МАШИННОЕ ОБУЧЕНИЕ
методы понижения размерности

Воронцов Константин Вячеславович
ФУПМ МФТИ • ВМК МГУ • Яндекс • FORECSYS

9 июля 2016
Сочи, Сириус • Проектная смена • 1–24 июля 2016

Содержание

Анализ ошибок классификации

Задача классификации на два класса, $y_i \in \{-1, +1\}$.
Алгоритм классификации $a(x_i) \in \{-1, +1\}$

число объектов	ответ классификатора	правильный ответ
TP, True Positive	$a(x_i) = +1$	$y_i = +1$
TN, True Negative	$a(x_i) = -1$	$y_i = -1$
FP, False Positive	$a(x_i) = +1$	$y_i = -1$
FN, False Negative	$a(x_i) = -1$	$y_i = +1$

Доля правильных классификаций (чем больше, тем лучше):

$$\text{Accuracy} = \frac{1}{\ell}\sum_{i=1}^{\ell}\big[a(x_i) = y_i\big] = \frac{\text{TP} + \text{TN}}{\text{FP} + \text{FN} + \text{TP} + \text{TN}}$$

Недостаток: не учитывается ни численность (дисбаланс) классов, ни цена ошибки на объектах разных классов.

Оценки качества двухклассовой классификации

В информационном поиске:

$$\text{Точность, Precision} = \frac{TP}{TP+FP}$$

$$\text{Полнота, Recall} = \frac{TP}{TP+FN}$$

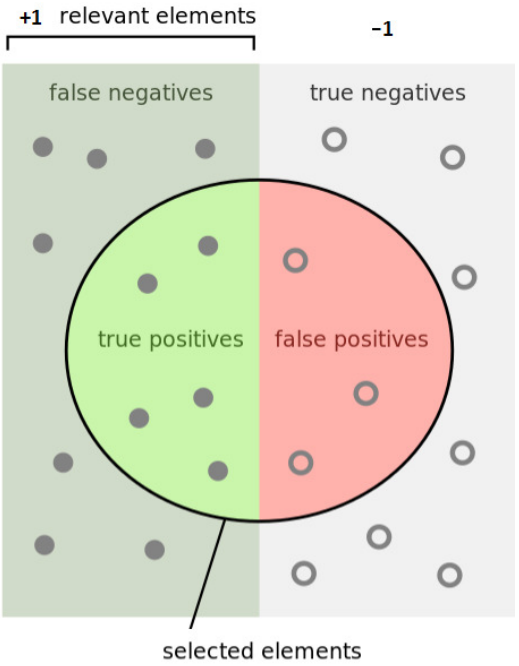
Precision — доля релевантных среди найденных
Recall — доля найденных среди релевантных

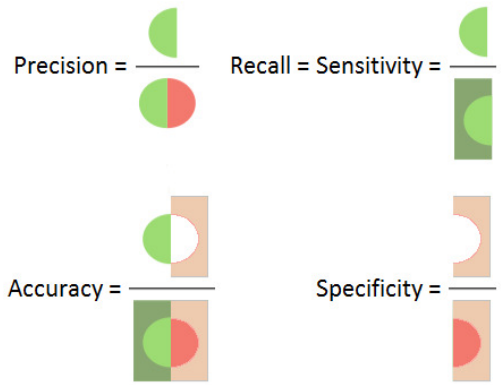
В медицинской диагностике:

$$\text{Чувствительность, Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{Специфичность, Specificity} = \frac{TN}{TN+FP}$$

Sensitivity — доля верных положительных диагнозов
Specificity — доля верных отрицательных диагнозов





Точность и полнота многоклассовой классификации

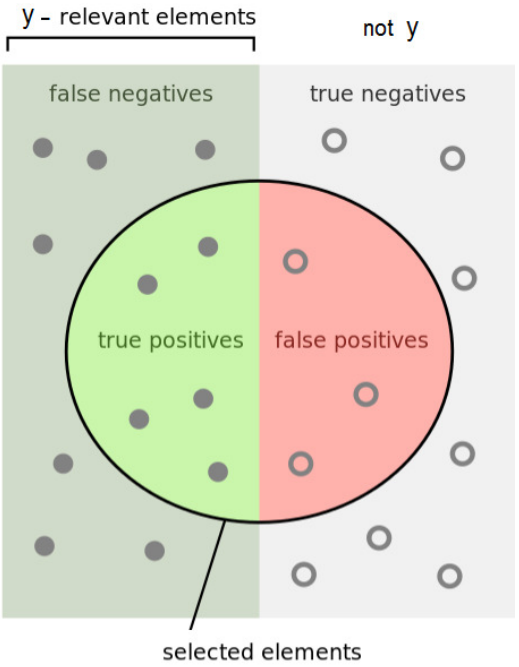
Для каждого класса $y \in Y$:
TP_y — верные положительные
FP_y — ложные положительные
FN_y — ложные отрицательные

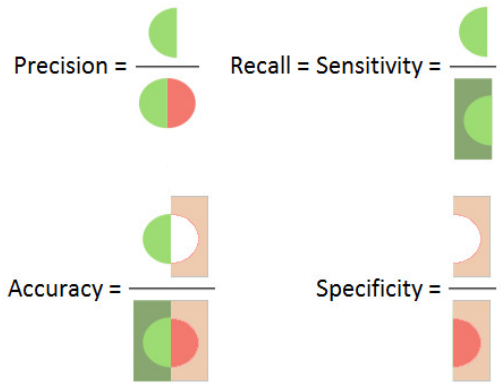
Точность и полнота с микроусреднением:

$$\text{Precision: } P = \frac{\sum_y \mathrm{TP}_y}{\sum_y (\mathrm{TP}_y + \mathrm{FP}_y)};$$

$$\text{Recall: } R = \frac{\sum_y \mathrm{TP}_y}{\sum_y (\mathrm{TP}_y + \mathrm{FN}_y)};$$

