Построение метрик и отбор признаков

К. В. Воронцов, А.В. Зухба
vokov@forecsys.ru
a__l@mail.ru

февраль 2016

Содержание

Задача построения метрики

X — объекты, Y — ответы (идентификаторы классов);
$X^\ell = (x_i, y_i)_{i=1}^\ell$ — обучающая выборка;
$f_j : X \to D_j$, $j = 1, \ldots, n$ — признаки объектов (features).
Матрица «*объекты–признаки*» (features data)

$$F = \|f_j(x_i)\|_{\ell \times n} = \begin{pmatrix} f_1(x_1) & \ldots & f_n(x_1) \\ \ldots & \ldots & \ldots \\ f_1(x_\ell) & \ldots & f_n(x_\ell) \end{pmatrix}$$

Задача построения метрики:

- Как построить функцию расстояния $\rho : X \times X \to [0, \infty)$?
- Как оценить качество функции расстояния?

Гипотезы компактности и непрерывности

Гипотеза непрерывности (для регрессии):
близким объектам соответствуют близкие ответы.

Гипотеза компактности (для классификации):
близкие объекты, как правило, лежат в одном классе.

Формализация понятия «близости»:
задана функция расстояния $\rho\colon X \times X \to [0, \infty)$.

Пример. Евклидово расстояние и его обобщение:

$$\rho(x, x_i) = \left(\sum_{j=1}^{n} \left|x^j - x_i^j\right|^2\right)^{1/2} \qquad \rho(x, x_i) = \left(\sum_{j=1}^{n} w_j \left|x^j - x_i^j\right|^p\right)^{1/p}$$

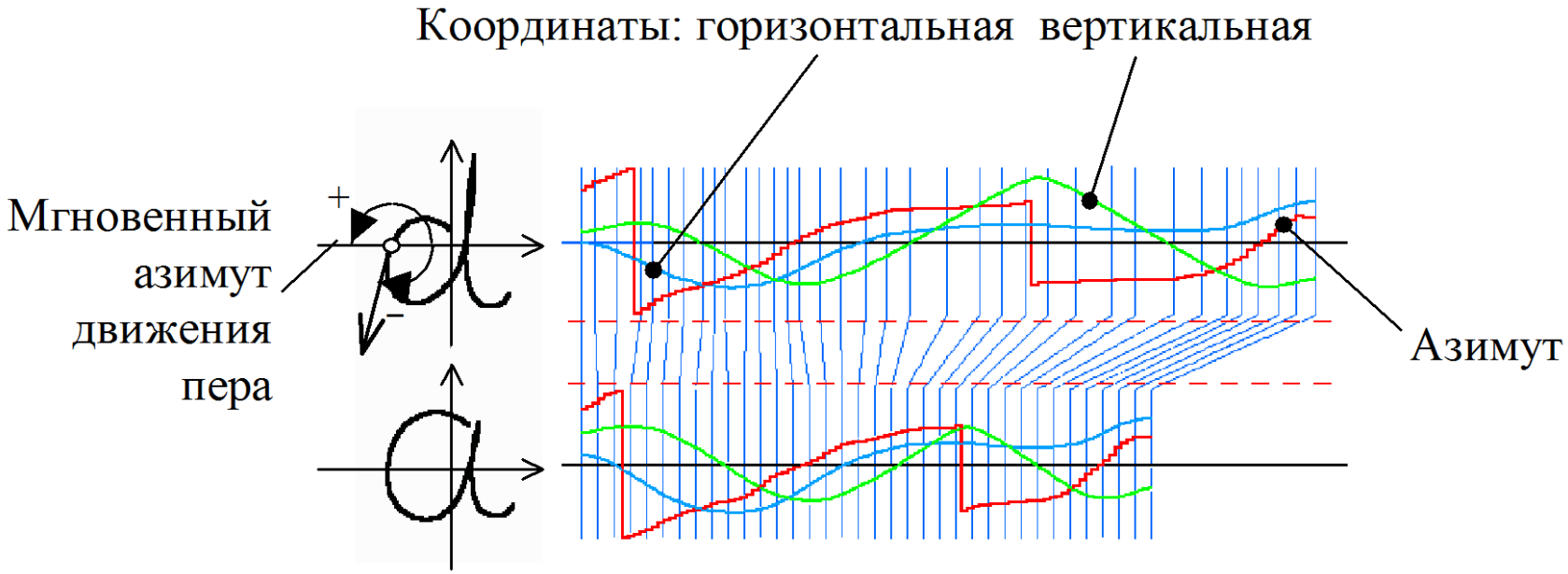
$x = (x^1, \ldots, x^n)$ — вектор признаков объекта x,
$x_i = (x_i^1, \ldots, x_i^n)$ — вектор признаков объекта x_i.

