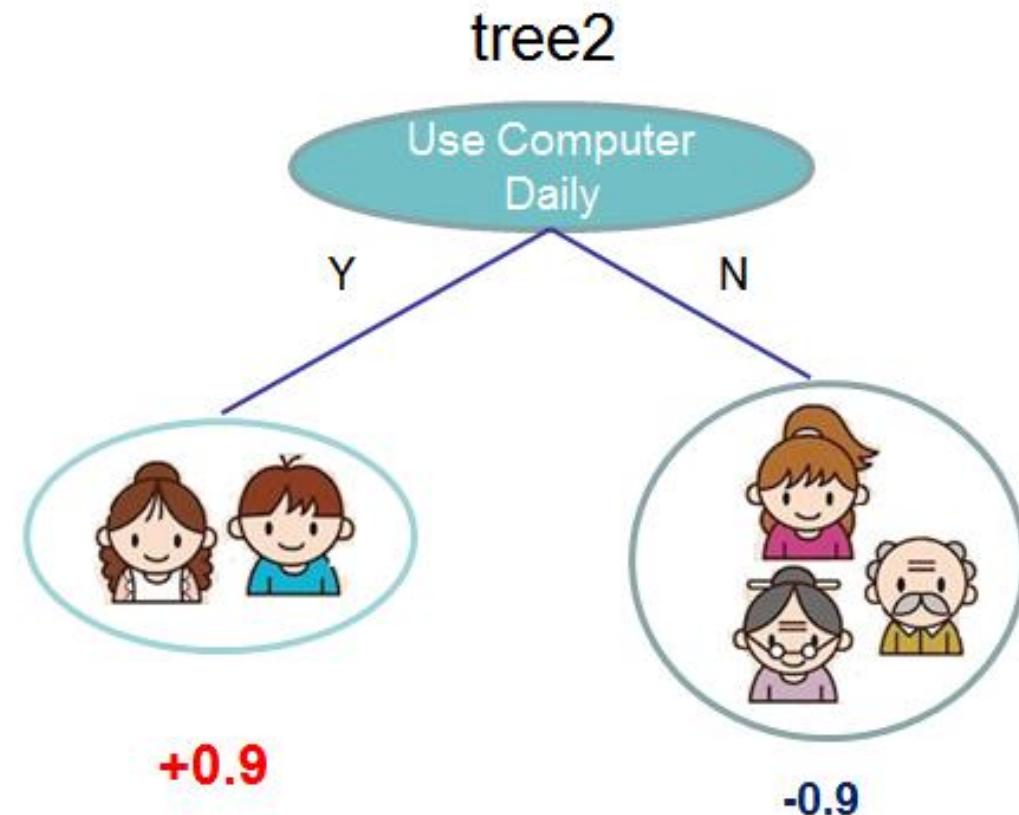
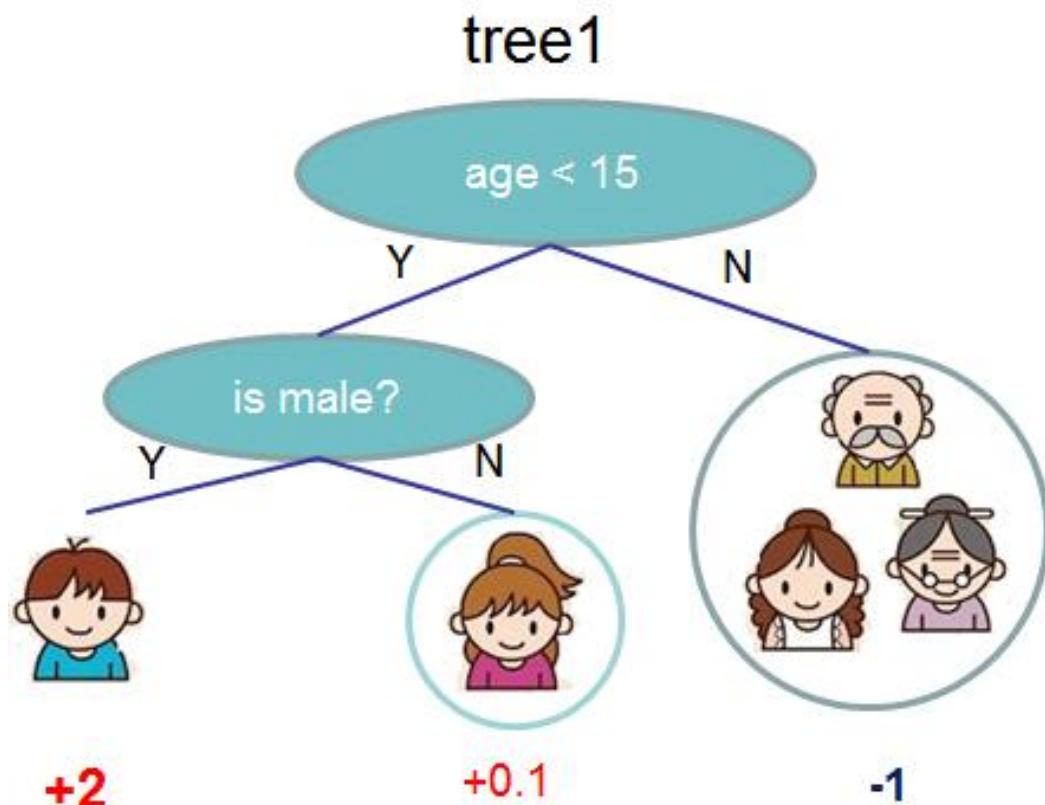
Does the person like computer games



