

# Критерии выбора моделей Теория обобщающей способности Методы отбора признаков

К. В. Воронцов  
vokov@forecsys.ru

Этот курс доступен на странице вики-ресурса  
<http://www.MachineLearning.ru/wiki>  
«Машинное обучение (курс лекций, К.В.Воронцов)»

23 октября 2013

## Содержание

- 1 Задачи и критерии выбора метода обучения**
  - Задачи выбора модели или метода обучения
  - Эмпирические оценки скользящего контроля
  - Аналитические оценки и критерии регуляризации
- 2 Теория обобщающей способности**
  - Вероятность переобучения и VC-теория
  - Бритва Оккама
  - Комбинаторная теория переобучения
- 3 Методы отбора признаков**
  - Полный перебор и жадные алгоритмы
  - Поиск в глубину и в ширину
  - Стохастический поиск

## Задачи выбора метода обучения

**Дано:**  $X$  — пространство объектов;  $Y$  — множество ответов;  
 $X^\ell = (x_i, y_i)_{i=1}^\ell$  — обучающая выборка,  $y_i = y^*(x_i)$ ;  
 $A_t = \{a: X \rightarrow Y\}$  — модели алгоритмов,  $t \in T$ ;  
 $\mu_t: (X \times Y)^\ell \rightarrow A_t$  — методы обучения,  $t \in T$ .

**Найти:** метод  $\mu_t$  с наилучшей *обобщающей способностью*.

**Частные случаи:**

- выбор лучшей модели  $A_t$  (model selection);
- выбор метода обучения  $\mu_t$  для заданной модели  $A$  (в частности, оптимизация *гиперпараметров*);
- отбор признаков (features selection):  
 $\mathcal{F} = \{f_j: X \rightarrow D_j: j = 1, \dots, n\}$  — множество признаков;  
метод обучения  $\mu_{\mathcal{G}}$  использует только признаки  $\mathcal{G} \subseteq \mathcal{F}$ .

## Как оценить качество обучения по прецедентам?

$\mathcal{L}(a, x)$  — функция потерь алгоритма  $a$  на объекте  $x$ ;

$Q(a, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} \mathcal{L}(a, x_i)$  — функционал качества  $a$  на  $X^\ell$ .

*Внутренний критерий* оценивает качество на обучении  $X^\ell$ :

$$Q_\mu(X^\ell) = Q(\mu(X^\ell), X^\ell).$$

**Недостаток:** эта оценка смещена, т.к.  $\mu$  минимизирует её же.

*Внешний критерий* оценивает качество «вне обучения», например, по отложенной (hold-out) контрольной выборке  $X^k$ :

$$Q_\mu(X^\ell, X^k) = Q(\mu(X^\ell), X^k).$$

**Недостаток:** эта оценка зависит от разбиения  $X^L = X^\ell \sqcup X^k$ .

## Скользящий контроль (cross-validation, CV)

Оценка *полного скользящего контроля* (complete CV) — усреднение hold-out по всем  $C_L^\ell$  разбиениям  $X^L = X^\ell \sqcup X^k$ :

$$\text{CCV}(\mu, X^L) = \frac{1}{C_L^\ell} \sum_{X^\ell \subset X^L} Q_\mu(X^\ell, X^k) = \mathbf{E} Q_\mu(X^\ell, X^k),$$

если полагать, что **все разбиения равновероятны**.

**Недостаток 1:** эта оценка вычислительно слишком сложна.

Устраняется либо *аналитическим оцениванием*, либо *эмпирическим оцениванием* по подмножеству разбиений.

**Недостаток 2:** эта оценка не учитывает дисперсию hold-out.

Устраняется *оцениванием распределения* значений hold-out:

$$R_\varepsilon(\mu, X^L) = \mathbf{P}[Q_\mu(X^\ell, X^k) \geq \varepsilon] \quad \text{или}$$
$$Q_\varepsilon(\mu, X^L) = \mathbf{P}[Q_\mu(X^\ell, X^k) - Q_\mu(X^\ell) \geq \varepsilon].$$

## Эмпирические оценки скользящего контроля

Контроль по отдельным объектам (leave one out CV):  $k = 1$ ,

$$\text{LOO}(\mu, X^L) = \frac{1}{L} \sum_{i=1}^L Q_{\mu}(X^L \setminus \{x_i\}, \{x_i\}).$$

Недостатки LOO: ресурсоёмкость, высокая дисперсия.

Контроль по  $q$  блокам ( $q$ -fold CV): случайное разбиение  $X^L = X_1^{\ell_1} \sqcup \dots \sqcup X_q^{\ell_q}$  на  $q$  блоков (почти) равной длины,

$$\widehat{\text{CV}}_q(\mu, X^L) = \frac{1}{q} \sum_{n=1}^q Q_{\mu}(X^L \setminus X_n^{\ell_n}, X_n^{\ell_n}).$$

Недостатки  $q$ -fold CV:

- оценка существенно зависит от разбиения на блоки;
- каждый объект лишь один раз участвует в контроле.

## Эмпирические оценки скользящего контроля

**Контроль  $t$  раз по  $q$  блокам ( $t \times q$ -fold CV)** — стандарт «де факто» для тестирования методов обучения.

Выборка  $X^L$  разбивается  $t$  раз случайным образом на  $q$  блоков

$$X^L = X_{s1}^{\ell_1} \sqcup \dots \sqcup X_{sq}^{\ell_q}, \quad s = 1, \dots, t, \quad \ell_1 + \dots + \ell_q = L;$$

$$\widehat{CV}_{t \times q}(\mu, X^L) = \frac{1}{t} \sum_{s=1}^t \frac{1}{q} \sum_{n=1}^q Q_{\mu}(X^L \setminus X_{sn}^{\ell_n}, X_{sn}^{\ell_n}).$$

**Преимущества  $t \times q$ -fold CV:**

- увеличением  $t$  можно улучшать точность оценки (компромисс между точностью и временем вычислений);
- каждый объект участвует в контроле ровно  $t$  раз;
- можно вычислять доверительные интервалы (при  $t \gtrsim 20$ ).

## Критерии непротиворечивости моделей

**Идея:** Если модель верна, то алгоритмы, настроенные по разным частям данных, не должны противоречить друг другу.

1. По одному случайному разбиению  $X^\ell \sqcup X^k = X^L$ ,  $\ell = k$ :

$$D_1(\mu, X^L) = \frac{1}{L} \sum_{i=1}^L |\mu(X^\ell)(x_i) - \mu(X^k)(x_i)|.$$

2. Аналог  $\widehat{CV}_{t \times 2}$ : по  $t$  разбиениям  $X^L = X_s^\ell \sqcup X_s^k$ ,  $s = 1, \dots, t$ :

$$D_t(\mu, X^L) = \frac{1}{t} \sum_{s=1}^t \frac{1}{L} \sum_{i=1}^L |\mu(X_s^\ell)(x_i) - \mu(X_s^k)(x_i)|.$$

**Недостатки:**

- длина обучения сокращается в 2 раза;
- трудоёмкость возрастает в 2 раза.



## Аналитические оценки и их обращение

Основная идея аналитического подхода:

1. Получить верхнюю оценку  $Q_\varepsilon$ , справедливую для любой выборки  $X^L$  и широкого класса методов обучения  $\mu \in M$ :

$$Q_\varepsilon(\mu, X^L) = P\left[Q_\mu(X^\ell, X^k) - Q_\mu(X^\ell) \geq \varepsilon\right] \leq \eta(\varepsilon).$$

2. Тогда для любой  $X^L$ , любого  $\mu \in M$  и любого  $\eta \in (0, 1)$  с вероятностью не менее  $(1 - \eta)$  справедлива оценка

$$Q_\mu(X^\ell, X^k) \leq Q_\mu(X^\ell) + \varepsilon(\eta),$$

где  $\varepsilon(\eta)$  — функция, обратная к  $\eta(\varepsilon)$ , зависящая от  $A$  и  $\mu$ , но не зависящая от скрытой контрольной выборки  $X^k$ .

3. Оптимизировать метод обучения:  $Q_\mu(X^\ell) + \varepsilon(\eta) \rightarrow \min_{\mu \in M}$

## Критерии регуляризации

*Регуляризатор* — аддитивная добавка к внутреннему критерию, обычно штраф за сложность (complexity penalty) модели  $A$ :

$$Q_{\text{рег}}(\mu, X^\ell) = Q_\mu(X^\ell) + \text{штраф}(A),$$

**Линейные модели:**  $A = \{a(x) = \text{sign}\langle w, x \rangle\}$  — классификация,  
 $A = \{a(x) = \langle w, x \rangle\}$  — регрессия.