Ещё примеры расстояний:

— **между текстами** (редакторское расстояние Левенштейна):

CTGGGCTAAAAGGTCCCTTAGCC..TTTAGAAAAA.GGGCCATTAGGAAATTGC
CTGGGACTAAA....CCTTAGCCTATTTACAAAAATGGGCCATTAGG...TTGC

— **между сигналами** (энергия сжатий и растяжений):



Стандартные метрики

Взвешенное расстояние Минковского

$$\rho_p(x,z) = \left(\sum_{i=1}^{n} w_i \left| f_i(x) - f_i(z) \right|^p \right)^{\frac{1}{p}}; \quad w_i \geqslant 0.$$

Евклидова метрика ($p = 2$)

$$\rho_2(x,z) = \left(\sum_{i=1}^{n} \left| f_i(x) - f_i(z) \right|^2 \right)^{\frac{1}{2}}.$$

Манхэттенское расстояние ($p = 1$) Минимальная длина пути из x в z при условии, что можно двигаться только параллельно осям координат.

Стандартные метрики

Метрика Чебышева ($p = \infty$) выбирает наибольшее из расстояний между векторами по каждой координате:

$$\rho_\infty(x, z) = \max_{i=1,\dots,n} |f_i(x) - f_i(z)| .$$

«Считающее» расстояние ($p = 0$) равно числу координат, по которым векторы x и z различаются:

$$\rho_0(x, z) = \sum_{i=1}^{n} [f_i(x) \neq f_i(z)] .$$

Стандартные метрики

Косинусная мера

$$\rho_{cos}(x,z) = arccos\left(\frac{\langle x,z\rangle}{\|x\|\,\|z\|}\right) = arccos\left(\frac{\sum_{i=1}^{n} f_i(x)f_i(z)}{\left(\sum_{i=1}^{n} f_i^2(x)\right)^{0,5}\left(\sum_{i=1}^{n} f_i^2(z)\right)^{0,5}}\right)$$

Расстояние Джаккарда используется, если объетами являются множества.

$$\rho_J(A,B) = 1 - \frac{|A\cap B|}{|A\cup B|}.$$

Редакторское расстояние равно минимальному числу вставок и удалений символов, с помощью которых можно преобразовать первую строку ко второй.

Стандартные метрики

Расстояние Махалонобиса

$$\rho_m(x, z) = \sqrt{(x - z)^T S^{-1}(x - z)}.$$

$$\widehat{S} = \frac{1}{n-1} X^T X$$

Теорема

Выборочная ковариационная матрица является неотрицательно определенной.

LOO и профиль компактности

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

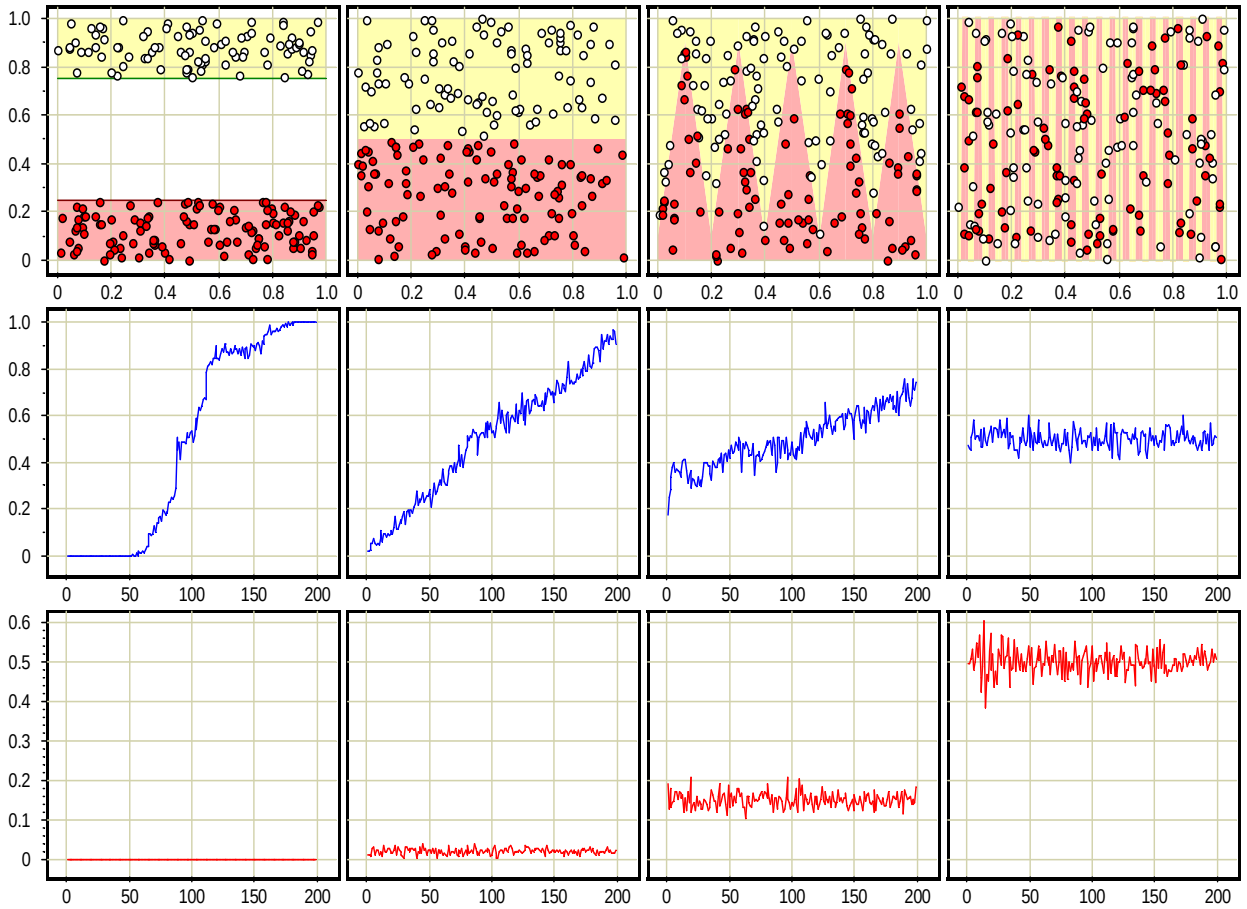
$$\mathrm{LOO}(\mu, X^L) = \frac{1}{L}\sum_{i=1}^{L} Q\big(\mu(X^L \setminus \{x_i\}), \{x_i\}\big).$$

