• Прикладные модели машинного обучения •
Методы обучения ранжированию

Воронцов Константин Вячеславович
k.v.vorontsov@phystech.edu
http://www.MachineLearning.ru/wiki?title=User:Vokov

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

МФТИ • 14 марта 2025

Содержание

1. Постановка задачи и основные подходы
   - Поточечный подход
   - Попарный подход
   - Списочный подход
2. Ранжирование в поисковых системах
   - Признаки ранжирования
   - Функционалы качества ранжирования
   - Вероятностная модель поведения пользователя
3. Нейросетевые модели поиска
   - Модель DSSM (Deep Structured Semantic Model)
   - Хэширование слов
   - Преимущества DSSM

Определения и обозначения

Дано: $X^\ell = \{x_1, \ldots, x_\ell\}$ — обучающая выборка,
$i \prec j$ — отношение «x_j лучше, чем x_i» между объектами из X^ℓ

Найти: ранжирующую функцию $a\colon X \to \mathbb{R}$,
восстанавливающую правильное отношение порядка:

$$i \prec j \quad \Rightarrow \quad a(x_i) < a(x_j)$$

Критерий конструируется по-разному в трёх подходах:

- Point-wise — поточечный (аналог регрессии/классификации)
- Pair-wise — попарный (качество парных сравнений)
- List-wise — списочный (качество ранжированного списка)

Линейная модель ранжирования:

$$a(x, w) = \langle x, w \rangle$$

где $x \mapsto \big(f_1(x), \ldots, f_n(x)\big) \in \mathbb{R}^n$ — вектор признаков объекта x

Примеры задач ранжирования

Ранжирование (Learning to Rank, LtR, L2R, LETOR) нужно везде, где система предлагает человеку варианты для принятия решения, возможная основа *гибридного интеллекта*.

Примеры приложений:

- ранжирование выдачи поисковой системы
- ранжирование рекомендаций пользователям (книги, фильмы, музыка, товары интернет-магазина, и т.п.)
- ранжирование вариантов автоматического завершения запроса (Query Auto Completion, auto-suggest)
- ранжирование возможных ответов в диалоговых системах (Question Answering Systems)
- ранжирование вариантов перевода фраз в машинном переводе художественного текста (Machine Translation)

Ранговая регрессия (Ordinal Regression)

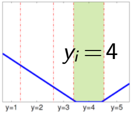
Обучающая выборка $(x_i, y_i)_{i=1}^{\ell}$, где $y_i \in Y = \{1 \prec 2 \prec \dots \prec R\}$.
Функция ранжирования с параметрами w
и порогами $b_0 = -\infty$, $b_1 \leqslant \ldots \leqslant b_{R-1}$, $b_R = +\infty$:

$$a(x, w, b) = y, \quad \text{если} \quad b_{y-1} < g(x, w) \leqslant b_y$$

Функция потерь $\mathscr{L}(M)$ — убывающая функция отступа M

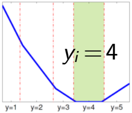
Сумма потерь по двум ближайшим порогам:

$$\sum_{i=1}^{\ell} \mathscr{L}\bigl(g(x_i, w) - b_{y_i-1}\bigr) + \mathscr{L}\bigl(b_{y_i} - g(x_i, w)\bigr) \to \min_{w,b}$$



Сумма потерь по всем порогам:

$$\sum_{i=1}^{\ell} \sum_{y=1}^{R} \mathscr{L}\Bigl(\bigl(b_y - g(x_i, w)\bigr) \operatorname{sign}\bigl(y - y_i\bigr)\Bigr) \to \min_{w,b}$$



J.D.M.Rennie, N.Srebro. Loss functions for preference levels: regression with discrete ordered labels. IJCAI-2005.

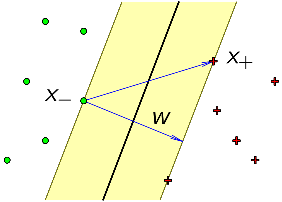
Напоминание. SVM — метод опорных векторов

Линейный классификатор, $Y = \{-1, +1\}$:

$$a(x, w, w_0) = \mathrm{sign}\bigl(\langle w, x\rangle - w_0\bigr), \quad w, x \in \mathbb{R}^n,\ w_0 \in \mathbb{R}$$

Задача обучения SVM:

$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum\limits_{i=1}^{\ell}\xi_i \to \min\limits_{w, w_0, \xi} \\ M_i(w, w_0) \geqslant 1 - \xi_i, \quad i = 1, \dots, \ell \\ \xi_i \geqslant 0, \quad i = 1, \dots, \ell \end{cases}$$



где $M_i(w, w_0) = y_i\bigl(\langle w, x_i\rangle - w_0\bigr)$ — отступ объекта x_i

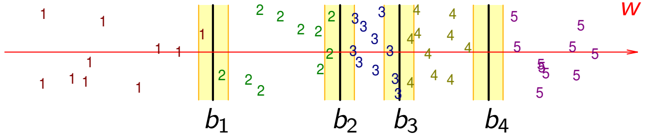
Эквивалентная задача безусловной минимизации:

$$\sum_{i=1}^{\ell}\bigl(1 - M_i(w, w_0)\bigr)_+ + \frac{1}{2C}\|w\|^2 \to \min_{w, w_0}$$

Ранговый SVM (Support Vector Ordinal Regression, SVOR)

Частный случай: линейная модель $g(x,w) = \langle w, x\rangle$, сумма по двум порогам, функция потерь $\mathscr{L}(M) = (1-M)_+$:

$$\sum_{i=1}^{\ell} \bigl(1 - \langle x_i, w\rangle + b_{y_i-1}\bigr)_+ + \bigl(1 + \langle x_i, w\rangle - b_{y_i}\bigr)_+ + \frac{1}{2C}\|w\|^2 \to \min_{w,b}$$



Эквивалентная задача квадратичного программирования:

$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum\limits_{i=1}^{\ell} [y_i \neq 1]\,\xi_i^* + [y_i \neq R]\,\xi_i \to \min\limits_{w,b,\xi} \\ \langle x_i, w\rangle \geqslant b_{y_i-1} + 1 - \xi_i^*, \quad \xi_i^* \geqslant 0, \quad y_i \neq 1; \\ \langle x_i, w\rangle \leqslant b_{y_i} - 1 + \xi_i, \quad \xi_i \geqslant 0, \quad y_i \neq R; \quad b_r \leqslant b_{r+1} \end{cases}$$

Ранговый SVM (Support Vector Ordinal Regression, SVOR)

Двойственная задача ($\lambda_i^* = 0$ при $y_i = 1$, $\lambda_i = 0$ при $y_i = R$):

$$\begin{cases} \sum_{i=1}^{\ell}(\lambda_i^* + \lambda_i) - \frac{1}{2}\sum_{i,j=1}^{\ell}(\lambda_i^* - \lambda_i)(\lambda_j^* - \lambda_j)K(x_i, x_j) \to \max\limits_{\lambda^*,\lambda,\mu}; \\ \mu_r + \sum_{i=1}^{\ell}\lambda_i[y_i = r] = \mu_{r+1} + \sum_{i=1}^{\ell}\lambda_i^*[y_i = r+1], \ \ r = 1, \dots, R-1; \\ 0 \leqslant \lambda_i^* \leqslant C; \quad 0 \leqslant \lambda_i \leqslant C; \quad \mu_r \geqslant 0 \end{cases}$$

Модель ранжирования после решения двойственной задачи:

$$\langle w, x\rangle = \sum_{i=1}^{\ell}(\lambda_i^* - \lambda_i)K(x, x_i)$$

Преимущества SVOR — те же, что у SVM:

- задача выпуклая, имеет единственное решение
- возможны нелинейные обобщения с ядрами $K(x, x')$
- решение разреженное, зависит только от опорных векторов

Wei Chu, Sathiya Keerthi. Support Vector Ordinal Regression. 2007.

Попарный подход (pair-wise approach)

Переход к гладкому функционалу качества ранжирования:

$$Q(w) = \sum_{i \prec j} \bigl[\underbrace{a(x_j, w) - a(x_i, w)}_{\text{margin } M_{ij}(w)} < 0\bigr]$$

$$\leqslant \sum_{i \prec j} \mathscr{L}\bigl(a(x_j, w) - a(x_i, w)\bigr) \;\to\; \min_w$$

где $a(x, w)$ — параметрическая модель ранжирования,

$\mathscr{L}(M)$ — убывающая функция парного отступа $M_{ij}(w)$:

- $\mathscr{L}(M) = (1 - M)_+$ — RankSVM, Ranking SVM и др.
- $\mathscr{L}(M) = \exp(-M)$ — RankBoost
- $\mathscr{L}(M) = \log(1 + e^{-M})$ — RankNet

R.Herbrich, T.Graepel, K.Obermayer. Support vector learning for ordinal regression. 1999
Y. Freund et al. An efficient boosting algorithm for combining preferences. 2003

Напоминание. Градиентная максимизация AUC

Модель классификации: $a(x_i, w, w_0) = \mathrm{sign}\bigl(g(x_i, w) - w_0\bigr)$.