Микроусреднение не чувствительно
к ошибкам на малочисленных классах





Точность и полнота многоклассовой классификации

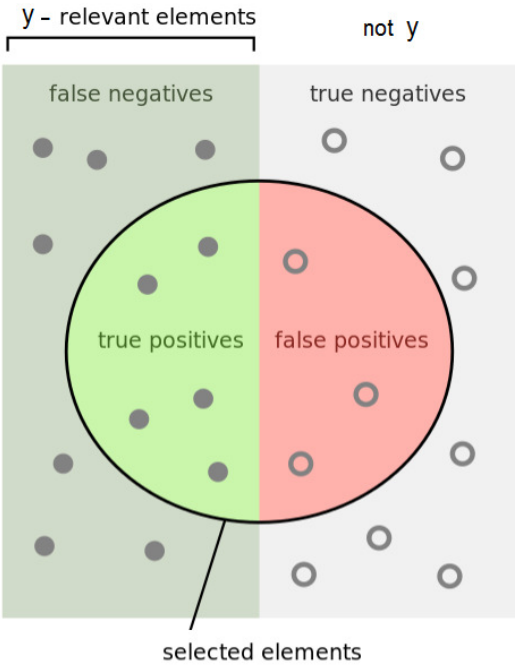
Для каждого класса $y \in Y$:
TP_y — верные положительные
FP_y — ложные положительные
FN_y — ложные отрицательные

Точность и полнота с макроусреднением:

$$\text{Precision: } P = \frac{1}{|C|}\sum_y \frac{\mathrm{TP}_y}{\mathrm{TP}_y + \mathrm{FP}_y};$$

$$\text{Recall: } R = \frac{1}{|C|}\sum_y \frac{\mathrm{TP}_y}{\mathrm{TP}_y + \mathrm{FN}_y};$$

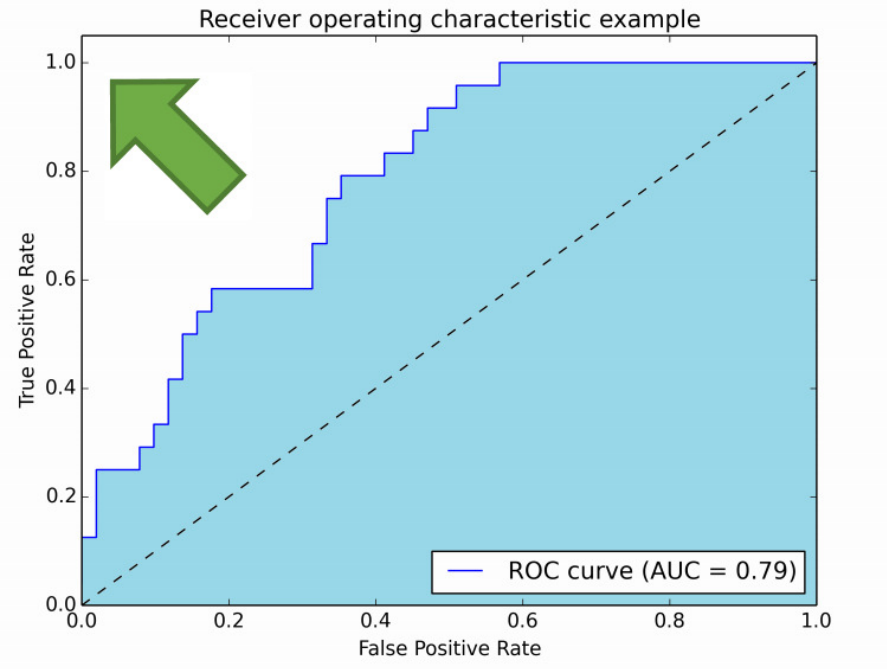
Макроусреднение чувствительно
к ошибкам на малочисленных классах

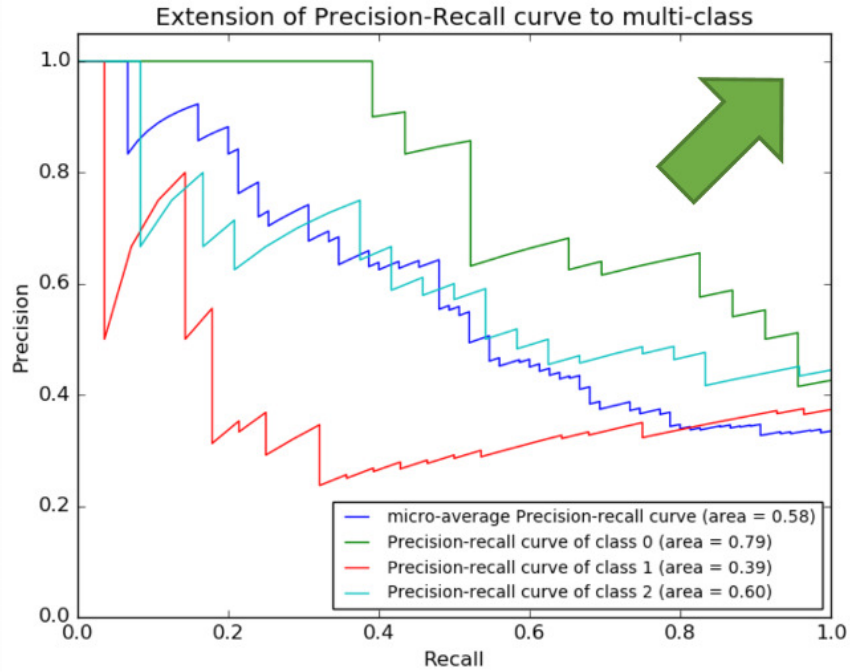


Кривые ROC и Precision-Recall

Модель классификации: $a(x) = \text{sign}\big(\langle x, w\rangle - w_0\big)$

Каждая точка кривой соответствует значению порога w_0





AUROC — площадь под ROC-кривой

AUPRC — площадь под кривой Precision-Recall

Примеры из Python scikit learn: `http://scikit-learn.org/dev`

Оценивание обобщающей (предсказательной) способности

Метод обучения μ строит алгоритм $a = \mu(X^\ell)$ по выборке X^ℓ.
$Q(a, X^\ell)$ — *критерий качества* a на X^ℓ.

Внутренний критерий оценивает качество на обучении X^ℓ:

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

Недостаток: эта оценка оптимистично смещена.

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

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

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

Эмпирические оценки обобщающей способности

Метод обучения μ строит алгоритм $a = \mu(X^\ell)$ по выборке X^ℓ.