$L_2$ -регуляризация (ридж-регрессия, weight decay):

$$\text{штраф}(w) = \tau \|w\|_2^2 = \tau \sum_{j=1}^n w_j^2.$$

$L_1$ -регуляризация (LASSO):

$$\text{штраф}(w) = \tau \|w\|_1 = \tau \sum_{j=1}^n |w_j|.$$

$L_0$ -регуляризация (AIC, BIC):

$$\text{штраф}(w) = \tau \|w\|_0 = \tau \sum_{j=1}^n [w_j \neq 0] = \tau |\mathcal{G}|.$$

## Разновидности $L_0$ -регуляризации

Информационный критерий Акаике (Akaike Information Criterion):

$$\text{AIC}(\mu, x) = Q_\mu(X^\ell) + \frac{2\hat{\sigma}^2}{\ell} |\mathcal{G}|,$$

где  $\hat{\sigma}^2$  — оценка дисперсии ошибки  $D(y_i - a(x_i))$ .

Байесовский информационный критерий (Bayes Inform. Criterion):

$$\text{BIC}(\mu, X^\ell) = \frac{\ell}{\hat{\sigma}^2} \left( Q_\mu(X^\ell) + \frac{\hat{\sigma}^2 \ln \ell}{\ell} |\mathcal{G}| \right).$$

Оценка Вапника-Червоненкиса (VC-bound):

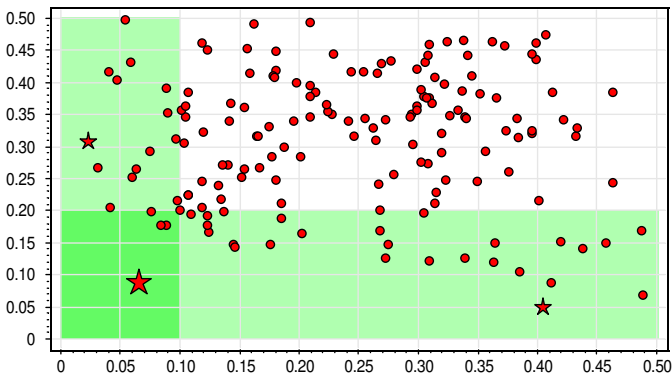
$$\text{VC}(\mu, X^\ell) = Q_\mu(X^\ell) + \sqrt{\frac{h}{\ell} \ln \frac{2e\ell}{h} + \frac{1}{\ell} \ln \frac{9}{4\eta}},$$

$h$  — VC-размерность; для линейных, опять-таки,  $h = |\mathcal{G}|$ ;

$\eta$  — уровень значимости; обычно  $\eta = 0.05$ .

## Выбор модели по совокупности внешних критериев

Модель, немного неоптимальная по обоим критериям, скорее всего, лучше, чем модель, оптимальная по одному критерию, но не оптимальная по другому.



## Резюме

- Обобщающая способность может быть формализована с помощью функционалов, построенных по принципу полного скользящего контроля (CCV).
- Эмпирические оценки CCV просто реализуются, но требуют многократного применения  $\mu$ , следовательно, длительных вычислений.
- Аналитические оценки CCV не требуют многократного применения  $\mu$ , но могут быть сильно завышенными или иметь узкие границы применимости.

## Бинарная функция потерь. Матрица ошибок

$X^L = \{x_1, \dots, x_L\}$  — конечное генеральное множество объектов;

$A = \{a_1, \dots, a_D\}$  — конечное семейство алгоритмов;

$\mathcal{L}(a, x) \equiv I(a, x) = [\text{алгоритм } a \text{ ошибается на объекте } x];$

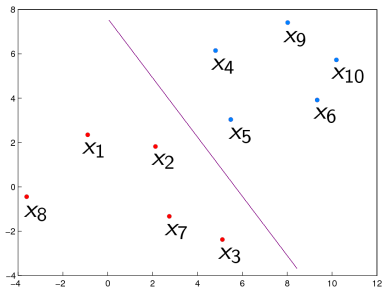
$L \times D$ -матрица ошибок с попарно различными столбцами:

	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$	$\dots$	$a_D$	
$x_1$	1	1	0	0	0	1	$\dots$	1	$X^\ell$ — наблюдаемая (обучающая) выборка длины $\ell$
$\dots$	0	0	0	0	1	1	$\dots$	1	
$x_\ell$	0	0	1	0	0	0	$\dots$	0	
$x_{\ell+1}$	0	0	0	1	1	1	$\dots$	0	$X^k$ — скрытая (контрольная) выборка длины $k = L - \ell$
$\dots$	0	0	0	1	0	0	$\dots$	1	
$x_L$	0	1	1	1	1	1	$\dots$	0	

$n(a, X) = \sum_{x \in X} I(a, x)$  — число ошибок  $a \in A$  на выборке  $X \subset X^L$ ;

$\nu(a, X) = n(a, X)/|X|$  — частота ошибок  $a$  на выборке  $X$ ;

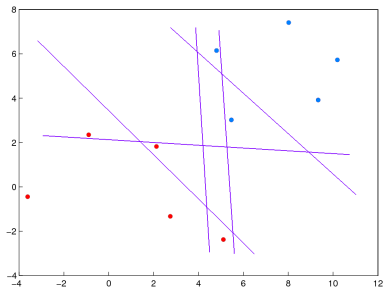
## Пример. Матрица ошибок линейных классификаторов



1 вектор с 0 ошибками

x <sub>1</sub>	0
x <sub>2</sub>	0
x <sub>3</sub>	0
x <sub>4</sub>	0
x <sub>5</sub>	0
x <sub>6</sub>	0
x <sub>7</sub>	0
x <sub>8</sub>	0
x <sub>9</sub>	0
x <sub>10</sub>	0

## Пример. Матрица ошибок линейных классификаторов

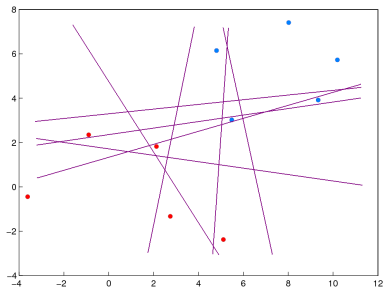


1 вектор с 0 ошибками  
5 векторов с 1 ошибкой

$x_1$	0	1	0	0	0	0
$x_2$	0	0	1	0	0	0
$x_3$	0	0	0	1	0	0
$x_4$	0	0	0	0	1	0
$x_5$	0	0	0	0	0	1
$x_6$	0	0	0	0	0	0
$x_7$	0	0	0	0	0	0
$x_8$	0	0	0	0	0	0
$x_9$	0	0	0	0	0	0
$x_{10}$	0	0	0	0	0	0



## Пример. Матрица ошибок линейных классификаторов



1 вектор с 0 ошибками  
 5 векторов с 1 ошибкой  
 8 векторов с 2 ошибками  
 и т. д...

$x_1$	0	1	0	0	0	0	1	0	0	0	0	1	1	0	...
$x_2$	0	0	1	0	0	0	1	1	0	0	0	0	0	0	...
$x_3$	0	0	0	1	0	0	0	1	1	0	0	0	0	1	...
$x_4$	0	0	0	0	1	0	0	0	1	1	0	0	0	0	...
$x_5$	0	0	0	0	0	1	0	0	0	1	1	1	0	0	...
$x_6$	0	0	0	0	0	0	0	0	0	0	1	0	1	0	...
$x_7$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	...
$x_8$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
$x_9$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
$x_{10}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...

## Задача оценивания вероятности переобучения

Основное вероятностное предположение:

все разбиения  $X^L \sqcup X^k = X^L$  равновероятны

(слабый вариант **гипотезы независимости** выборки  $X^L$ ).

*Переобученность* — разность частот ошибок на  $X^k$  и на  $X^L$ :

$$\delta(\mu, X^L) = \nu(\mu(X^L), X^k) - \nu(\mu(X^L), X^L).$$

*Переобучение* — это событие  $\delta(\mu, X^L) \geq \varepsilon$ .

Основная задача — оценить **вероятность** переобучения:

$$Q_\varepsilon(\mu, X^L) = \mathbf{P}[\delta(\mu, X^L) \geq \varepsilon].$$

## Простейший, но важный частный случай

Пусть  $A = \{a\}$  — одноэлементное множество,  $m = n(a, X^L)$ .

Тогда вероятность переобучения есть вероятность большого отклонения частот ошибок в двух подвыборках:

$$Q_\varepsilon(a, X^L) = P[\nu(a, X^k) - \nu(a, X^\ell) \geq \varepsilon].$$

### Теорема

Для любого  $X^L$ , любого  $\varepsilon \in [0, 1]$

$$Q_\varepsilon(a, X^L) = \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L}(m - \varepsilon k) \right),$$

где  $\mathcal{H}_L^{\ell, m}(z) = \sum_{s=0}^{\lfloor z \rfloor} \frac{C_m^s C_{L-m}^{\ell-s}}{C_L^\ell}$  — функция гипергеометрического распределения.

