

Методы машинного обучения. Отбор признаков (feature selection)

Воронцов Константин Вячеславович
www.MachineLearning.ru/wiki?title=User:Vokov
вопросы к лектору: voron@forecsys.ru

материалы курса:
github.com/MSU-ML-COURSE/ML-COURSE-22-23
орг.вопросы по курсу: ml.cmc@mail.ru

1 Задача отбора признаков

- Постановка задачи отбора признаков
- Критерии отбора признаков
- Подходы к отбору признаков

2 Беспереборные методы

- Критерии фильтрации признаков
- Отбор признаков с учителем и без учителя
- Встроенные методы

3 Методы комбинаторной оптимизации

- Полный перебор
- Жадные и полужадные алгоритмы
- Стохастические алгоритмы

Постановка задачи отбора признаков

$X^\ell = (x_i, y_i)_{i=1}^\ell$ — обучающая выборка, $y_i = y(x_i)$;
 $f_j(x)$, $j = 1, \dots, n$ — признаки объекта x

Почему полезно отбирать признаки:

- признак может быть неинформативным:
 $f_j(x)$ не содержит информации об $y(x)$
- признак может быть зависимым:
информация о $f_j(x)$ содержится в других признаках
- признак может быть слишком сильно зашумлённым,
и тогда его использование может повышать риск ошибки
- минимизация числа используемых признаков может
давать экономию ресурсов времени и памяти
- понижение размерности n может уменьшать переобучение

Напоминание. Разновидности внешних критериев

Эмпирические критерии:

- Проверка на отложенных данных (hold-out validation)
- Кросс-проверка (cross-validation, CV)
- Скользящий контроль (leave one out, LOO)
- Поблочная кросс-проверка (q -fold CV, $t \times q$ -fold CV)
- Непротиворечивость или согласованность моделей
- Устойчивость модели при малых изменениях данных

Аналитические критерии (регуляризаторы):

- L_0 -, L_1 -, L_2 -регуляризации линейных моделей
- Информационные критерии (AIC, BIC, CIC и др.)
- Сложностные оценки (Вапника-Червоненкиса и др.)

Задача отбора признаков в логических закономерностях

Закономерность R — конъюнкция пороговых условий:

$$R(x) = \bigwedge_{j \in J} [f_j(x) \geq a_j].$$

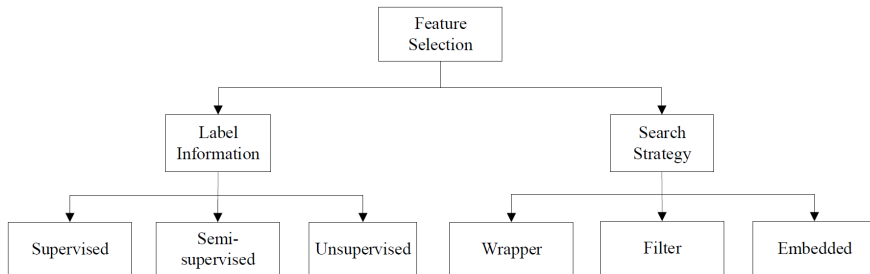
Критерий информативности относительно класса $c \in Y$:

$$I(p, n) \rightarrow \max_{J, \{a_j\}}; \quad \begin{cases} p(R) = \#\{x_i: R(x_i)=1 \text{ и } y_i=c\} \rightarrow \max \\ n(R) = \#\{x_i: R(x_i)=1 \text{ и } y_i \neq c\} \rightarrow \min \end{cases}$$

Информативность $I(p, n)$ имеет оптимум по сложности $|J|$, аналогично внешним критериям:

- слишком мало признаков \Rightarrow большие n , низкая $I(p, n)$
- оптимально признаков \Rightarrow малые n , большие p , высокая $I(p, n)$
- слишком много признаков \Rightarrow малые $p + n$, низкая $I(p, n)$

Обзор подходов к отбору признаков



- Без учителя — например, по корреляциям признаков
- Обёртки — использование готовых методов обучения
- Фильтры — оценивание признаков по отдельности
- Встроенные методы, в некоторых моделях (LASSO, ElasticNet)

Сколько информации об y содержится в f_j

$f_j: X \rightarrow V_j$ — дискретный признак, V_j — конечное множество
 $p(v)$ — частотная оценка вероятности $P(f_j(x) = v)$