- Кросс-проверка (cross-validation) по N разбиениям, $X^L = X^\ell_n \sqcup X^k_n, \quad L = \ell + k$:

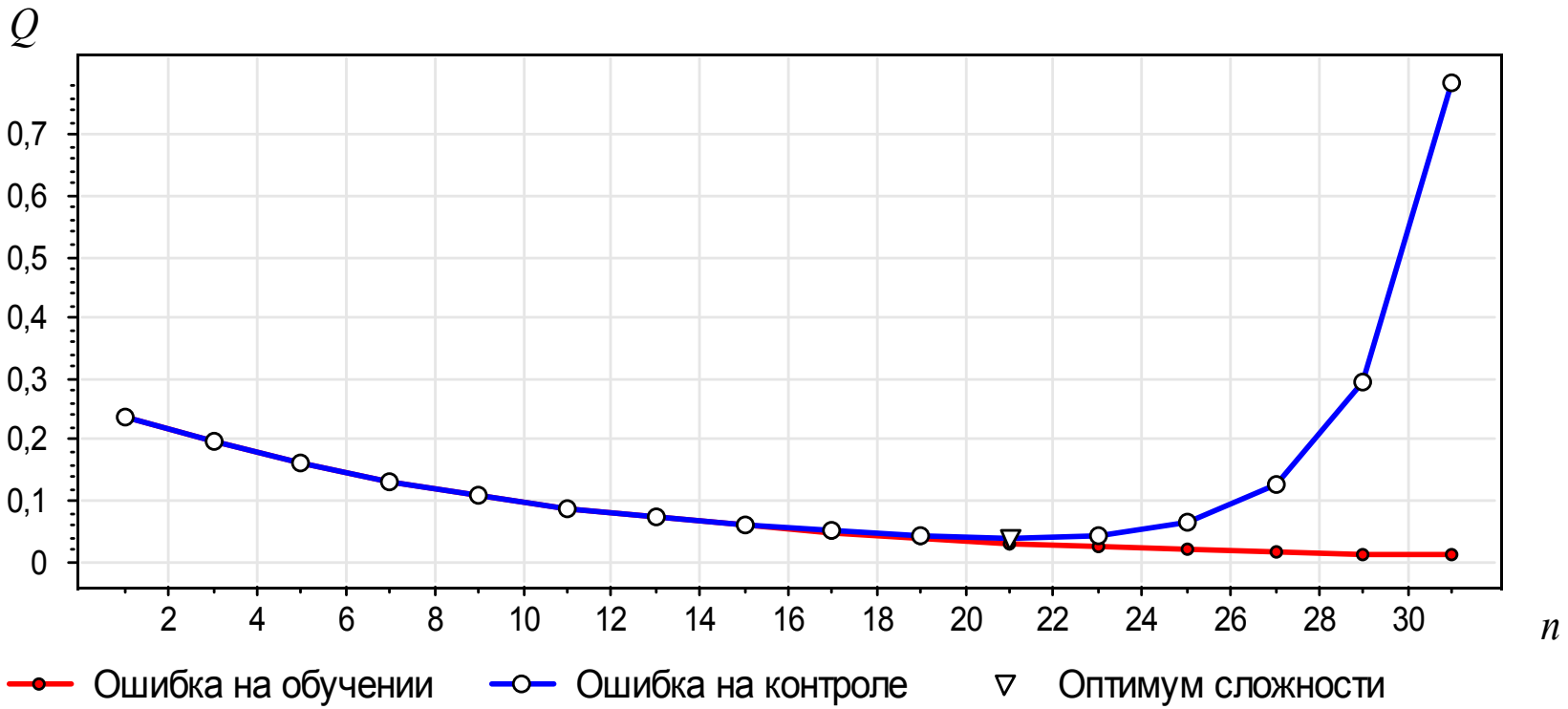
$$\mathrm{CV}(\mu, X^L) = \frac{1}{N} \sum_{n=1}^{N} Q(\mu(X^\ell_n), X^k_n) \to \text{opt}$$

- Скользящий контроль (leave-one-out): $k = 1$, $N = L$
- **Стратификация классов:**
  каждый класс X^L_y разбивается на блоки пропорциональных объёмов $X^L_y = X^{\ell_y}_{yn} \sqcup X^{k_y}_{yn}, \quad \ell_y : k_y = \ell : k$

Основное отличие внешних критериев от внутренних

Внутренний критерий монотонно убывает с ростом сложности модели (например, числа признаков).

Внешний критерий имеет характерный минимум, соответствующий оптимальной сложности модели.