## Доказательство

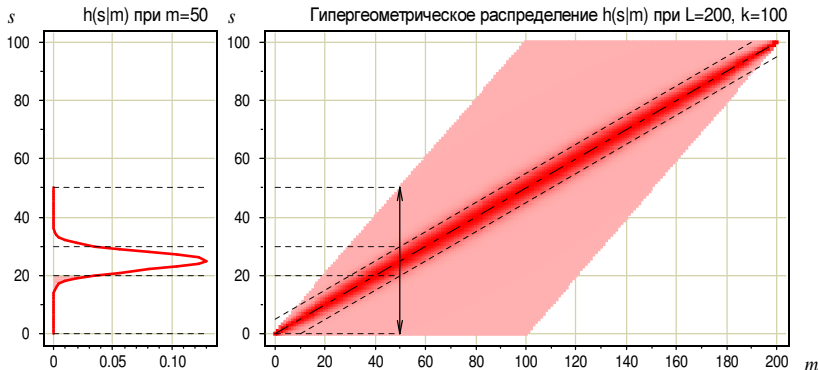
1. Обозначим  $s = n(a, X^\ell)$ .
2. «Школьная» задача по теории вероятностей:  
 в урне  $L$  шаров,  $m$  из них чёрные; извлекаем  $\ell$  шаров наугад.  
 Какова вероятность того, что  $s$  из них чёрные?

$$P[n(a, X^\ell) = s] = C_m^s C_{L-m}^{\ell-s} / C_L^\ell.$$

3. Распишем  $Q_\varepsilon$ , подставив  $\nu(a, X^k) = \frac{m-s}{k}$ ,  $\nu(a, X^\ell) = \frac{s}{\ell}$ :

$$\begin{aligned} Q_\varepsilon(a, X^\ell) &= P[\nu(a, X^k) - \nu(a, X^\ell) \geq \varepsilon] = \\ &= \sum_{s=0}^{\ell} \underbrace{\left[ \frac{m-s}{k} - \frac{s}{\ell} \geq \varepsilon \right]}_{s \leq \frac{\ell}{L}(m-\varepsilon k)} \underbrace{P[n(a, X^\ell) = s]}_{C_m^s C_{L-m}^{\ell-s} / C_L^\ell} = \\ &= \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L}(m - \varepsilon k) \right). \quad \blacksquare \end{aligned}$$

# Гипергеометрическое распределение $h(s|m) = C_m^s C_{L-m}^{\ell-s} / C_L^\ell$



Предсказание числа  $m = n(a, X^L)$  по числу  $s = n(a, X^\ell)$  возможно благодаря узости гипергеометрического пика, причём при  $\ell, k \rightarrow \infty$  он сужается, и  $\nu(a, X^\ell) \rightarrow \nu(a, X^k)$  (явление концентрации вероятности, закон больших чисел).

## Теория Вапника–Червоненкиса

Рассмотрим общий случай —  $A$  произвольное, конечное.

1. Вероятность переобучения оценим сверху вероятностью большого *равномерного отклонения* частот: для любых  $X^L, \mu$

$$\begin{aligned} Q_\varepsilon(\mu, X^L) &= P[\delta(\mu, X^L) \geq \varepsilon] \leq \\ &\leq P\left[\max_{a \in A} \delta(a, X^L) \geq \varepsilon\right] = \tilde{Q}_\varepsilon(A, X^L). \end{aligned}$$

2. Оценим вероятность объединения событий суммой их вероятностей (неравенство Буля, union bound):

$$\begin{aligned} \tilde{Q}_\varepsilon(A, X^L) &= P \max_{a \in A} [\delta(a, X^L) \geq \varepsilon] \leq \\ &\leq P \sum_{a \in A} [\delta(a, X^L) \geq \varepsilon] = \sum_{a \in A} \underbrace{P[\delta(a, X^L) \geq \varepsilon]}_{Q_\varepsilon(a, X^L)}. \end{aligned}$$

## Теория Вапника–Червоненкиса

Таким образом, доказали важную теорему:

### Теорема

Для любых  $X^L$ ,  $\mu$ , конечного  $A$  и  $\varepsilon \in [0, 1]$

$$\tilde{Q}_\varepsilon(A, X^L) \leq \sum_{a \in A} \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L} (m - \varepsilon k) \right),$$

где  $m = n(a, X^L)$ .

### Следствие (Вапник и Червоненкис, 1968)

Для любых  $X^L$ ,  $\mu$ , конечного  $A$  и  $\varepsilon \in [0, 1]$

$$\begin{aligned} \tilde{Q}_\varepsilon(A, X^L) &\leq |A| \cdot \max_m \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L} (m - \varepsilon k) \right) \leq \\ &\leq |A| \cdot \frac{3}{2} \exp(-\varepsilon^2 \ell), \quad \text{при } \ell = k. \end{aligned}$$

## Обобщение на случай бесконечных семейств $A$

Функция роста  $\Delta^A(L)$  семейства  $A$  — это максимальное по  $X^L$  число различных векторов ошибок  $\vec{a} = (I(a, x_1), \dots, I(a, x_L))$ .  
В оценке надо заменить  $|A|$  на функцию роста  $\Delta^A(L)$ .

Ёмкость (размерность Вапника-Червоненкиса) семейства  $A$  — это максимальная длина выборки  $h$ , для которой  $\Delta^A(h) = 2^h$ .

### Теорема

Если такое  $h$  существует, то  $\Delta^A(L) \leq C_L^0 + \dots + C_L^h \leq \frac{3}{2} \frac{L^h}{h!}$ .

### Теорема

Ёмкость семейства линейных классификаторов на два класса

$$a(x) = \text{sign}(w_1 x^1 + \dots + w_n x^n), \quad x = (x^1, \dots, x^n) \in X.$$

равна размерности пространства параметров,  $\text{VCdim}(A) = n$ .



## Обращение оценки Вапника-Червоненкиса (при $l = k$ )

1. Оценка:  $P \left[ \max_{a \in A} (\nu(a, X^k) - \nu(a, X^\ell)) \geq \varepsilon \right] \leq \Delta \frac{3}{2} \exp(-\ell \varepsilon^2).$

Тогда для любого  $a \in A$  с вероятностью не менее  $(1 - \eta)$

$$\nu(a, X^k) \leq \underbrace{\nu(a, X^\ell)}_{\text{эмпирический риск}} + \underbrace{\sqrt{\frac{1}{\ell} \ln \Delta + \frac{1}{\ell} \ln \frac{3}{2\eta}}}_{\text{штраф за сложность}}.$$

2. Оценка:  $P \left[ \max_{a \in A} (\nu(a, X^k) - \nu(a, X^\ell)) \geq \varepsilon \right] \leq \frac{3}{2} \frac{L^h}{h!} \cdot \frac{3}{2} \exp(-\ell \varepsilon^2).$

Тогда для любого  $a \in A$  с вероятностью не менее  $(1 - \eta)$

$$\nu(a, X^k) \leq \underbrace{\nu(a, X^\ell)}_{\text{эмпирический риск}} + \underbrace{\sqrt{\frac{h}{\ell} \ln \frac{2e\ell}{h} + \frac{1}{\ell} \ln \frac{9}{4\eta}}}_{\text{штраф за сложность}}.$$

## Метод структурной минимизации риска (СМР)

Дано: система вложенных подсемейств возрастающей ёмкости

$$A_0 \subset A_1 \subset \dots \subset A_h \subset \dots$$

Найти: оптимальную ёмкость  $h^*$ , такую, что

$$\nu(a, X^k) \leq \underbrace{\min_{a \in A_h} \nu(a, X^\ell)}_{\text{минимизация эмпирического риска}} + \underbrace{\sqrt{\frac{h}{\ell} \ln \frac{2e\ell}{h} + \frac{1}{\ell} \ln \frac{9}{4\eta}}}_{\text{штраф за сложность}} \rightarrow \min_h$$

Недостатки СМР:

- $h^*$  может оказаться заниженной из-за завышенности  $Q_\varepsilon$ .
- На практике эмпирический CV предпочтительнее этих оценок.

## Причины завышенности оценок Вапника-Червоненкиса

- Оценка равномерного отклонения сильно завышена, когда большая часть алгоритмов имеет исчезающе малую вероятность быть результатом обучения.

На практике распределение

$$q(a) = P[\mu(X^\ell) = a], \quad a \in A$$

как правило, существенно неравномерно!

Будем называть это **эффектом расслоения семейства  $A$** .

- **Неравенство Буля** сильно завышено, когда среди бинарных векторов ошибок есть много похожих.

Будем называть это **эффектом сходства алгоритмов**.

## Оценка «бритва Оккама»

Мы не можем знать распределение  $q(a) = P[\mu(X^\ell) = a]$ , но можем попробовать его «угадать».