Энтропия, мера неопределённости значений y :

$$H(y) = - \sum_{y \in Y} p(y) \log p(y)$$

Условная энтропия, неопределённость y при известном f_j :

$$H(y|f_j) = - \sum_{v \in V_j} p(v) \sum_{y \in Y} p(y|v) \log p(y|v)$$

Взаимная информация (mutual information, information gain):

$$I(y, f_j) = H(y) - H(y|f_j) = \sum_{v \in V_j} \sum_{y \in Y} p(y, v) \log \frac{p(y, v)}{p(y)p(v)}$$

$I(y, f_j) = 0 \Leftrightarrow$ переменные y и f_j независимы

Отбор признаков с учителем и без учителя

Отбор признаков с учителем:

- вычисление полезности признаков: MI или корреляции (y, f_j)
- ранжирование признаков по их полезности
- отбор top- k полезных признаков по выбранному критерию
- **плюс:** очень быстро
- **минус:** не учитываются зависимости между признаками

Отбор признаков без учителя:

- вычисление парных корреляций признаков (f_j, f_s)
- выделение кластеров или связанных подграфов
- выбор представителей от каждого кластера
(метод корреляционных плеяд П.В.Терентьева, 1928)
- **минус:** учитываются только парные зависимости

Отбор признаков в линейных моделях

Негладкие регуляризаторы с параметром селективности μ :

$$\sum_{i=1}^{\ell} \mathcal{L}(\langle x_i, w \rangle, y_i) + \tau \sum_{j=1}^n R_{\mu}(w_j) \rightarrow \min_w.$$

LASSO (L_1): $R_{\mu}(w) = \mu|w|$

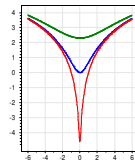
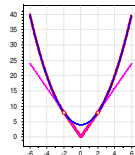
Elastic Net: $R_{\mu}(w) = \mu|w| + w^2$

Support Features Machine (SFM):

$$R_{\mu}(w) = \begin{cases} 2\mu|w|, & |w| \leq \mu; \\ \mu^2 + w^2, & |w| \geq \mu; \end{cases}$$

Relevance Features Machine (RFM):

$$R_{\mu}(w) = \ln(\mu w^2 + 1)$$



Задача отбора признаков по внешнему критерию

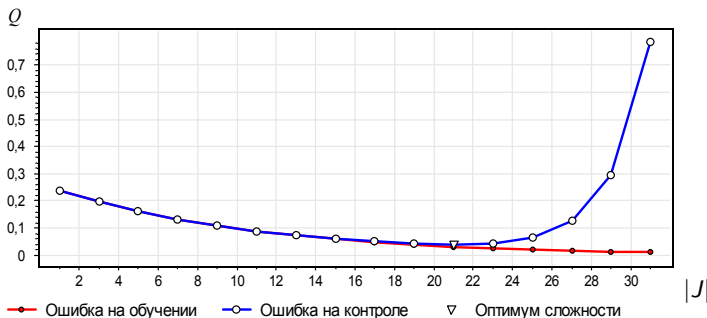
$F = \{f_j: X \rightarrow D_j: j = 1, \dots, n\}$ — множество признаков;

μ_J — метод обучения, использующий только признаки $J \subseteq F$;

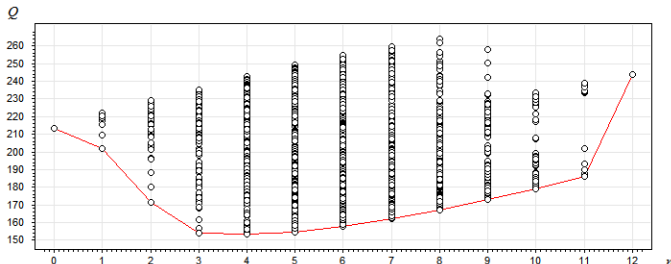
$Q(J) = Q(\mu_J, X^\ell)$ — выбранный внешний критерий.

$Q(J) \rightarrow \min$ — задача дискретной оптимизации.