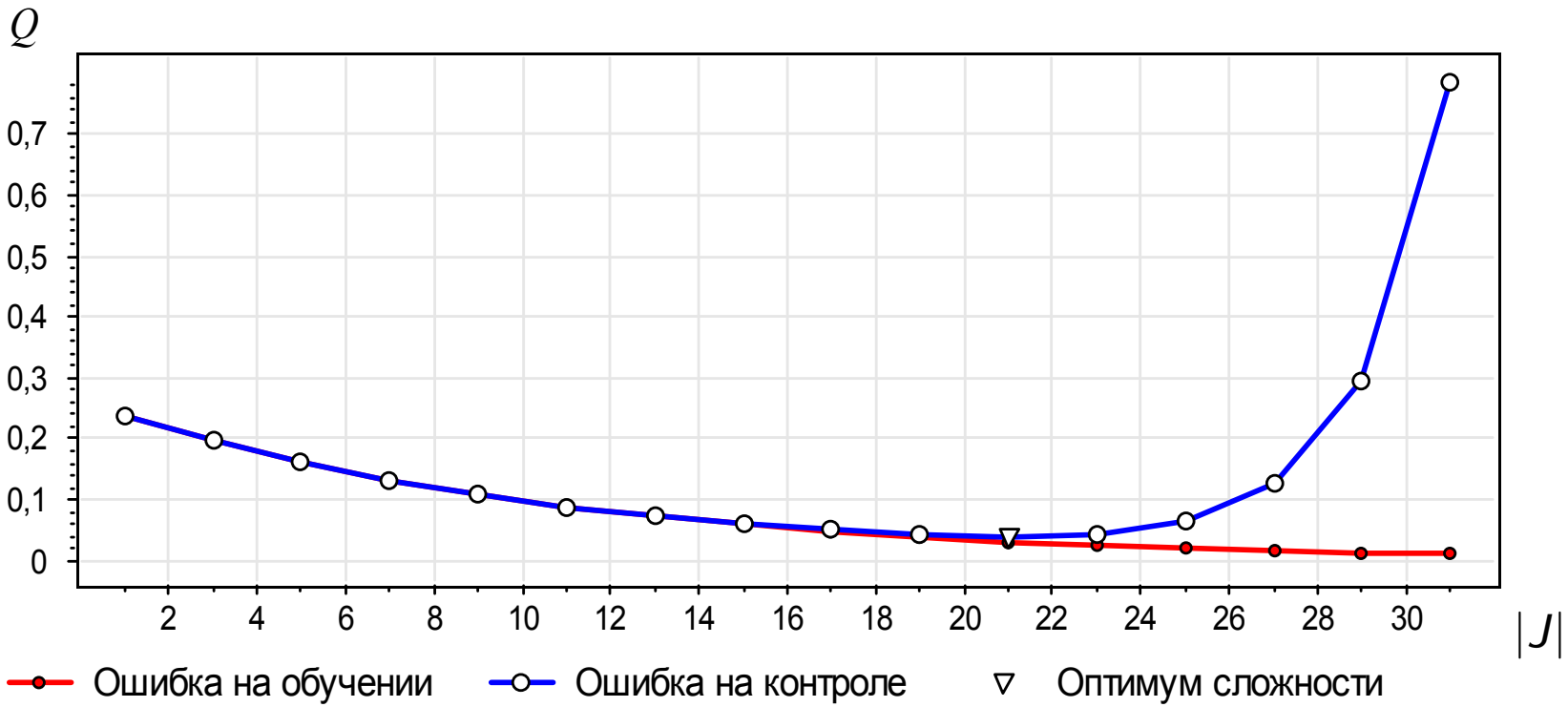
Задача отбора признаков по внешнему критерию

$F = \{f_j(x)\colon j = 1, \ldots, n\}$ — множество признаков;

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

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

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



Сортировка с отсечением

Фольклорный метод отбора признаков:

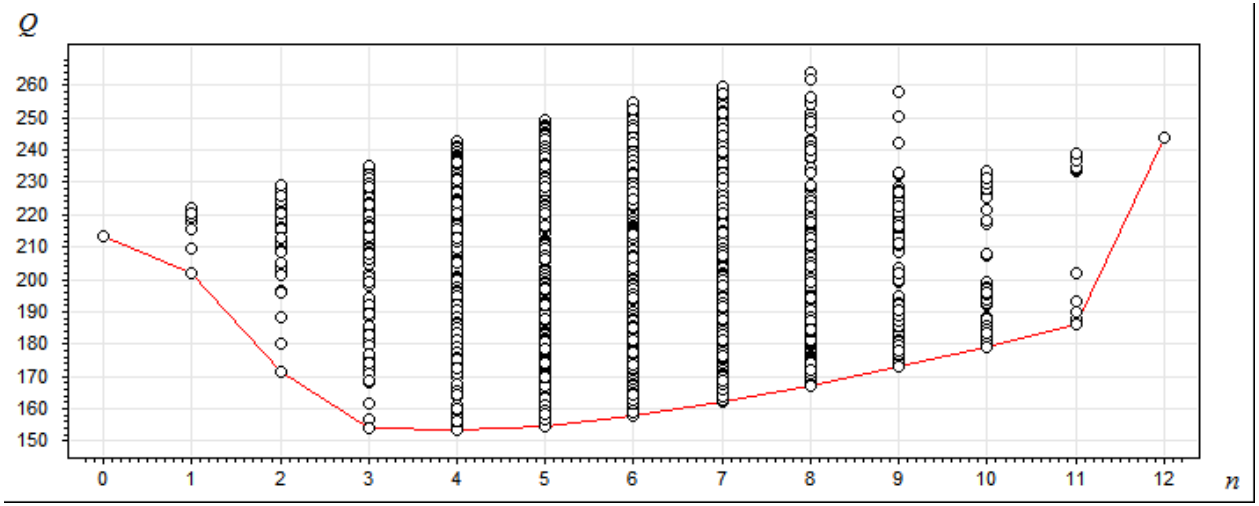
1. Ввести меру индивидуальной информативности признака:
   - $|w_j|$ — модуль веса в линейном классификаторе,
   - $\langle f_j, y \rangle$ — сонаправленность векторов признака и ответов,
   - отношение средних значений признака в классах $+1$ и -1,
   - доля лучших наборов J: $|J| \leqslant 3$, содержащих признак f_j.
2. Отсортировать признаки по убыванию информативности
3. Взять лучшие K признаков
4. Параметр K оптимизировать по внешнему критерию

Часто применяется для предварительного отсева признаков.

Преимущества: простота реализации и скорость.

Недостаток — не во всех задачах это приводит к успеху.

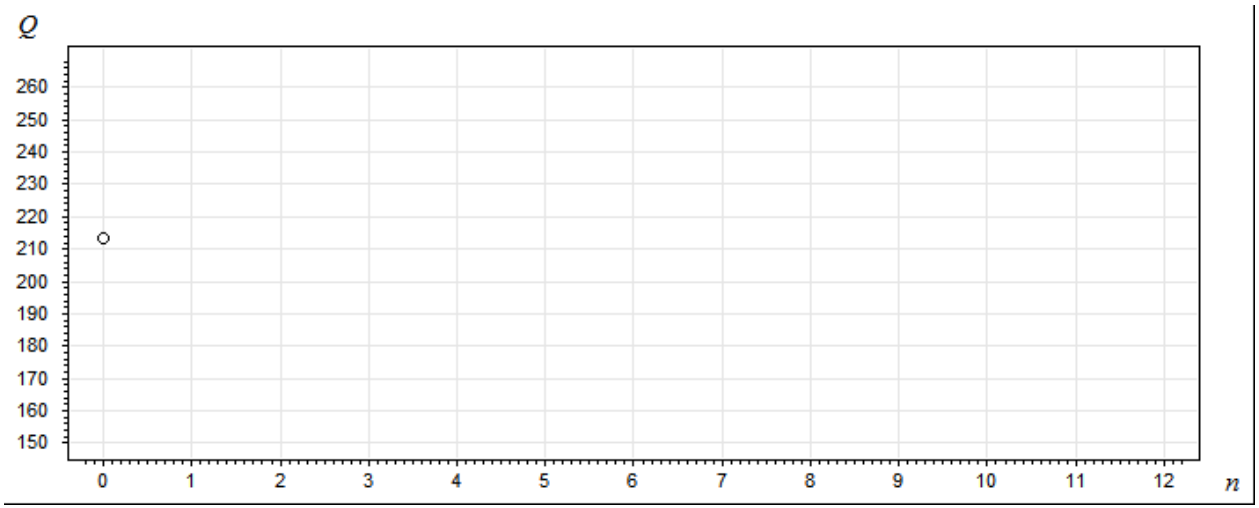
Алгоритм полного перебора (Full Search)



Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

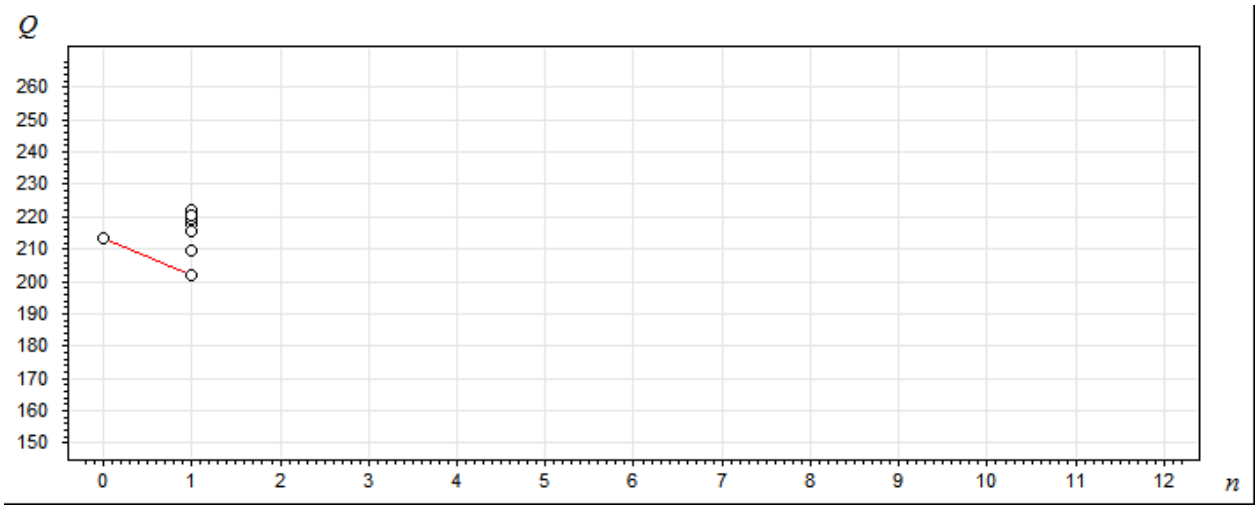


$d = 3$
$j = 0$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

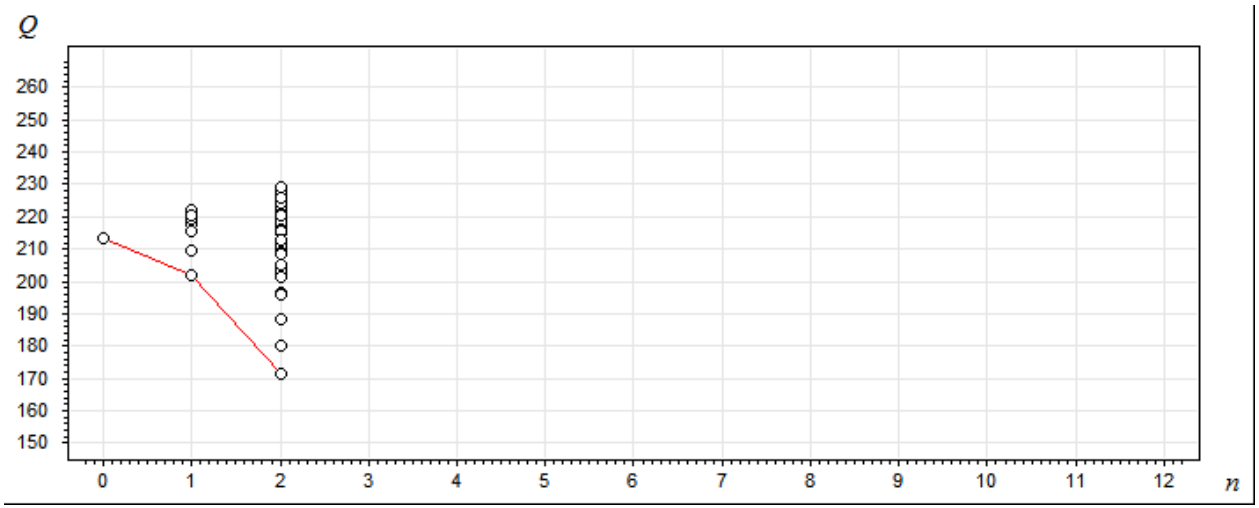


$d = 3$
$j = 1$
$j^* = 1$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

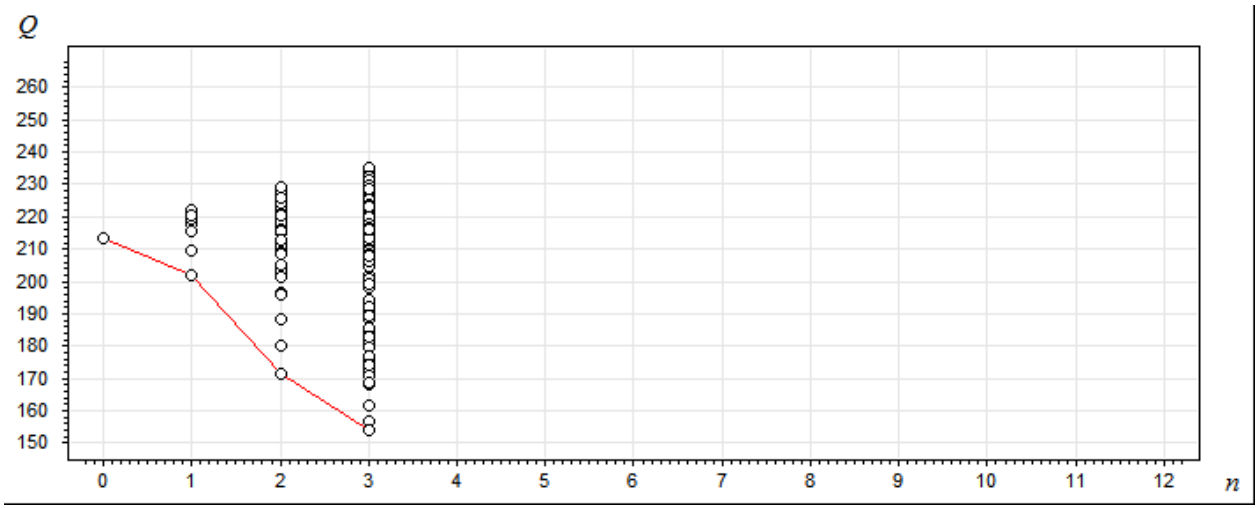


$d = 3$
$j = 2$
$j^* = 2$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

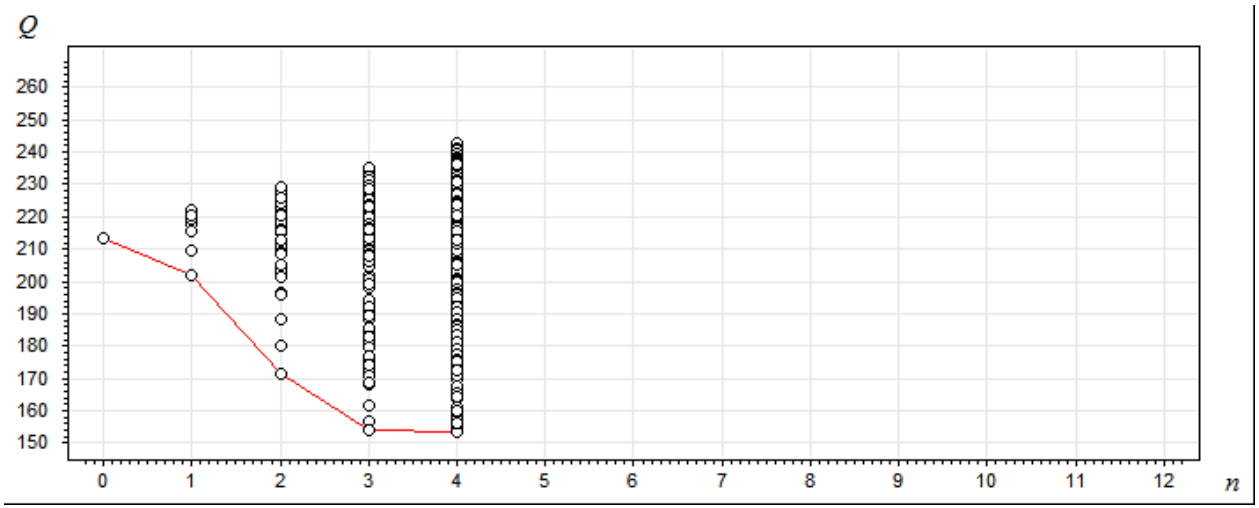


$d = 3$
$j = 3$
$j^* = 3$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

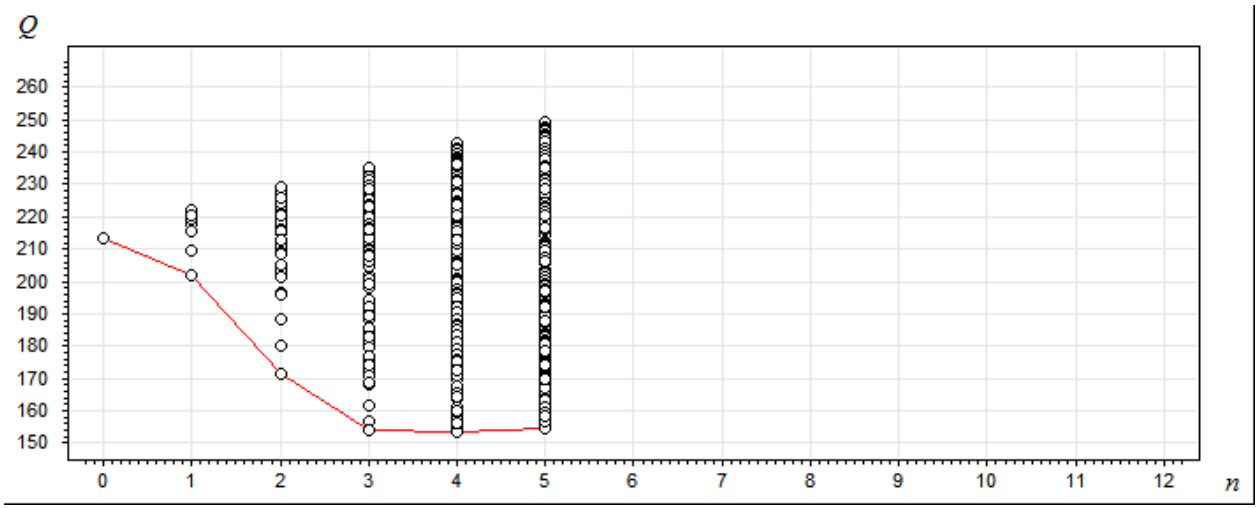


$d = 3$
$j = 4$
$j^* = 4$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

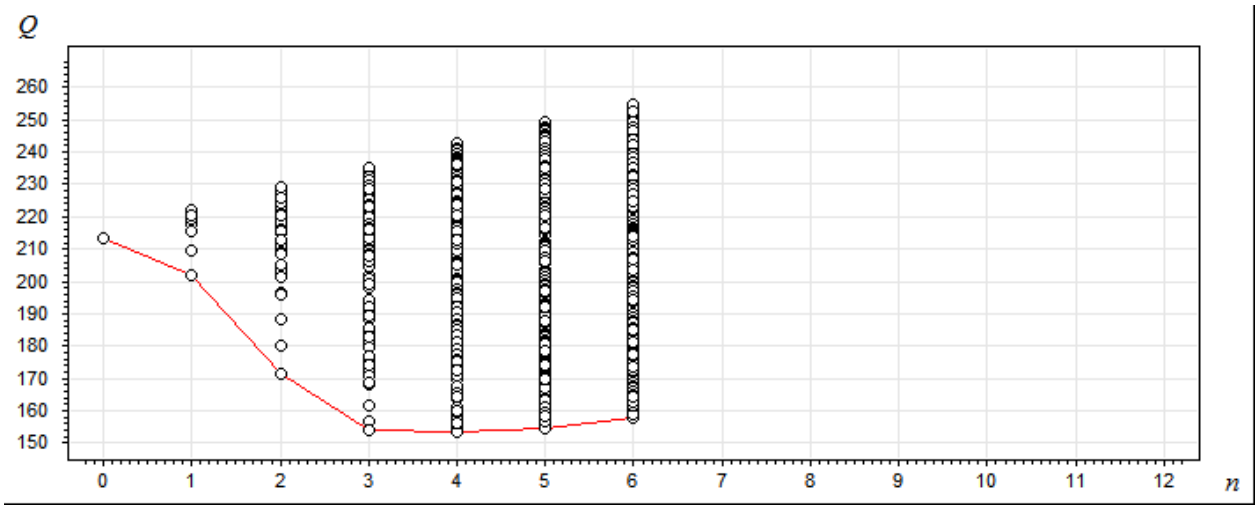


$d = 3$
$j = 5$
$j^* = 4$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)

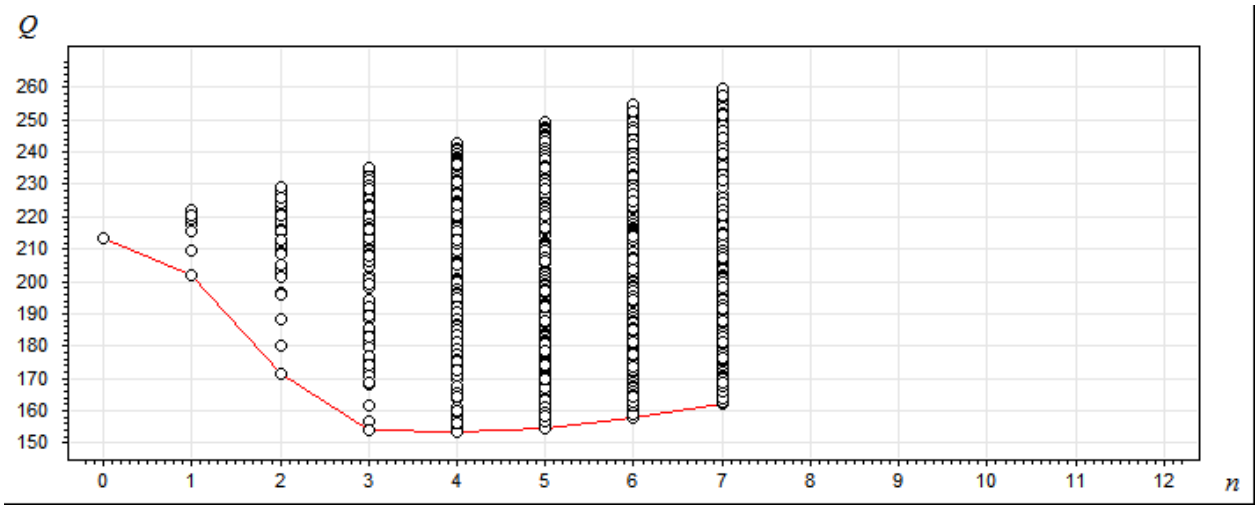


$d = 3$
$j = 6$
$j^* = 4$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Алгоритм полного перебора (Full Search)



$d = 3$
$j = 7$
$j^* = 4$

Вход: множество F, критерий Q, параметр d

1 $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов:
3 $\quad J_j := \arg\min_{J:\,|J|=j} Q(J)$ — лучший набор сложности j;