### Теорема (Дж. Лангфорд, 2002)

Для произвольной нормированной функции,  $p(a)$ ,  $\sum_{a \in A} p(a) = 1$ , любого  $\eta \in (0, 1)$ , любого  $a \in A$  с вероятностью не менее  $1 - \eta$

$$\nu(a, X^k) \leq \nu(a, X^\ell) + \sqrt{\frac{1}{\ell} \ln \frac{1}{p(a)} + \frac{1}{\ell} \ln \frac{3}{2\eta}}.$$

**Утв 1.** Если угадали,  $p(a) = q(a)$ , то  $E \ln \frac{1}{p(\mu(X^\ell))}$  минимально.

**Утв 2.** Бритва Оккама может учитывать эффект расслоения, но не учитывает эффект схождения, поэтому тоже завышена.

## Оценка «бритва Оккама»: как задать $p(a)$ ?

### Пример 1.

Равномерное распределение  $p(a) = \frac{1}{|A|}$

даёт оценку Вапника–Червоненкиса:

для любых  $a \in A$ ,  $\eta \in (0, 1)$  с вероятностью не менее  $1 - \eta$

$$\nu(a, X^k) \leq \nu(a, X^\ell) + \sqrt{\frac{1}{\ell} \ln |A| + \frac{1}{\ell} \ln \frac{3}{2\eta}}.$$

## Оценка «бритва Оккама»: как ещё задать $p(a)$ ?

**Пример 2.** Задача классификации на 2 класса  $Y = \{-1, +1\}$ ,  
 $A$  — линейные классификаторы в  $\mathbb{R}^n$ :

$$a(x) = \text{sign}(w_1 x^1 + \dots + w_n x^n), \quad x = (x^1, \dots, x^n) \in X.$$

*Гауссовское распределение:* веса  $w \in \mathbb{R}^n$  — независимые с.в.,  
 с нулевым ожиданием и равными дисперсиями  $\sigma^2$ :


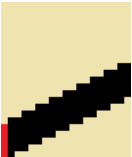


$$p(a) = Z \exp\left(-\frac{1}{2\sigma^2} \|w\|^2\right),$$

где  $Z$  — нормировочный множитель, не зависящий от  $w$ .

Подставляя в «бритву Оккама», получаем... регуляризацию!

$$\nu(a, X^k) \leq \nu(a, X^\ell) + \sqrt{\frac{\|w\|^2}{2\ell\sigma^2} + \frac{1}{\ell} \ln \frac{3}{2\eta Z}}.$$

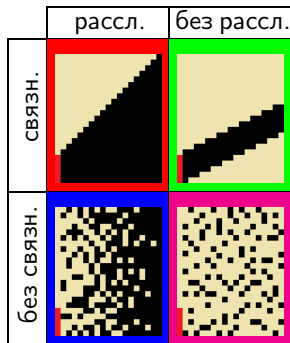
## Эксперимент: четыре семейства алгоритмов, заданных матрицами ошибок; лучший алгоритм у всех одинаков

	с расслоением по числу ошибок	без расслоения по числу ошибок
каждая пара соседних алгоритмов отличается только на одном объекте (образуется <i>цепь</i> )		
соседние алгоритмы существенно различны, ( <i>цепь</i> не образуется)		

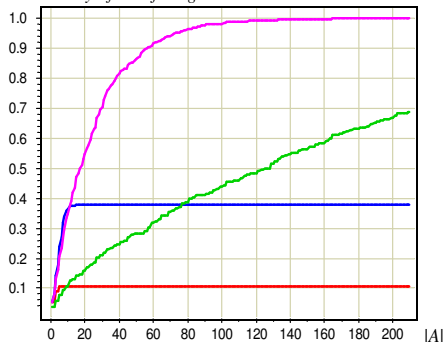
Постепенно добавляя алгоритмы в  $\{a_1, \dots, a_D\}$ , построим зависимости вероятности переобучения  $Q_\epsilon$  от числа  $D$ .

## Эксперимент с «трижды испорченной» монотонной цепью

$\ell = k = 100$ ,  $\varepsilon = 0.05$ ,  $N = 1000$  разбиений Монте-Карло.



Probability of overfitting

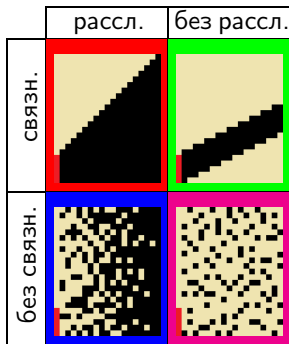


- Огромные семейства с P&C могут почти не переобучаться
- Без P&C даже 30 алгоритмов могут сильно переобучаться

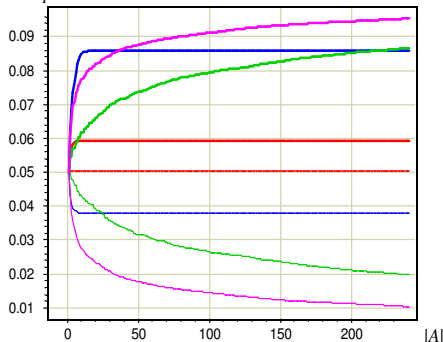


## Эксперимент с «трижды испорченной» монотонной цепью

$\ell = k = 100$ ,  $\varepsilon = 0.05$ ,  $N = 1000$  разбиений Монте-Карло.



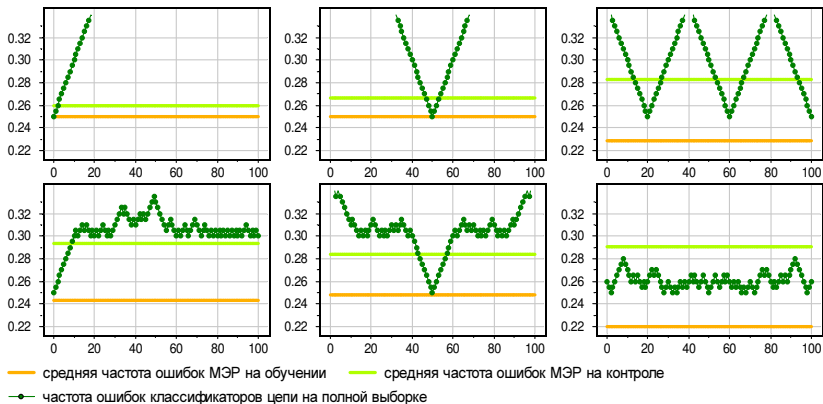
Complete Cross-Validation



- Получение точных оценок вероятности переобучения невозможно без учёта эффектов расслоения и связности.

## Эксперимент. Переобучение цепей с различным расслоением

Условия эксперимента:  $L = 100$ ,  $\ell = 50$ ,  $m = 25$ ,  $\varepsilon = 0.05$ ,  
метод Монте-Карло по  $N = 100000$  случайных разбиений.



## Граф расслоения–связности множества алгоритмов

Определим бинарные отношения на множестве алгоритмов  $A$ :  
частичный порядок  $a \leq b$ :  $I(a, x) \leq I(b, x)$  для всех  $x \in X^L$ ;  
предшествование  $a \prec b$ :  $a \leq b$  и  $\|b - a\| = 1$ .

Опр. Граф расслоения–связности  $\langle A, E \rangle$ :

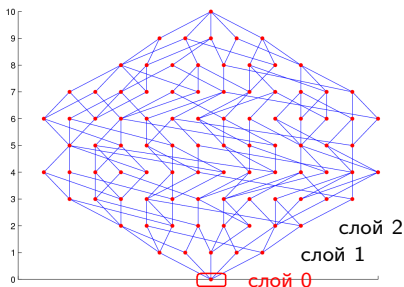
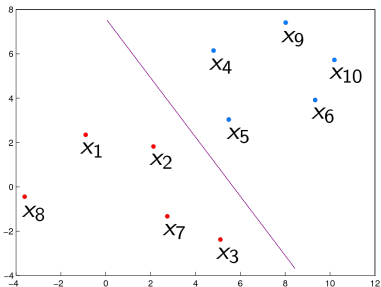
$A$  — множество попарно различных векторов ошибок;

$E = \{(a, b) : a \prec b\}$ .