Внутренний критерий и внешний критерий:



Алгоритм полного перебора (Full Search)



Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

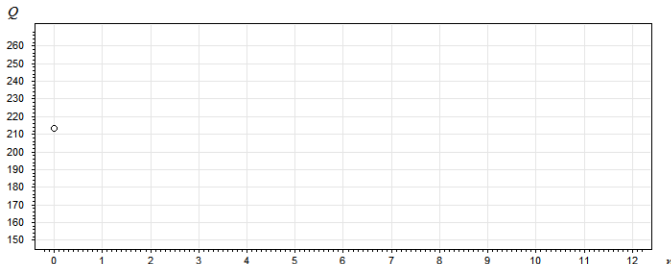
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$d = 3$$

$$j = 0$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

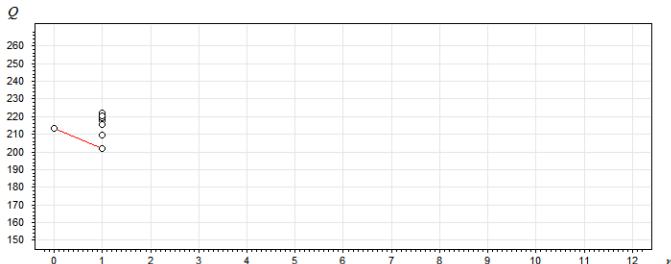
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned}d &= 3 \\j &= 1 \\j^* &= 1\end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

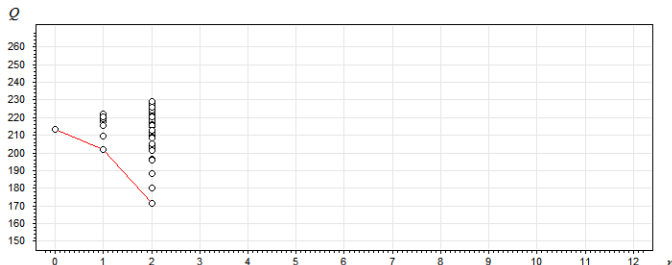
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$d = 3$$

$$j = 2$$

$$j^* = 2$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

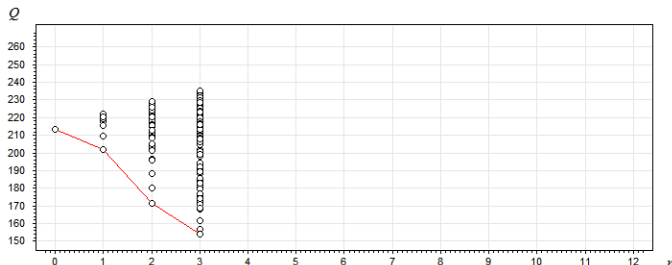
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned} d &= 3 \\ j &= 3 \\ j^* &= 3 \end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

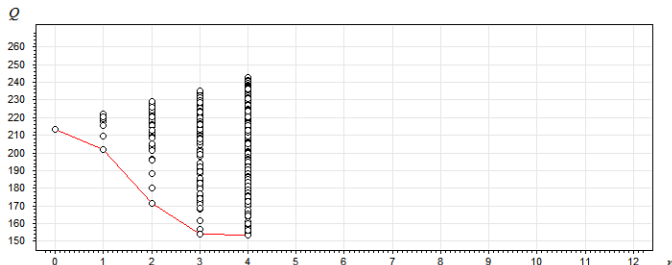
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned}d &= 3 \\j &= 4 \\j^* &= 4\end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

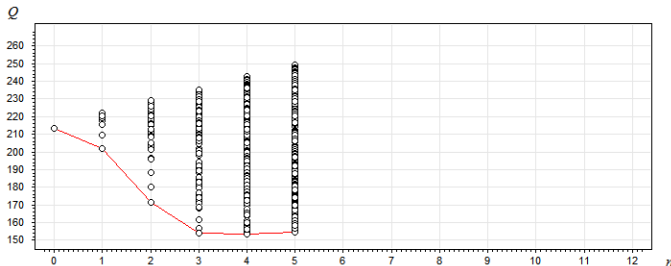
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned} d &= 3 \\ j &= 5 \\ j^* &= 4 \end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

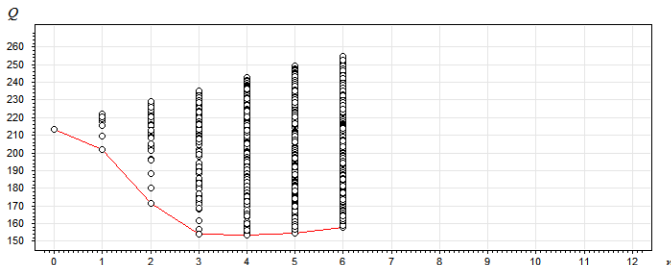
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned}d &= 3 \\j &= 6 \\j^* &= 4\end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