4 $\quad$ **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
5 $\quad$ **если** $j - j^* \geqslant d$ **то выход** 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 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 F, критерий Q, параметр d;

1 $J_0 := \varnothing$; $Q^* := Q(\varnothing)$ — инициализация;
2 **для всех** $j = 1, \ldots, n$, где j — сложность наборов
3 найти признак, наиболее выгодный для добавления:
$f^* := \arg\min_{f \in F \setminus J_{j-1}} Q\left(J_{j-1} \cup \{f\}\right)$;
4 добавить этот признак в набор:
$J_j := J_{j-1} \cup \{f^*\}$;
5 **если** $Q(J_j) < Q^*$ **то** $j^* := j$; $Q^* := Q(J_j)$;
6 **если** $j - j^* \geqslant d$ **то выход** J_{j^*} ;

Переборные методы отбора признаков

Преимущества жадного Add:

- относительно быстрый — $O(n^2)$, точнее $O(n(j^* + d))$;

Недостатки жадного Add:

- склонен включать в набор лишние признаки

Другие методы отбора признаков:

- Add-Del — чередование добавлений и удалений
- поиск в ширину (метод группового учёта аргументов)
- поиск в глубину (метод ветвей и границ)
- эволюционные (генетические) алгоритмы
- случайный поиск с адаптацией

Определения и обозначения

U — множество субъектов (users/пользователей/клиентов);
I — множество объектов (items/предметов/товаров/ресурсов);

r_{ui} — матрица рейтингов или бинарных значений,
например, $r_{ui} = [\text{клиент } u \text{ использовал товар } i]$;

$D = \{(u, i) \mid \text{значение } r_{ui} \text{ известно}\}$

Задачи:

- прогнозирование незаполненных ячеек r_{ui};
- формирование списка рекомендаций для u или для i.
- оценивание сходства: $\rho(u, u')$, $\rho(i, i')$, $\rho(u, i)$;

Рекомендательная система на основе рейтингов

U — клиенты интернет-магазина;

I — товары (книги, фильмы, музыка, и т.п.);

r_{ui} = рейтинг, который клиент u выставил товару i;

Пример: конкурс Netflix [www.netflixprize.com]

- 2 октября 2006 — 21 сентября 2009; главный приз — \$$10^6$;
- $|U| = 0.48 \cdot 10^6$; $|I| = 17 \cdot 10^3$;
- 10^8 рейтингов $\{1, 2, 3, 4, 5\}$;
- точность прогнозов оценивается по тестовой выборке D':

$$\text{RMSE}^2 = \frac{1}{|D'|} \sum_{(u,i) \in D'} \left(r_{ui} - \hat{r}_{ui}\right)^2;$$

- задача: уменьшить RMSE с 0.9514 до 0.8563 (на 10%).