prediction score in each leaf → +2

+0.1

-1



$$f(\text{boy icon}) = 2 + 0.9 = 2.9$$

$$f(\text{old person icon}) = -1 - 0.9 = -1.9$$

eXtreme Gradient Boosting (XGBoost)

$$H_t(\mathbf{x}) = H_{t-1}(\mathbf{x}) + h_t(\mathbf{x}) = \sum_{j=1}^t h_j(\mathbf{x})$$

Минимизируем:

$$E_t = \sum_{i=1}^N L(H_t(\mathbf{x}_i), y_i) + \sum_{j=1}^t \Omega(h_j) = \sum_{i=1}^N L\left((H_{t-1}(\mathbf{x}_i) + h(\mathbf{x}_i)), y_i\right) + \sum_{j=1}^{t-1} \Omega(h_j) + \Omega(h_t)$$


Регуляризация

XGBoost

$$E_t = \sum_{i=1}^N L\left(\left(H_{t-1}(\mathbf{x}_i) + h(\mathbf{x}_i)\right), y_i\right) + \sum_{j=1}^{t-1} \Omega(h_j) + \Omega(h_t) = \sum_{i=1}^N \left(2(H_{t-1}(\mathbf{x}_i) - y_i)h_t(\mathbf{x}_i) + (h_t(\mathbf{x}_i))^2\right) + \Omega(h_t) + const$$

В общем случае:

$$E_t = \sum_{i=1}^N \left(L(H_{t-1}(\mathbf{x}_i), y_i) + u_i h_t(\mathbf{x}_i) + \frac{1}{2} v_i (h_t(\mathbf{x}_i))^2\right) + \Omega(h_t) + const,$$

$$u_i = \partial_{H_{t-1}(\mathbf{x}_i)} (L(H_{t-1}(\mathbf{x}_i), y_i))$$

$$v_i = \partial_{H_{t-1}(\mathbf{x}_i)}^2 (L(H_{t-1}(\mathbf{x}_i), y_i))$$

Минимизируем:

$$E_t = \sum_{i=1}^N \left(u_i h_t(\mathbf{x}_i) + \frac{1}{2} v_i (h_t(\mathbf{x}_i))^2\right) + \Omega(h_t)$$

Для MSE:

$$E_t = \sum_{i=1}^N \left((H_{t-1}(\mathbf{x}_i) + h_t(\mathbf{x}_i)) - y_i\right)^2 + \sum_{j=1}^t \Omega(h_j) = \sum_{i=1}^N \left(2(H_{t-1}(\mathbf{x}_i) - y_i)h_t(\mathbf{x}_i) + (h_t(\mathbf{x}_i))^2\right) + \Omega(h_t) + const$$