Профиль компактности выборки X^L — это функция доли объектов x_i, у которых m-й сосед $x_i^{(m)}$ лежит в другом классе:

$$K(m, X^L) = \frac{1}{L}\sum_{i=1}^{L} \big[y_i \neq y_i^{(m)}\big]; \quad m = 1, \ldots, L-1,$$

где $x_i^{(m)}$ — m-й сосед объекта x_i среди X^L;
$y_i^{(m)}$ — ответ на m-м соседе объекта x_i.

Профили компактности



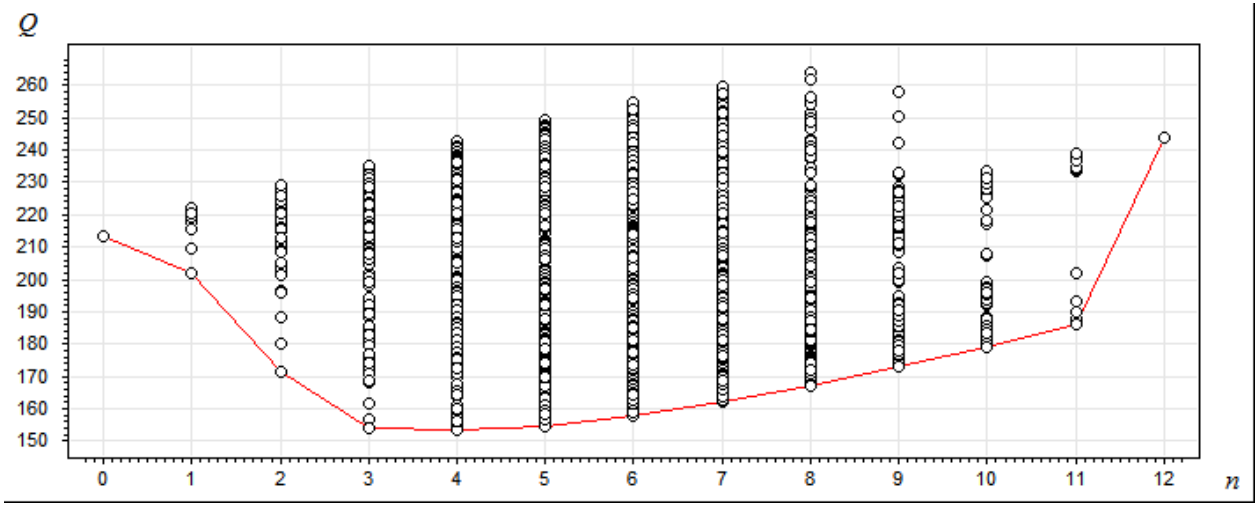
средний ряд: профили компактности,
нижний ряд: зависимость CCV от длины контроля k.

Проклятие размерности

Проклятие размерности — проблема, связанная с экспоненциальным возрастанием количества данных из-за увеличения размерности пространства.