AUC — это доля правильно упорядоченных пар (x_i, x_j):

$$\mathrm{AUC}(w) = \frac{1}{\ell_-} \sum_{i=1}^{\ell} \bigl[y_i = -1\bigr] \mathrm{TPR}_i = \\ = \frac{1}{\ell_- \ell_+} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \bigl[y_i < y_j\bigr] \bigl[g(x_i, w) < g(x_j, w)\bigr] \to \max_w .$$

Явная максимизация аппроксимированного AUC:

$$1 - \mathrm{AUC}(w) \leqslant Q(w) = \sum_{i,j\colon\, y_i < y_j} \mathscr{L}\bigl(\underbrace{g(x_j, w) - g(x_i, w)}_{M_{ij}(w)}\bigr) \to \min_w,$$

где $\mathscr{L}(M)$ — убывающая функция парного отступа $M_{ij}(w)$.

Напоминание. Алгоритм SG для максимизации AUC

Возьмём для простоты линейный классификатор:

$$g(x, w) = \langle x, w \rangle, \qquad M_{ij}(w) = \langle x_j - x_i, w \rangle, \qquad y_i < y_j.$$

Вход: выборка X^ℓ, темп обучения h, темп забывания λ;
Выход: вектор весов w;

инициализировать веса w_j, $j = 0, \ldots, n$;
инициализировать оценку: $\bar{Q} := \frac{1}{\ell_+\ell_-} \sum\limits_{i,j} [y_i < y_j]\, \mathscr{L}(M_{ij}(w))$;
повторять
- выбрать пару объектов (i, j): $y_i < y_j$, случайным образом;
- вычислить потерю: $\varepsilon_{ij} := \mathscr{L}(M_{ij}(w))$;
- сделать градиентный шаг: $w := w - h\,\mathscr{L}'(M_{ij}(w))(x_j - x_i)$;
- оценить функционал: $\bar{Q} := (1-\lambda)\bar{Q} + \lambda\varepsilon_{ij}$;

пока значение $\bar{Q}$ и/или веса w не сойдутся;

Ranking SVM: метод опорных векторов для ранжирования

Постановка задачи SVM для попарного подхода:

$$Q(w) = \tfrac{1}{2}\|w\|^2 + C \sum_{i \prec j} \mathscr{L}\bigl(\underbrace{a(x_j, w) - a(x_i, w)}_{\text{margin } M_{ij}(w)}\bigr) \;\to\; \min_w,$$

где $a(x, w) = \langle w, x\rangle$ — линейная функция ранжирования
$\mathscr{L}(M) = (1 - M)_+$ — «шарнирная» функция потерь (hinge loss)
$M = M_{ij}(w) = \langle w, x_j - x_i\rangle$ — отступ

Постановка задачи квадратичного программирования:

$$\begin{cases} \tfrac{1}{2}\|w\|^2 + C \sum\limits_{i \prec j} \xi_{ij} \to \min\limits_{w,\xi} \\ \langle w, x_j - x_i\rangle \geqslant 1 - \xi_{ij}, \quad i \prec j \\ \xi_{ij} \geqslant 0, \quad i \prec j \end{cases}$$

T. Joachims. Optimizing search engines using clickthrough data. 2002.

RankNet: логистическая регрессия для ранжирования

RankNet: функция потерь $\mathscr{L}(M) = \log(1 + e^{-\sigma M})$,
модель $a(x_i, w) = a_i(w)$ — нейронная сеть или бустинг:

$$Q(w) = \sum_{i \prec j} \mathscr{L}\bigl(a_j(w) - a_i(w)\bigr) \;\to\; \min_w$$

Метод стохастического градиента:
выбираем на каждой итерации случайную пару $i \prec j$:

$$w := w - \eta \cdot \underbrace{\mathscr{L}'\bigl(a_j(w) - a_i(w)\bigr)}_{\lambda_{ij}} \cdot \nabla_w \bigl(a_j(w) - a_i(w)\bigr)$$

Более эффективное обновление: выбираем случайный объект x_i и пакет (mini-batch) всех объектов, с которыми он сравним:

$$w := w - \eta \sum_j \lambda_{ij} \cdot \Bigl([i \succ j] - [i \prec j]\Bigr) \cdot \nabla_w a_i(w)$$

C.Burges et al. Learning to rank using gradient descent. 2005

От попарного RankNet к списочному LambdaRank

Пусть $\tilde{Q}$ — негладкий функционал качества ранжирования, в частности, для его вычисления список объектов x_i может ранжироваться по убыванию значений $a(x_i, w)$.