Свойства графа расслоения–связности:

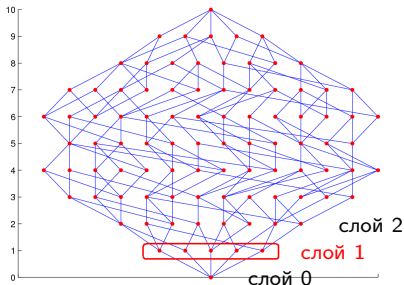
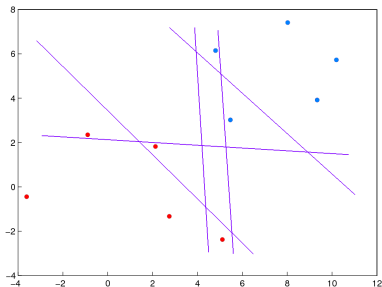
- это подграф графа Хассе отношения порядка  $\leq$  на  $A$ ;
- каждому ребру  $(a, b)$  соответствует объект  $x_{ab} \in X^L$ , такой, что  $I(a, x_{ab}) = 0$ ,  $I(b, x_{ab}) = 1$ ;
- граф является многодольным со слоями  
 $A_m = \{a \in A : m(a, X^L) = m\}$ ,  $m = 0, \dots, L$ ;

## Пример. Семейство линейных алгоритмов классификации



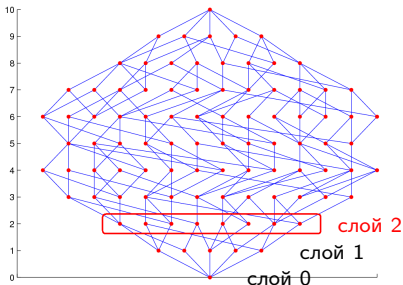
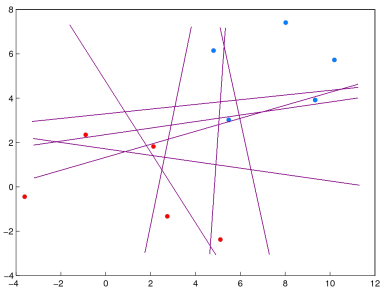
	слой 0
x <sub>1</sub>	0
x <sub>2</sub>	0
x <sub>3</sub>	0
x <sub>4</sub>	0
x <sub>5</sub>	0
x <sub>6</sub>	0
x <sub>7</sub>	0
x <sub>8</sub>	0
x <sub>9</sub>	0
x <sub>10</sub>	0

## Пример. Семейство линейных алгоритмов классификации



	слой 0	слой 1				
X <sub>1</sub>	0	1	0	0	0	0
X <sub>2</sub>	0	0	1	0	0	0
X <sub>3</sub>	0	0	0	1	0	0
X <sub>4</sub>	0	0	0	0	1	0
X <sub>5</sub>	0	0	0	0	0	1
X <sub>6</sub>	0	0	0	0	0	0
X <sub>7</sub>	0	0	0	0	0	0
X <sub>8</sub>	0	0	0	0	0	0
X <sub>9</sub>	0	0	0	0	0	0
X <sub>10</sub>	0	0	0	0	0	0

# Пример. Семейство линейных алгоритмов классификации



	слой 0	слой 1						слой 2								
X <sub>1</sub>	0	1	0	0	0	0	0	1	0	0	0	0	1	1	0	...
X <sub>2</sub>	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	...
X <sub>3</sub>	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	...
X <sub>4</sub>	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	...
X <sub>5</sub>	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	...
X <sub>6</sub>	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	...
X <sub>7</sub>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	...
X <sub>8</sub>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
X <sub>9</sub>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
X <sub>10</sub>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...

## Характеристики **расслоения** и **связности** алгоритма $a \in A$

Множество объектов, соответствующих рёбрам, исходящим из  $a$ :

$$X_a = \{x_{ab} \in X^L \mid a \prec b\},$$

Множество объектов, соответствующих всем рёбрам на путях, ведущих в  $a$ :

$$X'_a = \{x \in X^L \mid \exists b \in A: b \prec a, l(b, x) < l(a, x)\}.$$

**Опр.** Характеристики **расслоения** и **связности** алгоритма  $a$ :

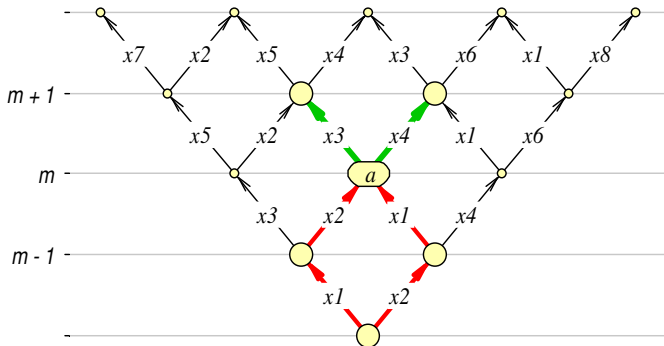
$u(a) = |X_a|$  — **верхняя связность** алгоритма  $a$ ;

$q(a) = |X'_a|$  — **неполноценность** алгоритма  $a$ ;

## Пример: двумерная сеть алгоритмов

Верхняя связность:  $u(a) = \#\{x_3, x_4\} = 2$ ;

Неполноценность:  $q(a) = \#\{x_1, x_2\} = 2$ ;





## Монотонные методы обучения

**Опр.** Метод обучения  $\mu$  называется *монотонным*, если

$$\mu(X^\ell) = \arg \min_{a \in A} K(a, X^\ell),$$

где  $K(a, X)$  — строго монотонная функция вектора ошибок  $a$ :

$$\forall X \subset X^L, \forall a, b \in A \quad a < b \rightarrow K(a, X) < K(b, X).$$

**Пример 1.** Взвешенный эмпирический риск с весами  $w_i$ :

$$K(a, X) = \sum_{x_i \in X} w_i I(a, x_i), \quad w_i > 0.$$

**Опр.** Монотонный метод  $\mu$  называется *пессимистичным*, если

$$\mu(X^\ell) = \arg \max_{a \in A(X^\ell)} \delta(a, X^\ell), \quad A(X^\ell) = \text{Arg} \min_{a \in A} K(a, X^\ell).$$

## Верхняя оценка расслоения–связности

### Теорема (Воронцов, Ивахненко, Решетняк, 2010)

Для любого монотонного метода  $\mu$ , любых  $X^L$ ,  $A$  и  $\varepsilon \in (0, 1)$

$$Q_\varepsilon \leq \sum_{a \in A} \frac{C_{L-u-q}^{\ell-u}}{C_L^\ell} \mathcal{H}_{L-u-q}^{\ell-u, m-q} \left( \frac{\ell}{L} (m - \varepsilon k) \right)$$

где  $u = |X_a|$  — верхняя связность алгоритма  $a$ ,

$q = |X'_a|$  — неполноценность алгоритма  $a$ ,

$m = m(a, X^L)$  — число ошибок алгоритма  $a$ ,

$$\mathcal{H}_L^{\ell, m}(z) = \sum_{s=0}^{\lfloor z \rfloor} \frac{C_m^s C_{L-m}^{\ell-s}}{C_L^\ell}, \quad z = 0, \dots, \ell$$

— функция гипергеометрического распределения:

## Идея доказательства

1. Пусть  $\mu$  — произвольный монотонный метод обучения,  $\bar{\mu}$  — монотонный пессимистичный метод обучения. Тогда

$$Q_\varepsilon(\mu, X^L) \leq Q_\varepsilon(\bar{\mu}, X^L).$$

2. Если  $\bar{\mu}(X^\ell) = a$ , то  $\begin{cases} X_a \subseteq X^\ell & \text{в силу пессимистичности } \bar{\mu}, \\ X'_a \subseteq X^k & \text{в силу монотонности } \bar{\mu}. \end{cases}$

$$3. P[\bar{\mu}(X^\ell) = a] \leq P[\underbrace{X_a \subseteq X^\ell \text{ и } X'_a \subseteq X^k}_{S(a, X^\ell)}] = \frac{C_{L-|X_a|-|X'_a|}^{\ell-|X_a|}}{C_L^\ell} = \frac{C_{L-u-q}^{\ell-u}}{C_L^\ell}.$$

4. По формуле полной вероятности:

$$Q_\varepsilon(\bar{\mu}, X^L) = \sum_{a \in A} \underbrace{P[S(a, X^\ell)]}_{C_{L-u-q}^{\ell-u} / C_L^\ell} \cdot \underbrace{P[\delta(a, X^\ell) \geq \varepsilon \mid S(a, X^\ell)]}_{\mathcal{H}_{L-u-q}^{\ell-u, m-q} \left( \frac{\ell}{L} (m - \varepsilon k) \right)}. \quad \blacksquare$$

## Свойства оценки

$$Q_\varepsilon \leq \sum_{a \in A} \frac{C_{L-u-q}^{\ell-u}}{C_L^\ell} \mathcal{H}_{L-u-q}^{\ell-u, m-q} \left( \frac{\ell}{L} (m - \varepsilon k) \right)$$

- 1 Вклад алгоритма  $a \in A$  убывает экспоненциально по  $u(a) \Rightarrow$  **связные семейства меньше переобучаются;** по  $q(a) \Rightarrow$  **только нижние слои вносят вклад в  $Q_\varepsilon$ .**
- 2 Оценка обращается в равенство в случае многомерных монотонных сетей алгоритмов.
- 3 Вероятность получить алгоритм в результате обучения

$$P[\mu(X^\ell) = a] \leq P_a = \frac{C_{L-u-q}^{\ell-u}}{C_L^\ell}.$$

- 4 Если  $q(a) > k$ , то  $P_a = 0$  и вклад алгоритма  $a$  равен 0  $\Rightarrow$  при малых  $k$  оценка вырождается.
- 5  $\sum_{a \in A} P_a$  — оценка степени завышенности.