- Трудоемкость вычислений.
- Необходимость хранения огромного количества данных.
- Увеличение доли шумов.
- Расстояния во всех парах объектов стремятся к одному и тому же значению, а значит, становятся неинформативными.

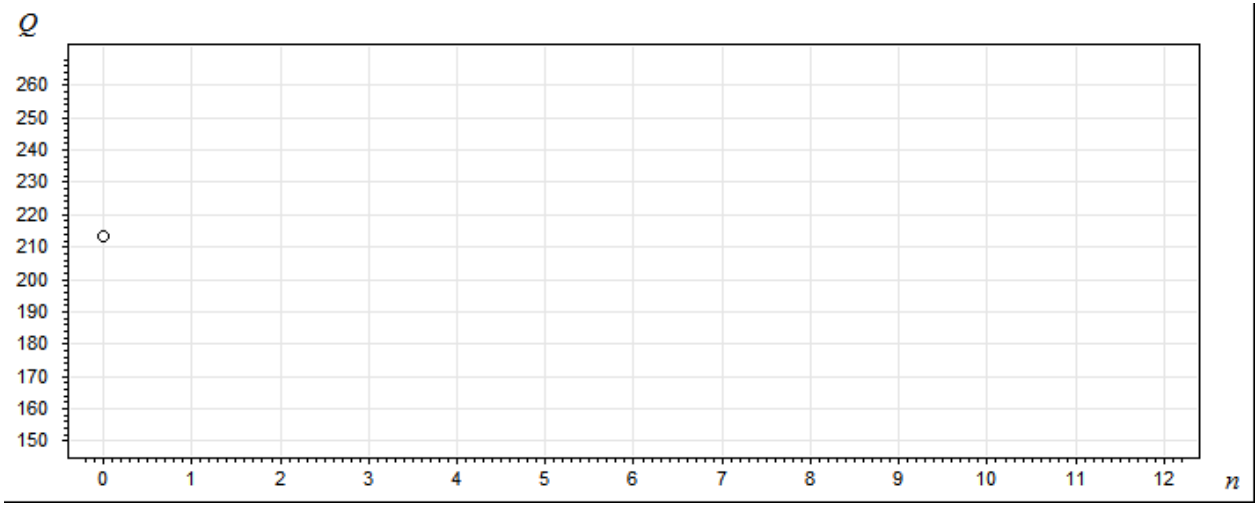
Алгоритм полного перебора (Full Search)



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

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

Алгоритм полного перебора (Full Search)

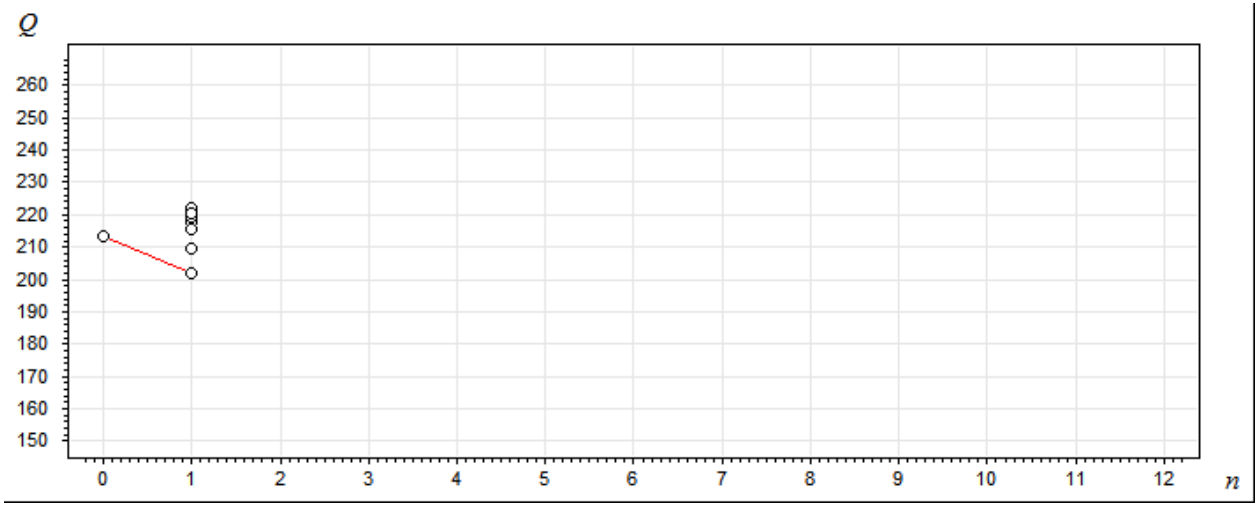


$j = 0$

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

1: $Q^* := Q(\varnothing)$; — инициализация;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: найти лучший набор сложности j:
$\mathscr{G}_j := \underset{\mathscr{G}:\ |\mathscr{G}|=j}{\arg\min}\, Q(\mathscr{G})$;
4: **если** $Q(\mathscr{G}_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j)$;
5: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}$;