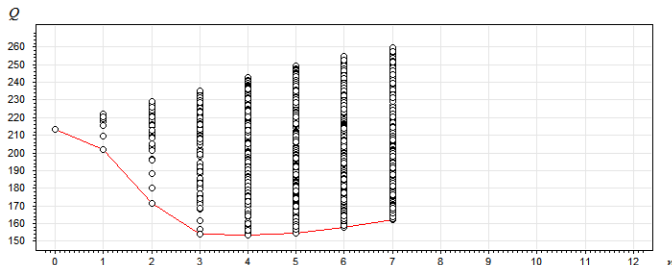
для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$$\begin{aligned} d &= 3 \\ j &= 7 \\ j^* &= 4 \end{aligned}$$

Вход: множество F , критерий Q , параметр d ;

инициализация: $Q^* := Q(\emptyset)$;

для $j = 1, \dots, n$, где j — сложность наборов

$J_j := \arg \min_{J: |J|=j} Q(J)$ — найти лучший набор сложности j ;

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{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 F , критерий Q , параметр d ;

инициализация: $J_0 := \emptyset$; $Q^* := Q(\emptyset)$;

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

 найти признак, наиболее выгодный для добавления:

$$f^* := \arg \min_{f \in F \setminus J_{j-1}} Q(J_{j-1} \cup \{f\});$$

 добавить этот признак в набор:

$$J_j := J_{j-1} \cup \{f^*\};$$

если $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;

если $j - j^* \geq d$ **то вернуть** J_{j^*} ;

Преимущество: скорость $O(n^2)$, точнее $O(nj^*)$, вместо $O(2^n)$

Недостаток: склонность включать в набор лишние признаки

Способы устранения: Del, Add-Del, Beam Search

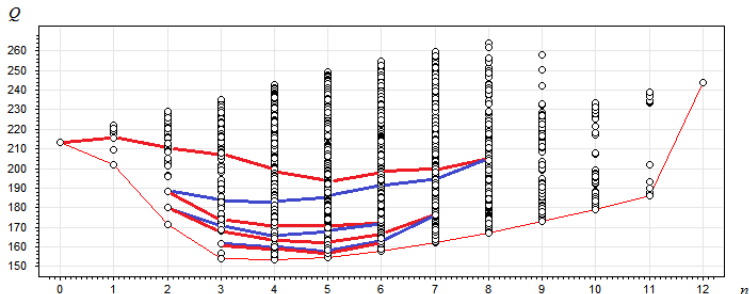
Алгоритм поочерёдного добавления и удаления (Add-Del)

Преимущества:

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

Недостатки:

- работает дольше, оптимальность не гарантирует.



Алгоритм поочерёдного добавления и удаления (Add-Del)

инициализация: $J_0 := \emptyset$; $Q^* := Q(\emptyset)$; $t := 0$;

повторять

пока $|J_t| < n$ добавлять признаки (итерации Add):

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

$f^* := \arg \min_{f \in F \setminus J_{t-1}} Q(J_{t-1} \cup \{f\})$; $J_t := J_{t-1} \cup \{f^*\}$;

если $Q(J_t) < Q^*$ **то** $t^* := t$; $Q^* := Q(J_t)$;

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

пока $|J_t| > 0$ удалять признаки (итерации Del):

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

$f^* := \arg \min_{f \in J_{t-1}} Q(J_{t-1} \setminus \{f\})$; $J_t := J_{t-1} \setminus \{f^*\}$;

если $Q(J_t) < Q^*$ **то** $t^* := t$; $Q^* := Q(J_t)$;

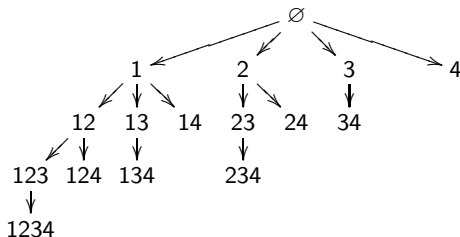
если $t - t^* \geq d$ **то прервать цикл**;

пока значения критерия $Q(J_{t^*})$ уменьшаются;

вернуть J_{t^*} ;

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 J бесперспективен, то больше не пытаться его наращивать.