## Свойства оценки

$$Q_\varepsilon \leq \sum_{a \in A} \frac{C_{L-u-q}^{\ell-u}}{C_L^\ell} \mathcal{H}_{L-u-q}^{\ell-u, m-q} \left( \frac{\ell}{L} (m - \varepsilon k) \right)$$

- 6 При  $|A| = 1$  это вероятность большого отклонения частот в двух выборках:

$$Q_\varepsilon = \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L} (m - \varepsilon k) \right) \rightarrow 0 \text{ при } \ell, k \rightarrow \infty.$$

- 7 При  $q = u = 0$  и  $\ell = k$  это оценка Вапника-Червоненкиса:

$$Q_\varepsilon \leq \sum_{a \in A} \mathcal{H}_L^{\ell, m} \left( \frac{\ell}{L} (m - \varepsilon k) \right) \leq |A| \cdot \frac{3}{2} \exp(-\varepsilon \ell^2).$$

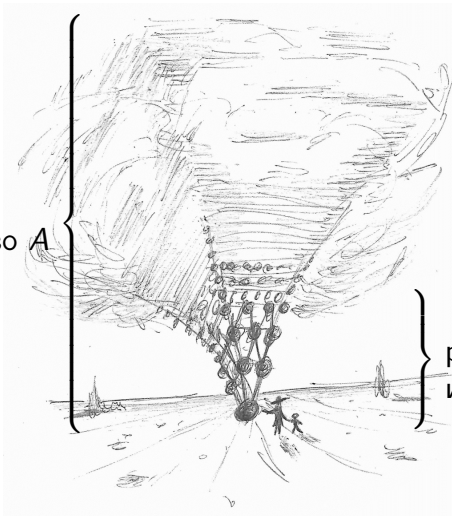
- 8 При замене неполноценности  $q$  на нижнюю связность  $d$  это верхняя оценка функционала равномерного отклонения

$$\tilde{Q}_\varepsilon(A, X^L) = P \left[ \sup_{a \in A} (\nu(a, X^k) - \nu(a, X^\ell)) \geq \varepsilon \right],$$

который учитывает связность, но не учитывает расслоение.

## Идея использования оценок расслоения–связности

всё семейство  $A$



реально  
используемая часть  $A$

## Резюме

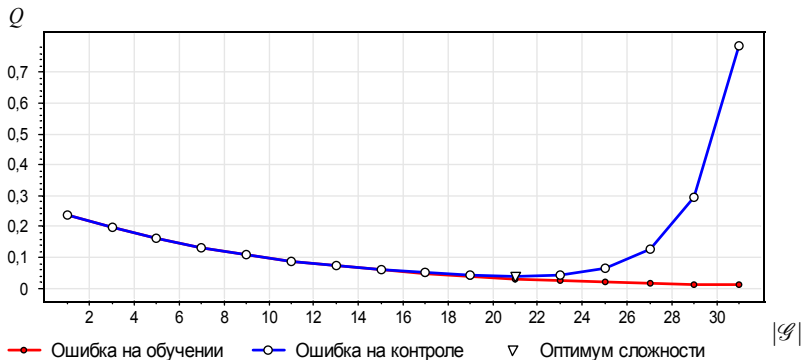
- Свойства расслоения и связности уменьшают переобучение.
- На практике семейства, как правило, ими обладают. Иначе вероятность переобучения была бы близка к 1 уже при  $|A|$  порядка нескольких десятков.
- Практическое применение комбинаторных оценок:
  - 1) оценить  $\eta = Q_\varepsilon(\mu, X^L)$  по нескольким нижним слоям;
  - 2) применив обращение, оценить  $\varepsilon$  через  $\eta$ ;
  - 3) использовать оценку  $\nu(X^\ell) + \varepsilon$  как внешний критерий для выбора модели, метода или отбора признаков.
- Информацию о нижних слоях можно получать в ходе итераций метода обучения, либо случайным блужданием по  $A$  [Соколов, 2013].

## Задача отбора признаков по внешнему критерию

$\mathcal{F} = \{f_j: X \rightarrow D_j: j = 1, \dots, n\}$  — множество признаков;

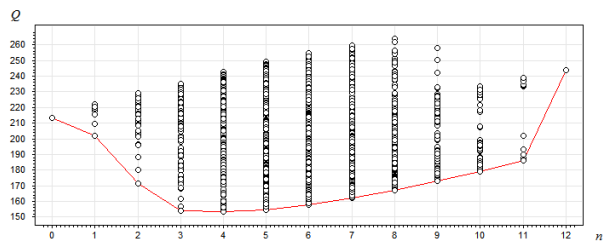
$\mu_{\mathcal{G}}$  — метод обучения, использующий только признаки  $\mathcal{G} \subseteq \mathcal{F}$ ;

$Q(\mathcal{G}) = Q(\mu_{\mathcal{G}}, X^{\ell})$  — какой-либо внешний критерий.





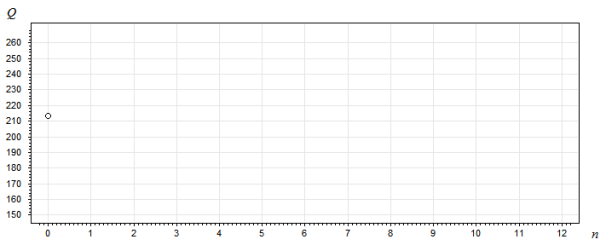
## Алгоритм полного перебора (Full Search)



**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

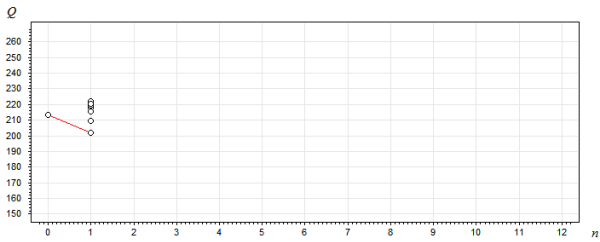


$$j = 0$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

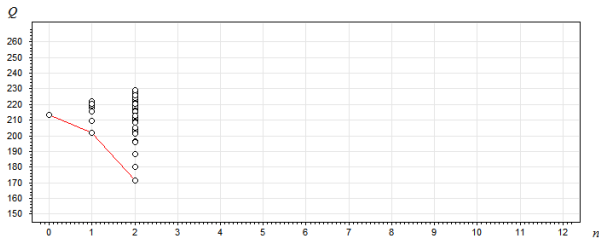


$$j = 1$$
$$j^* = 1$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

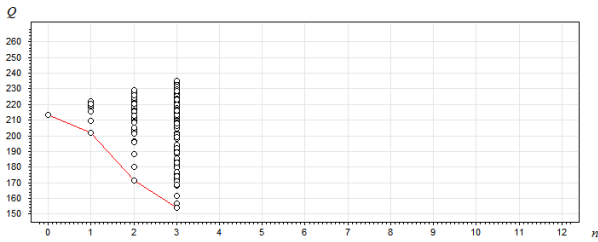


$$j = 2$$
$$j^* = 2$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

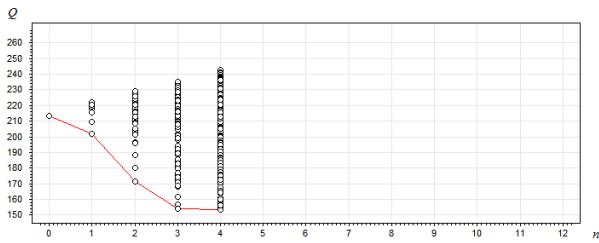


$$j = 3$$
$$j^* = 3$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

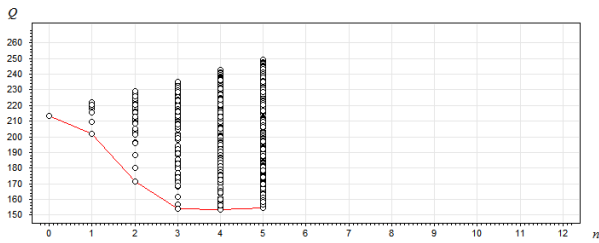


$$j = 4$$
$$j^* = 4$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

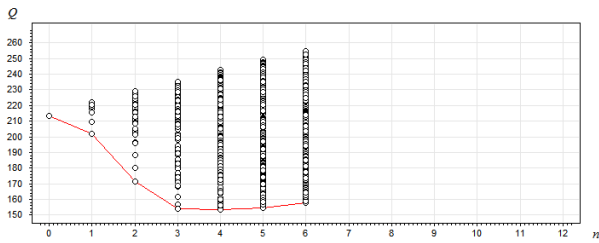


$$j = 5$$
$$j^* = 4$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)