Алгоритм полного перебора (Full Search)

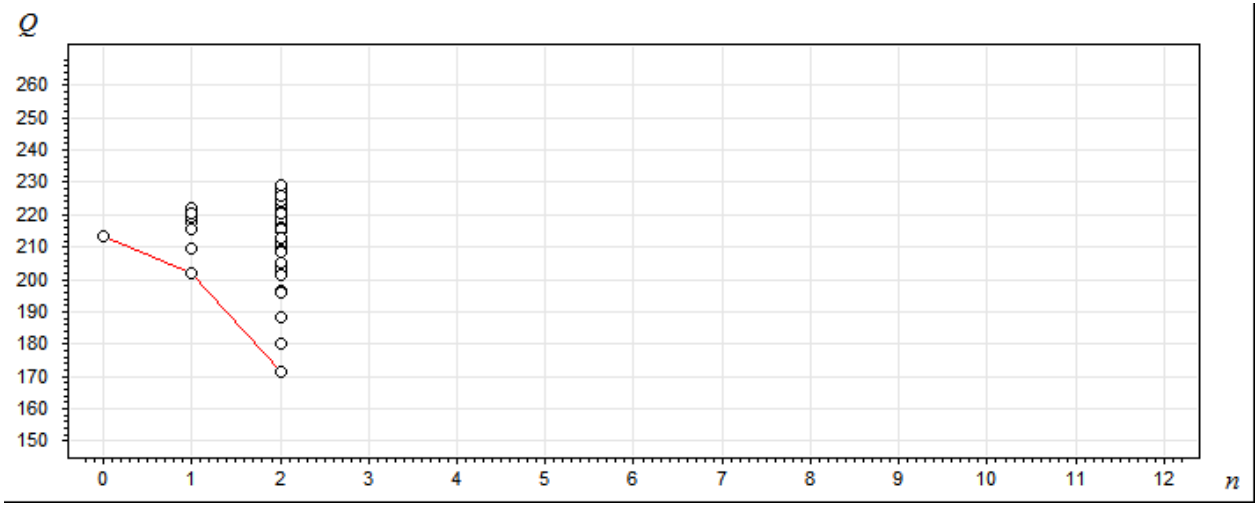


$j = 1$
$j^* = 1$

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

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

Алгоритм полного перебора (Full Search)

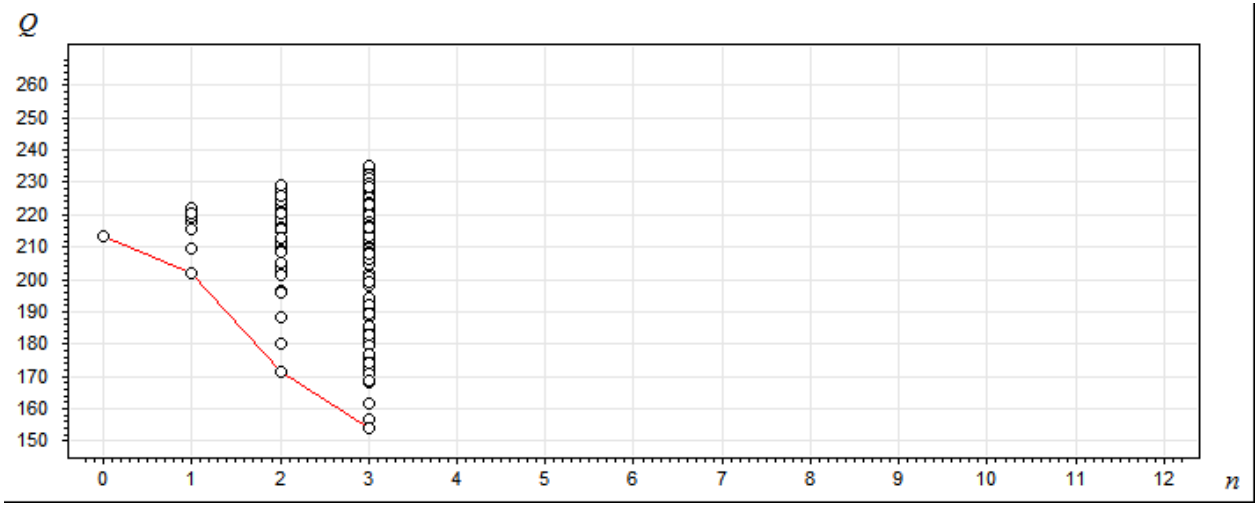


$j = 2$
$j^* = 2$

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

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

Алгоритм полного перебора (Full Search)