XGBoost

$$\Omega(f) = \gamma M + \frac{1}{2} \lambda \sum_{j=1}^M w_j^2, M \text{ -- количество листьев, } w \text{ -- коэффициент листа}$$

$$E_t = \sum_{i=1}^N \left(u_i w_{q(x_i)} + \frac{1}{2} v_i (w_{q(x_i)})^2 \right) + \gamma M + \frac{1}{2} \lambda \sum_{j=1}^M w_j^2 \quad \text{Где } q(x_i) \text{ -- лист, к которому принадлежит } x_i.$$

Сгруппируем по листьям:

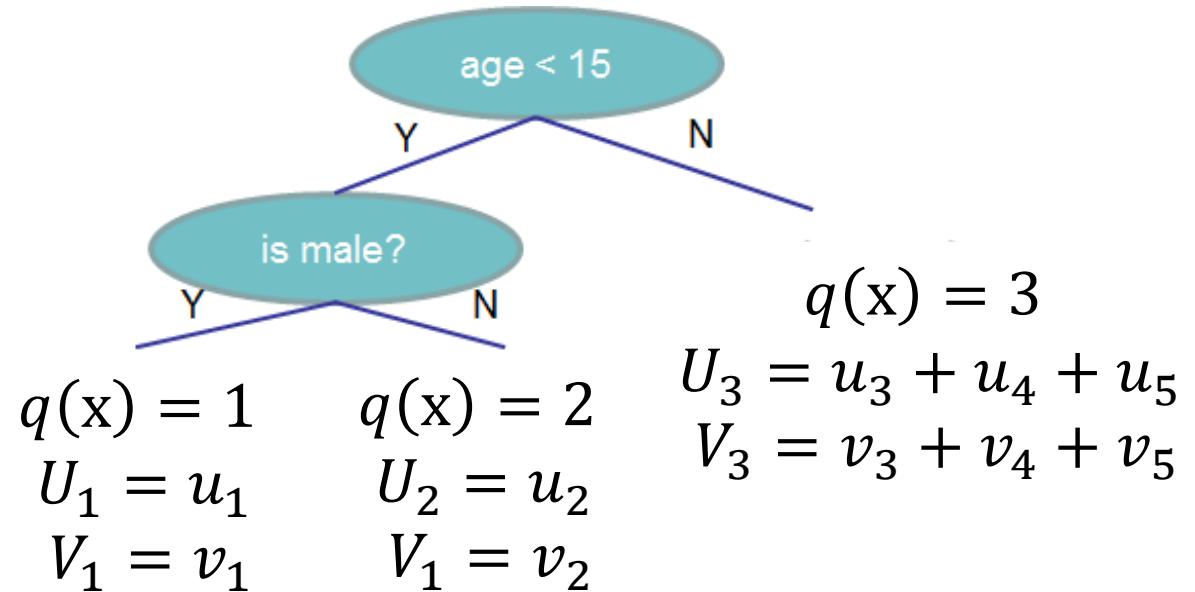
$$E_t = \sum_{j=1}^M \left(\sum_{q(x_i)=j} u_i w_j + \frac{1}{2} \left(\sum_{q(x_i)=j} v_i + \lambda \right) w_j^2 \right) + \gamma M = \sum_{j=1}^M \left(U_j w_j + \frac{1}{2} (V_j + \lambda) w_j^2 \right) + \gamma M$$

$$U_j = \sum_{q(x_i)=j} u_i \quad V_j = \sum_{q(x_i)=j} v_i$$

XGBoost

Instance index gradient statistics

1		u_1, v_1
2		u_2, v_2
3		u_3, v_3
4		u_4, v_4
5		u_5, v_5



$$U_j = \sum_{q(\mathbf{x}_i)=j} u_i \quad V_j = \sum_{q(\mathbf{x}_i)=j} v_i$$

XGBoost

$$E_t = \sum_{j=1}^M \left(U_j w_j + \frac{1}{2} (V_j + \lambda) w_j^2 \right) + \gamma M$$

$$w_j^{opt} = -\frac{U_j}{V_j + \lambda} \quad E_t^{opt} = -\frac{1}{2} \sum_{j=1}^M \frac{U_j^2}{V_j + \lambda} + \gamma M$$

$$Gain = \frac{1}{2} \left[\frac{U_L^2}{V_L + \lambda} + \frac{U_R^2}{V_R + \lambda} - \frac{(U_L + U_R)^2}{V_L + V_R + \lambda} \right] - \gamma$$