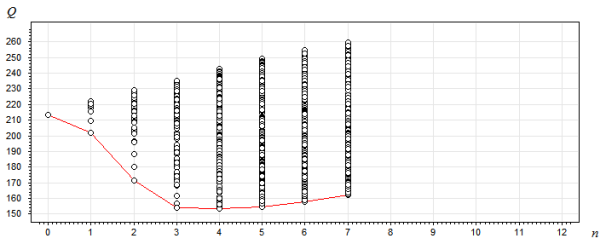
$$j = 6$$
$$j^* = 4$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;



## Алгоритм полного перебора (Full Search)



$$j = 7$$
$$j^* = 4$$

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

- 1:  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти лучший набор сложности  $j$ :  
$$\mathcal{G}_j := \arg \min_{\mathcal{G}: |\mathcal{G}|=j} Q(\mathcal{G});$$
- 4: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 5: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм полного перебора (Full Search)

### Преимущества:

- простота реализации;
- гарантированный результат;
- полный перебор эффективен, когда
  - информативных признаков не много,  $j^* \lesssim 5$ ;
  - всего признаков не много,  $n \lesssim 20..100$ .

### Недостатки:

- в остальных случаях оооооочень долго —  $O(2^n)$ ;
- чем больше перебирается вариантов, тем больше переобучение (особенно, если лучшие из вариантов существенно различны и одинаково плохи).

### Способы устранения:

- эвристические методы сокращённого перебора.

## Алгоритм жадного добавления (Add)

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметр  $d$ ;

---

- 1:  $\mathcal{G}_0 := \emptyset$ ;  $Q^* := Q(\emptyset)$ ; — инициализация;
- 2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:
- 3: найти признак, наиболее выгодный для добавления:  
$$f^* := \arg \min_{f \in \mathcal{F} \setminus \mathcal{G}_{j-1}} Q(\mathcal{G}_{j-1} \cup \{f\});$$
- 4: добавить этот признак в набор:  
$$\mathcal{G}_j := \mathcal{G}_{j-1} \cup \{f^*\};$$
- 5: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 6: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Алгоритм жадного добавления (Add)

### Преимущества:

- работает быстро —  $O(n^2)$ , точнее  $O(n(j^* + d))$ ;
- возможны быстрые инкрементные алгоритмы;

### Недостатки:

- Add склонен включать в набор лишние признаки.

### Способы устранения:

- Add-Del — чередование добавлений и удалений (см. далее);
- поиск в ширину (см. ещё далее).

## Алгоритм поочерёдного добавления и удаления (Add-Del)

1:  $\mathcal{G}_0 := \emptyset$ ;  $Q^* := Q(\emptyset)$ ;  $t := 0$ ; — инициализация;

2: **повторять**

---

3: **пока**  $|\mathcal{G}_t| < n$  добавлять признаки (Add):

4:  $t := t + 1$ ; — началась следующая итерация;

5:  $f^* := \arg \min_{f \in \mathcal{F} \setminus \mathcal{G}_{t-1}} Q(\mathcal{G}_{t-1} \cup \{f\})$ ;  $\mathcal{G}_t := \mathcal{G}_{t-1} \cup \{f^*\}$ ;

6: **если**  $Q(\mathcal{G}_t) < Q^*$  **то**  $t^* := t$ ;  $Q^* := Q(\mathcal{G}_t)$ ;

7: **если**  $t - t^* \geq d$  **то прервать цикл**;

---

8: **пока**  $|\mathcal{G}_t| > 0$  удалять признаки (Del):

9:  $t := t + 1$ ; — началась следующая итерация;

10:  $f^* := \arg \min_{f \in \mathcal{G}_{t-1}} Q(\mathcal{G}_{t-1} \setminus \{f\})$ ;  $\mathcal{G}_t := \mathcal{G}_{t-1} \setminus \{f^*\}$ ;

11: **если**  $Q(\mathcal{G}_t) < Q^*$  **то**  $t^* := t$ ;  $Q^* := Q(\mathcal{G}_t)$ ;

12: **если**  $t - t^* \geq d$  **то прервать цикл**;

---

13: **пока** значения критерия  $Q(\mathcal{G}_{t^*})$  уменьшаются;

14: **вернуть**  $\mathcal{G}_{t^*}$ ;

## Алгоритм поочерёдного добавления и удаления (Add-Del)

### Преимущества:

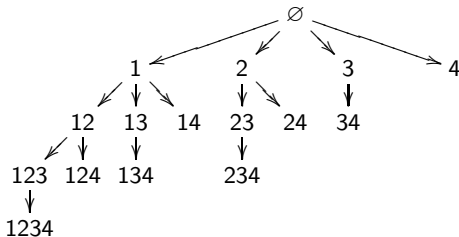
- как правило, лучше, чем Add и Del по отдельности;
- возможны быстрые инкрементные алгоритмы (пример — *шаговая регрессия*).

### Недостатки:

- работает дольше, оптимальность не гарантирует.

## Поиск в глубину (DFS, метод ветвей и границ)

Пример: дерево наборов признаков,  $n = 4$



Основные идеи:

- нумерация признаков по возрастанию номеров — чтобы избежать повторов при переборе подмножеств;
- если набор  $\mathcal{S}$  бесперспективен, то больше не пытаться его наращивать.

## Поиск в глубину (DFS, метод ветвей и границ)

Обозначим  $Q_j^*$  — значение критерия на самом лучшем наборе мощности  $j$  из всех до сих пор просмотренных.

**Оценка бесперспективности** набора признаков  $\mathcal{G}$ :  
набор  $\mathcal{G}$  не наращивается, если

$$\exists j: \quad Q(\mathcal{G}) \geq \kappa Q_j^* \quad \text{и} \quad |\mathcal{G}| \geq j + d,$$

$d \geq 0$  — целочисленный параметр,

$\kappa \geq 1$  — вещественный параметр.

Чем меньше  $d$  и  $\kappa$ , тем сильнее сокращается перебор.



## Поиск в глубину (DFS, метод ветвей и границ)

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметры  $d$  и  $\kappa$ ;

---

- 1: **ПРОЦЕДУРА** Нарастить ( $\mathcal{G}$ );
  - 2: **если** найдётся  $j \leq |\mathcal{G}| - d$  такое, что  $Q(\mathcal{G}) \geq \kappa Q_j^*$ , **то**
  - 3: **выход**;
  - 4:  $Q_{|\mathcal{G}|}^* := \min\{Q_{|\mathcal{G}|}^*, Q(\mathcal{G})\}$ ;
  - 5: **для всех**  $f_s \in \mathcal{F}$  таких, что  $s > \max\{t \mid f_t \in \mathcal{G}\}$   
Нарастить ( $\mathcal{G} \cup \{f_s\}$ );
- 

- 6: Инициализация массива лучших значений критерия:  
 $Q_j^* := Q(\emptyset)$  для всех  $j = 1, \dots, n$ ;
- 7: Упорядочить признаки по убыванию информативности;
- 8: Нарастить ( $\emptyset$ );
- 9: **вернуть**  $\mathcal{G}$ , для которого  $Q(\mathcal{G}) = \min_{j=1, \dots, n} Q_j^*$ ;

## Поиск в ширину (BFS)

Он же *многорядный итерационный алгоритм МГУА*  
(МГУА — метод группового учёта аргументов).

Философский принцип *неокончателных решений* Габора:  
принимая решения, следует оставлять максимальную свободу  
выбора для принятия последующих решений.

Усовершенствуем алгоритм Add:  
на каждой  $j$ -й итерации будем строить не один набор,  
а множество из  $B_j$  наборов, называемое  $j$ -м *рядом*:

$$R_j = \{\mathcal{G}_j^1, \dots, \mathcal{G}_j^{B_j}\}, \quad \mathcal{G}_j^b \subseteq \mathcal{F}, \quad |\mathcal{G}_j^b| = j, \quad b = 1, \dots, B_j.$$

где  $B_j \leq B$  — параметр *ширины поиска*.