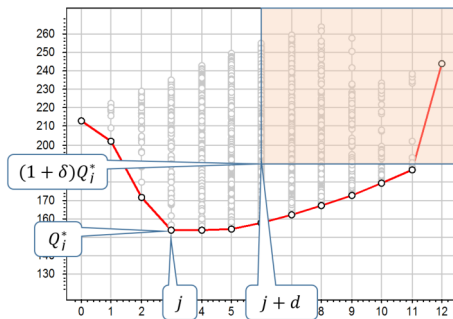
Поиск в глубину (DFS, метод ветвей и границ)

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

Оценка перспективности:
набор J не наращивается,
если найдётся j такой, что

$$\begin{cases} Q(J) \geq (1 + \delta)Q_j^*; \\ |J| \geq j + d; \end{cases}$$

$d \geq 0$ и $\delta \geq 0$ — параметры.



Чем меньше d и δ , тем сильнее сокращается перебор.

Поиск в глубину (DFS, метод ветвей и границ)

Вход: множество F , критерий Q , параметры d и δ ;

процедура *нарастить* ($J \subseteq F$)

если найдётся $j: j \leq |J| - d$ и $Q(J) \geq (1 + \delta)Q_j^*$, то
 набор J бесперспективный; **выход**;
 $Q_{|J|}^* := \min\{Q_{|J|}^*, Q(J)\}$;
 для всех $f_s \in F$ таких, что $s > \max\{t \mid f_t \in J\}$:
 нарастить ($J \cup \{f_s\}$);

инициализировать массив лучших значений критерия:

$Q_j^* := Q(\emptyset)$ для всех $j = 1, \dots, n$;

упорядочить признаки по убыванию информативности;

нарастить (\emptyset);

вернуть J , для которого $Q(J) = \min_{j=1, \dots, n} Q_j^*$;

Усечённый поиск в ширину (Beam Search)

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

Философия — 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*:

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

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

Ивахненко А. Г., Юрачковский Ю. П. Моделирование сложных систем
по экспериментальным данным, 1987.

Усечённый поиск в ширину (Beam Search)

Вход: множество F , критерий Q , параметры d, B ;

первый ряд состоит из всех наборов длины 1:

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

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

отсортировать ряд $R_j = \{J_j^1, \dots, J_j^{B_j}\}$

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

если $B_j > B$ **то**

$\lfloor R_j := \{J_j^1, \dots, J_j^B\}$ — оставить B лучших наборов ряда;

если $Q(J_j^1) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j^1)$;

если $j - j^* \geq d$ **то вернуть** $J_{j^*}^1$;

породить следующий ряд:

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

Усечённый поиск в ширину: дополнительные эвристики

- **Трудоёмкость:**
 $O(Bn^2)$, точнее $O(Bn(j^* + d))$.
- **Проблема дубликатов:**
после сортировки $Q(J_j^1) \leq \dots \leq Q(J_j^{B_j})$ проверить на совпадение только соседние наборы с равными значениями внутреннего и внешнего критерия.
- **Адаптивный отбор признаков:**
на последнем шаге добавлять к j -му ряду только признаки f с наибольшей информативностью $I_j(f)$:

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

Эволюционный алгоритм поиска (идея и терминология)

$J \subseteq F$ — индивид (в МГУА «модель»);

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

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

Бинарная операция скрещивания (crossover) $\beta = \beta' \times \beta''$:

- вариант 1: $\beta_j = \rho_j \beta'_j + (1 - \rho_j) \beta''_j$, $\rho_j \sim \text{uni}(0, 1)$
- вариант 2: $\beta = (\beta'_1, \dots, \beta'_s, \beta''_{s+1}, \dots, \beta''_n)$, $s \sim \text{uni}(1, \dots, n)$,
надо задавать «естественное» ранжирование признаков

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

- $\beta_j = \rho_j (1 - \beta'_j) + (1 - \rho_j) \beta'_j$, $\rho_j \sim \text{bin}(p_m)$,
где p_m — параметр вероятности мутации.