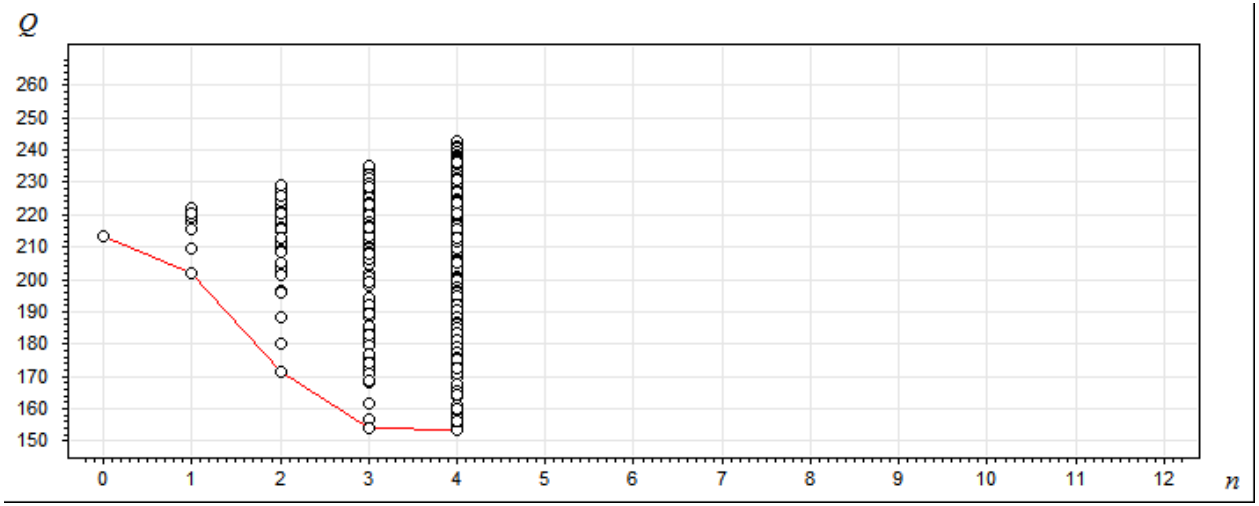


$j = 3$
$j^* = 3$

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

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

Алгоритм полного перебора (Full Search)

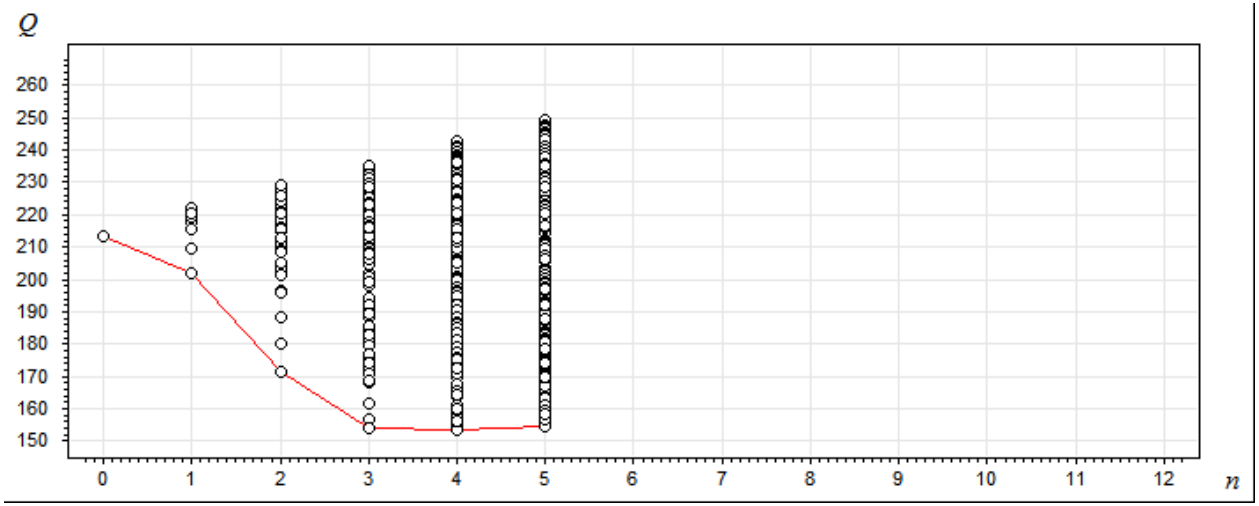


$j = 4$
$j^* = 4$

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

1: $Q^* := Q(\varnothing)$; — инициализация;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: найти лучший набор сложности j:
$\mathscr{G}_j := \underset{\mathscr{G}:\ |\mathscr{G}|=j}{\arg\min}\, Q(\mathscr{G})$;
4: **если** $Q(\mathscr{G}_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j)$;
5: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}$;