## Поиск в ширину (BFS)

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметры  $d, B$ ;

---

1: первый ряд состоит из всех наборов длины 1:

$$R_1 := \{\{f_1\}, \dots, \{f_n\}\}; \quad Q^* = Q(\emptyset);$$

2: **для всех**  $j = 1, \dots, n$ , где  $j$  — сложность наборов:

3: отсортировать ряд  $R_j = \{\mathcal{G}_j^1, \dots, \mathcal{G}_j^{B_j}\}$

по возрастанию критерия:  $Q(\mathcal{G}_j^1) \leq \dots \leq Q(\mathcal{G}_j^{B_j})$ ;

4: **если**  $B_j > B$  **то**

5:  $R_j := \{\mathcal{G}_j^1, \dots, \mathcal{G}_j^B\}$ ; —  $B$  лучших наборов ряда;

6: **если**  $Q(\mathcal{G}_j^1) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j^1)$ ;

7: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}^1$ ;

8: породить следующий ряд:

$$R_{j+1} := \{\mathcal{G} \cup \{f\} \mid \mathcal{G} \in R_j, f \in \mathcal{F} \setminus \mathcal{G}\};$$

## Поиск в ширину (BFS)

- **Трудоёмкость:**  
 $O(Bn^2)$ , точнее  $O(Bn(j^* + d))$ .
- **Проблема дубликатов:**  
после сортировки (шаг 3) проверить на совпадение только соседние наборы с равными значениями внутреннего и внешнего критерия.
- **Адаптивный отбор признаков:**  
на шаге 8 добавлять к  $j$ -му ряду только признаки  $f$  с наибольшей информативностью  $I_j(f)$ :

$$I_j(f) = \sum_{b=1}^{B_j} [f \in \mathcal{G}_j^b].$$

## Генетический алгоритм поиска (идея и терминология)

$\mathcal{G} \subseteq \mathcal{F}$  — индивид (в МГУА «модель»);

$R_t := \{\mathcal{G}_t^1, \dots, \mathcal{G}_t^{B_t}\}$  — поколение (в МГУА — «ряд»);

$\beta = (\beta_j)_{j=1}^n$ ,  $\beta_j = [f_j \in \mathcal{G}]$  — хромосома, кодирующая  $\mathcal{G}$ ;

Бинарная операция скрещивания  $\beta = \beta' \times \beta''$ :

$$\beta_j = \begin{cases} \beta'_j, & \text{с вероятностью } 1/2; \\ \beta''_j, & \text{с вероятностью } 1/2; \end{cases}$$

Унарная операция мутации  $\beta = \sim \beta'$

$$\beta_j = \begin{cases} 1 - \beta'_j, & \text{с вероятностью } p_m; \\ \beta'_j, & \text{с вероятностью } 1 - p_m; \end{cases}$$

где параметр  $p_m$  — вероятность мутации.

## Генетический (эволюционный) алгоритм

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметры:  $d, p_m$ ,  
 $B$  — размер популяции,  $T$  — число поколений;

- 
- 1: инициализировать случайную популяцию из  $B$  наборов:  
 $B_1 := B$ ;  $R_1 := \{\mathcal{G}_1^1, \dots, \mathcal{G}_1^{B_1}\}$ ;  $Q^* := Q(\emptyset)$ ;
  - 2: **для всех**  $t = 1, \dots, T$ , где  $t$  — номер поколения:
  - 3: ранжирование индивидов:  $Q(\mathcal{G}_t^1) \leq \dots \leq Q(\mathcal{G}_t^{B_t})$ ;
  - 4: **если**  $B_t > B$  **то**
  - 5:     селекция:  $R_t := \{\mathcal{G}_t^1, \dots, \mathcal{G}_t^B\}$ ;
  - 6: **если**  $Q(\mathcal{G}_t^1) < Q^*$  **то**  $t^* := t$ ;  $Q^* := Q(\mathcal{G}_t^1)$ ;
  - 7: **если**  $t - t^* \geq d$  **то вернуть**  $\mathcal{G}_{t^*}^1$ ;
  - 8: породить  $t+1$ -е поколение путём скрещиваний и мутаций:  
 $R_{t+1} := \{\sim(\mathcal{G}' \times \mathcal{G}'') \mid \mathcal{G}', \mathcal{G}'' \in R_t\} \cup R_t$ ;

## Эвристики для управления процессом эволюции

- Увеличивать вероятности перехода признаков от более успешного родителя к потомку.
- Накапливать оценки информативности признаков. Чем более информативен признак, тем выше вероятность его включения в набор во время мутации.
- Применение совокупности критериев качества.
- Скрещивать только лучшие индивиды (элитаризм).
- Переносить лучшие индивиды в следующее поколение.
- В случае стагнации увеличивать вероятность мутаций.
- Параллельно выращивается несколько изолированных популяций (островная модель эволюции).

## Генетический (эволюционный) алгоритм

### Преимущества:

- it is fun!
- возможность введения различных эвристик;
- решает задачи даже с очень большим числом признаков.

### Недостатки:

- относительно медленная сходимость;
- отсутствие теории;
- подбор параметров — непростое искусство;



## Случайный поиск — упрощенный генетический алгоритм

**Модификация:** шаг 8

- породить  $t+1$ -е поколение путём многократных *мутаций*:

$$R_{t+1} := \{\sim \mathcal{G}, \dots, \sim \mathcal{G} \mid \mathcal{G} \in R_t\} \cup R_t;$$

**Недостатки:**

- ничем не лучше ГА;
- очень медленная сходимость.

**Способ устранения:**

- СПА — случайный поиск с адаптацией.

**Основная идея адаптации:**

- увеличивать вероятность появления тех признаков, которые часто входят в наилучшие наборы,
- одновременно уменьшать вероятность появления признаков, которые часто входят в наихудшие наборы.

## Случайный поиск с адаптацией (СПА)

**Вход:** множество  $\mathcal{F}$ , критерий  $Q$ , параметры  $d, j_0, T, r, h$ ;

- 1:  $p_1 = \dots = p_n := 1/n$ ; — равные вероятности признаков;
- 2: **для всех**  $j = j_0, \dots, n$ , где  $j$  — сложность наборов:
- 3: **для всех**  $t = 1, \dots, T$ , где  $t$  — номер итерации:
- 4:  $r$  случайных наборов признаков из распределения  $\{p_1, \dots, p_n\}$ :  
 $R_{jt} := \{\mathcal{G}_{jt}^1, \dots, \mathcal{G}_{jt}^r\}, \quad |\mathcal{G}_{jt}^1| = \dots = |\mathcal{G}_{jt}^r| = j$ ;
- 5:  $\mathcal{G}_{jt}^{\min} := \arg \min_{\mathcal{G} \in R_{jt}} Q(\mathcal{G})$ ; — лучший из  $r$  наборов;
- 6:  $\mathcal{G}_{jt}^{\max} := \arg \max_{\mathcal{G} \in R_{jt}} Q(\mathcal{G})$ ; — худший из  $r$  наборов;
- 7:  $H := 0$ ; наказание для всех  $f_s \in \mathcal{G}_{jt}^{\max}$ :  
 $\Delta p_s := \min\{p_s, h\}; \quad p_s := p_s - \Delta p_s; \quad H := H + \Delta p_s$ ;
- 8: поощрение для всех  $f_s \in \mathcal{G}_{jt}^{\min}$ :  $p_s := p_s + H/j$ ;
- 9:  $\mathcal{G}_j := \arg \min_{\mathcal{G} \in R_{j1}, \dots, R_{jT}} Q(\mathcal{G})$ ; — лучший набор сложности  $j$ ;
- 10: **если**  $Q(\mathcal{G}_j) < Q^*$  **то**  $j^* := j$ ;  $Q^* := Q(\mathcal{G}_j)$ ;
- 11: **если**  $j - j^* \geq d$  **то вернуть**  $\mathcal{G}_{j^*}$ ;

## Случайный поиск с адаптацией (СПА)

**Рекомендации по выбору параметров  $r, T, h$ :**

$T \approx 10..50$  — число итераций;

$r \approx 20..100$  — число наборов, создаваемых на каждой итерации;

$h \approx \frac{1}{rn}$  — скорость адаптации;

**Преимущества:**

- трудоёмкость порядка  $O(Tr(j^* + d))$  операций;
- меньшее число параметров, по сравнению с генетикой;
- довольно быстрая сходимость.

**Недостатки:**

- при большом числе признаков СПА малоэффективен.

## Резюме в конце лекции

- Отбор признаков надо вести по внешним критериям.
- Критерии, основанные на оценках обобщающей способности, наиболее вычислительно эффективны.
- Для отбора признаков могут использоваться любые эвристические методы дискретной оптимизации

$$Q(\mathcal{G}) \rightarrow \min_{\mathcal{G} \subseteq \mathcal{F}} .$$

- Большинство эвристик эксплуатируют две основные идеи:
  - среди признаков есть наиболее информативные;
  - $Q(\mathcal{G})$  изменяется не сильно при небольшом изменении  $\mathcal{G}$ .
- МГУА, ГА, СПА очень похожи — на их основе легко создавать различные «симбиотические» алгоритмы.