Эволюционный (генетический) алгоритм

Вход: множество F , критерий Q , параметры: d, p_m ,
 B — размер популяции, T — число поколений;

инициализировать случайную популяцию из B наборов:

$B_1 := B$; $R_1 := \{J_1^1, \dots, J_1^{B_1}\}$; $Q^* := Q(\emptyset)$;

для $t = 1, \dots, T$, где t — номер поколения:

ранжирование индивидов: $Q(J_t^1) \leq \dots \leq Q(J_t^{B_t})$;

если $B_t > B$ **то** селекция: $R_t := \{J_t^1, \dots, J_t^B\}$;

если $Q(J_t^1) < Q^*$ **то** $t^* := t$; $Q^* := Q(J_t^1)$;

если $t - t^* \geq d$ **то вернуть** $J_{t^*}^1$;

породить $t+1$ -е поколение путём скрещиваний и мутаций:

$R_{t+1} := \{\sim(J' \times J'') \mid J', J'' \in R_t\} \cup R_t$;

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).

Обобщение: случайный поиск с адаптацией (СПА)

Вход: множество F , критерий Q , параметры: d ,
 B — размер популяции, T — число поколений;

равные вероятности признаков: $p_1 = \dots = p_n := 1/n$;

инициализировать случайную популяцию из B_1 наборов:

$R_1 := \{J_1^1, \dots, J_1^{B_1} \sim \{p_1, \dots, p_n\}\}$; $Q^* := Q(\emptyset)$;

для $t = 1, \dots, T$, где t — номер поколения:

ранжирование индивидов: $Q(J_t^1) \leq \dots \leq Q(J_t^{B_t})$;

если $B_t > B$ **то** селекция: $R_t := \{J_t^1, \dots, J_t^B\}$;

если $Q(J_t^1) < Q^*$ **то** $t^* := t$; $Q^* := Q(J_t^1)$;

если $t - t^* \geq d$ **то вернуть** $J_{t^*}^1$;

увеличить p_j для признаков из лучших наборов;

уменьшить p_j для признаков из худших наборов;

породить $t+1$ -е поколение из B_t наборов:

$R_{t+1} := \{J_{t+1}^1, \dots, J_{t+1}^{B_t} \sim \{p_1, \dots, p_n\}\} \cup R_t$;

Попытка обоснования. Теорема схемы

Схема — вектор $H = (h_1, \dots, h_n)$, где $h_j \in \{0, 1, *\}$

$o(H)$ — порядок схемы, число не* в схеме

$d(H)$ — длина схемы, расстояние между первым и последним

не*, число мест, в которых кроссовер может нарушить схему

$f(H, t)$ — степень приспособленности схемы, среднее Q по всем векторам, подходящим под схему в поколении t

$\bar{f}(t) = f(*^n, t)$ — средняя приспособленность популяции

p_c — вероятность кроссовера (только второй вариант)

p_m — вероятность мутации

Теорема схемы [Холланд, 1975]

Число индивидов схемы H в популяции поколения t :

$$Em(H, t + 1) \geq m(H, t) \frac{f(H, t)}{\bar{f}(t)} \left(1 - p_c \frac{d(H)}{n - 1} - p_m o(H) \right)$$

Интерпретация теоремы схемы

Число индивидов схемы H в популяции поколения t :

$$Em(H, t + 1) \geq m(H, t) \frac{f(H, t)}{\bar{f}(t)} \left(1 - p_c \frac{d(H)}{n - 1} - p_m o(H) \right)$$

- **Строительный блок** — схема H с низким порядком $o(H)$, короткой длиной $d(H)$, высокой приспособленностью $f(H, t)$
- Если приспособленность строительного блока выше средней в популяции, то число его индивидов будет расти экспоненциально в последующих популяциях
- **Гипотеза (building block hypothesis)**: «строительные блоки объединяются, чтобы сформировать ещё лучшие блоки»

John Henry Holland. Adaptation in natural and artificial systems. 1992

David White. An overview of schema theory. 2014

- Для отбора признаков могут использоваться любые эвристические методы дискретной оптимизации

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

- $Q(J)$ должен быть внешним критерием, с характерным минимумом по сложности модели
- Большинство эвристик эксплуатируют две основные идеи:
 - признаки ранжируются по их полезности;
 - $Q(J)$ изменяется не сильно при малом изменении J .
- МГУА, ЭА и СПА очень похожи — на их основе можно изобретать новые «симбиотические» мета-эвристики.