Алгоритм полного перебора (Full Search)

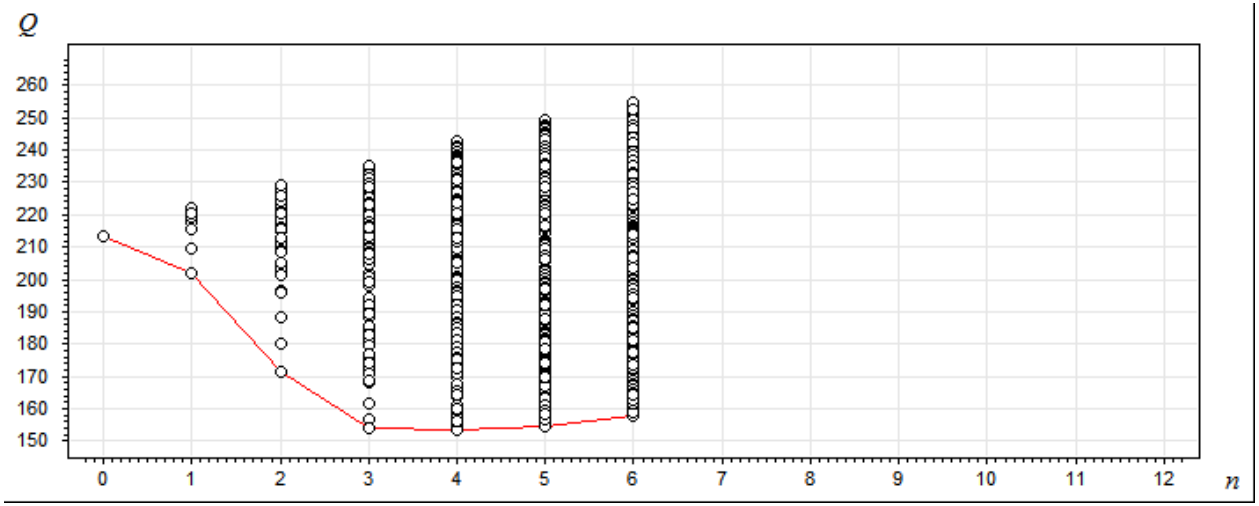


$j = 5$
$j^* = 4$

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

1: $Q^* := Q(\varnothing)$; — инициализация;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: найти лучший набор сложности j:
$\mathscr{G}_j := \underset{\mathscr{G}:\ |\mathscr{G}|=j}{\arg\min}\, Q(\mathscr{G})$;
4: **если** $Q(\mathscr{G}_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j)$;
5: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}$;

Алгоритм полного перебора (Full Search)

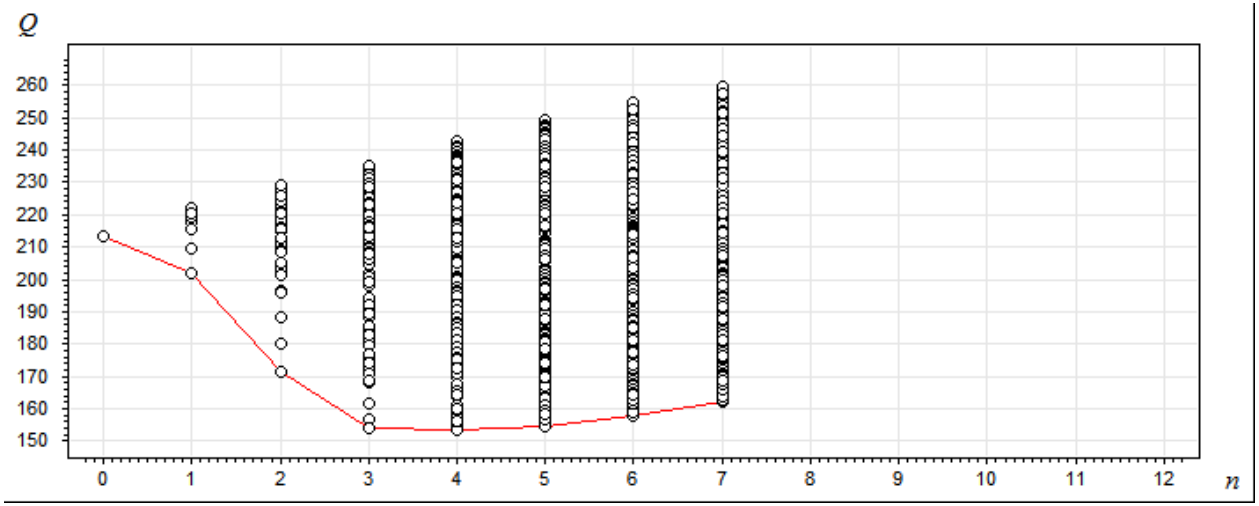


$j = 6$
$j^* = 4$

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

1: $Q^* := Q(\varnothing)$; — инициализация;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: найти лучший набор сложности j:
$\mathscr{G}_j := \underset{\mathscr{G}:\ |\mathscr{G}|=j}{\arg\min}\, Q(\mathscr{G})$;
4: **если** $Q(\mathscr{G}_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j)$;
5: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}$;