Матричные разложения

T — множество тем (интересов): $|T| \ll |U|$, $|T| \ll |I|$;
p_{tu} — неизвестный *профиль клиента* u; $P = (p_{tu})_{|T| \times |U|}$;
q_{ti} — неизвестный *профиль объекта* i; $Q = (q_{ti})_{|T| \times |I|}$;

Задача: найти разложение $r_{ui} = \sum_{t \in T} p_{tu} q_{ti}$

Матричная запись: $R = P^{\mathsf{T}} Q$

Вероятностный смысл: $\underbrace{p(i|u)}_{r_{ui}} = \sum_{t \in T} \underbrace{p(i|t)}_{q_{ti}} \cdot \underbrace{p(t|u)}_{p_{tu}}$;

Методы решения:
SVD — сингулярное разложение (плохо интерпретируется);
NNMF — неотрицательное матричное разложение: $p_{tu} \geqslant 0$, $q_{ti} \geqslant 0$;
PLSA — вероятностный латентный семантический анализ.

Разреженный SVD (Singular Value Decomposition)

Обычный не разреженный SVD: $\|R - P^{\mathsf{T}}Q\|^2 \to \min\limits_{P,Q}$.

Разреженный SVD: $\sum\limits_{(u,i)\in D} \Big(\underbrace{r_{ui} - \bar{r}_u - \bar{r}_i - \sum\limits_{t\in T} p_{tu}q_{ti}}_{\varepsilon_{ui}} \Big)^2 \to \min\limits_{P,Q}$

Метод стохастического градиента:
перебираем все $(u,i) \in D$ многократно в случайном порядке
и делаем каждый раз градиентный шаг для задачи $\varepsilon_{ui}^2 \to \min\limits_{p_u,q_i}$:

$$p_{tu} := p_{tu} + \eta\varepsilon_{ui}q_{ti}, \quad t \in T;$$
$$q_{ti} := q_{ti} + \eta\varepsilon_{ui}p_{tu}, \quad t \in T;$$

Tacáks G., Pilászy I., Németh B., Tikk D. Scalable collaborative filtering approaches for large recommendation systems // JMLR, 2009, No. 10, Pp. 623–656.

Разреженный SVD (Singular Value Decomposition)

Преимущества метода стохастического градиента:

- легко вводится регуляризация:
$$\varepsilon_{ui}^2 + \lambda\|p_u\|^2 + \mu\|q_i\|^2 \to \min_{p_u, q_i};$$
- легко вводятся ограничения неотрицательности:
$$p_{tu} \geqslant 0, \quad q_{ti} \geqslant 0 \quad \text{(метод проекции градиента)};$$
- легко вводится обобщение для ранговых данных:
$$\sum_{(u,i)\in D} \left(r_{ui} - \bar{r}_u - \bar{r}_i - \beta\Big(\sum_{t\in T} p_{tu} q_{ti}\Big)\right)^2 \to \min_{P,Q,\beta}.$$
- легко реализуются все виды инкрементности: добавление
  – ещё одного клиента u,
  – ещё одного объекта i,
  – ещё одного значения r_{ui}.
- высокая численная эффективность на больших данных;

NNMF (Non-Negative Matrix Factorization)

Метод чередующихся наименьших квадратов (Alternating Least Squares, ALS):

$$D = \left\|R - \sum_{t\in T} p_t q_t^{\mathsf{T}}\right\|^2 = \left\|R_t - p_t q_t^{\mathsf{T}}\right\|^2 \to \min_{\{p_t \geqslant 0,\ q_t \geqslant 0\}}$$

Идея: искать поочерёдно то строки p_t, то строки q_t при фиксированных остальных $s \neq t$, $R_{tui} = r_{ui} - \sum_{s\in T\setminus t} p_{su} q_{si}$.

$$\frac{\partial D}{\partial p_t} = 0 \;\Rightarrow\; (p_t^{\mathsf{T}} q_t - R_t) q_t^{\mathsf{T}} = 0 \quad\Rightarrow\quad p_{tu} = \left(\frac{\sum_i q_{ti} R_{tui}}{\sum_i q_{ti}^2}\right)_+$$

$$\frac{\partial D}{\partial q_t} = 0 \;\Rightarrow\; p_t (p_t^{\mathsf{T}} q_t - R_t) = 0 \quad\Rightarrow\quad q_{ti} = \left(\frac{\sum_u p_{tu} R_{tui}}{\sum_u p_{tu}^2}\right)_+$$

Cichocki A., Zdunek R., Amari S., Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization // Springer LNCS, 2007, v.4666, pp.169–176.

Использование латентных профилей

Предсказание незаполненных ячеек:

$$\hat{r}_{ui} = \sum_{t \in T} p_{tu} q_{ti}.$$

Рекомендация клиенту u — список объектов i, ранжированных по убыванию $\hat{r}_{ui}$.

Поиск и ранжирование клиентов и объектов по сходству:

$$\rho^2(u, u') = \sum_{t \in T} (p_{tu} - p_{tu})^2,$$

$$\rho^2(i, i') = \sum_{t \in T} (q_{ti} - q_{ti})^2,$$

p_{tu} — сжатое признаковое описание клиента u,
q_{ti} — сжатое признаковое описание объекта i.

Резюме

- Недостаточная сложность модели ведёт к недообучению
- Избыточная сложность модели ведёт к переобучению
- Для оптимизации сложности нужен внешний критерий
- Чем больше данных, тем более сложную модель можно построить
- Способы сокращения размерности/сложности модели:
  — отбор признаков;
  — преобразование признаков;
  — латентные модели типа матричных разложений

Воронцов Константин Вячеславович
voron@forecsys.ru

www.MachineLearning.ru • Участник:Vokov

Если что-то было не понятно,
не стесняйтесь подходить и спрашивать :)