Алгоритм полного перебора (Full Search)



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

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

1: $Q^* := Q(\varnothing)$; — инициализация;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: найти лучший набор сложности j:
$\mathscr{G}_j := \underset{\mathscr{G}:\ |\mathscr{G}|=j}{\arg\min}\, Q(\mathscr{G})$;
4: **если** $Q(\mathscr{G}_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j)$;
5: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}$;

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 ооооооочень 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 $\mathscr{F}$, критерий Q, параметр d;

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

Алгоритм жадного добавления (Add)

Преимущества:

- работает быстро — $O(n^2)$, точнее $O(n(j^* + d))$;
- возможны быстрые инкрементные алгоритмы, пример — *шаговая регрессия* (step-wise regression).

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: $\mathscr{G}_0 := \varnothing$; $Q^* := Q(\varnothing)$; $t := 0$; — инициализация;
2: **повторять**

3: **пока** $|\mathscr{G}_t| < n$ добавлять признаки (Add):
4: $t := t + 1$; — началась следующая итерация;
5: $f^* := \underset{f \in \mathscr{F} \setminus \mathscr{G}_{t-1}}{\arg\min} Q\big(\mathscr{G}_{t-1} \cup \{f\}\big)$; $\mathscr{G}_t := \mathscr{G}_{t-1} \cup \{f^*\}$;
6: **если** $Q(\mathscr{G}_t) < Q^*$ **то** $t^* := t$; $Q^* := Q(\mathscr{G}_t)$;
7: **если** $t - t^* \geqslant d$ **то прервать цикл**;

8: **пока** $|\mathscr{G}_t| > 0$ удалять признаки (Del):
9: $t := t + 1$; — началась следующая итерация;
10: $f^* := \underset{f \in \mathscr{G}_{t-1}}{\arg\min} Q\big(\mathscr{G}_{t-1} \setminus \{f\}\big)$; $\mathscr{G}_t := \mathscr{G}_{t-1} \setminus \{f^*\}$;
11: **если** $Q(\mathscr{G}_t) < Q^*$ **то** $t^* := t$; $Q^* := Q(\mathscr{G}_t)$;
12: **если** $t - t^* \geqslant d$ **то прервать цикл**;

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

Алгоритм поочерёдного добавления и удаления (Add-Del)

Преимущества:

- как правило, лучше, чем Add и Del по отдельности;
- возможны быстрые инкрементные алгоритмы,

Недостатки:

- работает дольше, оптимальность не гарантирует.

Поиск в ширину (BFS)

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

Философский принцип *неокончательных 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*:

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

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

Поиск в ширину (BFS)

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

1: первый ряд состоит из всех наборов длины 1:
$R_1 := \{\{f_1\}, \ldots, \{f_n\}\}$; $Q^* = Q(\varnothing)$;
2: **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3: отсортировать ряд $R_j = \{\mathscr{G}_j^1, \ldots, \mathscr{G}_j^{B_j}\}$
по возрастанию критерия: $Q(\mathscr{G}_j^1) \leqslant \ldots \leqslant Q(\mathscr{G}_j^{B_j})$;
4: **если** $B_j > B$ **то**
5: $R_j := \{\mathscr{G}_j^1, \ldots, \mathscr{G}_j^B\}$; — B лучших наборов ряда;
6: **если** $Q(\mathscr{G}_j^1) < Q^*$ **то** $j^* := j$; $Q^* := Q(\mathscr{G}_j^1)$;
7: **если** $j - j^* \geqslant d$ **то вернуть** $\mathscr{G}_{j^*}^1$;
8: породить следующий ряд:
$R_{j+1} := \{\mathscr{G} \cup \{f\} \mid \mathscr{G} \in R_j,\ f \in \mathscr{F} \setminus \mathscr{G}\}$;

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 $I_j(f)$:

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

Резюме в конце лекции

- Для отбора признаков могут использоваться любые эвристические методы дискретной оптимизации

$$Q(\mathscr{G}) \to \min_{\mathscr{G} \subseteq \mathscr{F}}.$$

- Большинство эвристик эксплуатируют две основные идеи:
  — не все признаки информативны;
  — $Q(\mathscr{G})$ изменяется не сильно при небольшом изменении $\mathscr{G}$.