$\Delta\tilde{Q}_{ij}$ — изменение $\tilde{Q}$ при перестановке $x_i \leftrightarrows x_j$ в списке, разностный аналог производной $\lambda_{ij} = \mathscr{L}'\bigl(a_j(w) - a_i(w)\bigr)$.

LambdaRank: домножение градиента на $|\Delta\tilde{Q}_{ij}|$ приводит к приближённой оптимизации негладкого функционала $\tilde{Q}$:

$$w := w - \eta \sum\nolimits_j |\Delta\tilde{Q}_{ij}| \cdot \Bigl([i \succ j] - [i \prec j]\Bigr) \cdot \nabla_w a_i(w)$$

Если $i \succ j$, то w изменяется в сторону увеличения $a_i(w)$.
Если $i \prec j$, то w изменяется в сторону уменьшения $a_i(w)$.
Сумма этих изменений смещает x_i выше или ниже по списку.

C.Burges. From RankNet to LambdaRank to LambdaMART: an overview. 2010

Задача ранжирования поисковой выдачи

D — множество web-страниц или документов (documents)
Q — множество запросов (queries)
$D_q \subseteq D$ — множество документов, найденных по запросу q
$X = Q \times D$ — объектами являются пары «запрос, документ»:

$$x \equiv (q, d),\ \ q \in Q,\ \ d \in D_q$$

Y — упорядоченное множество рейтингов
$y\colon X \to Y$ — оценки релевантности, поставленные асессорами:
чем выше оценка $y(q, d)$, тем релевантнее документ d запросу q

Правильный порядок определён только между документами, найденными по одному и тому же запросу q:

$$(q, d) \prec (q, d') \quad \Leftrightarrow \quad y(q, d) < y(q, d')$$

Признаки $f(q,d)$, $f(d)$ для ранжирования поисковой выдачи

- **текстовые, документные**
  слова запроса q встречаются в d чаще обычного
  слова запроса q встречаются в d подряд как фраза
  слова запроса q есть в заголовках или выделены в d
  длина d, возраст d, читабельность d, мультимедиа в d
- **ссылочные**
  число ссылок на документ d, на сайт, на домен
  число ссылок из тематически близких документов (ТИЦ)
  число полезных ссылок, содержащихся в документе d
- **поведенческие, кликовые**
  на документ d часто кликают (всего / по запросу q)
  на документе d долго задерживаются
  после документа d редко возвращаются к поиску
- **пользовательские** — для персонализации поиска

TF-IDF(q,d) — мера релевантности документа d запросу q

n_{dw} (term frequency) — число вхождений слова w в текст d
N_w (document frequency) — число документов, содержащих w
N — число документов в коллекции D

N_w/N — оценка вероятности встретить слово w в документе
$(N_w/N)^{n_{dw}}$ — оценка вероятности встретить его n_{dw} раз

$P(q,d) = \prod_{w\in q}(N_w/N)^{n_{dw}}$ — оценка вероятности встретить в документе d слова запроса $q = \{w_1,\ldots,w_k\}$ *чисто случайно*

Оценка релевантности запроса q документу d:

$$\text{TF-IDF}(q,d) = -\log P(q,d) = \sum_{w\in q} \underbrace{n_{dw}}_{\text{TF}(w,d)} \underbrace{\log(N/N_w)}_{\text{IDF}(w)} \to \max$$

$\text{TF}(w,d) = n_{dw}$ — term frequency
$\text{IDF}(w) = \log(N/N_w)$ — inverted document frequency

Семейство мер релевантности Best Matching (Okapi BM25)

Модификация TF-IDF:

- рост TF ограничивается сверху
- TF уменьшается для длинных документов
- вес IDF для частых слов становится ещё меньше

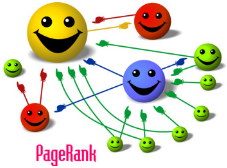
$$\mathrm{BM}(q,d) = \sum_{w\in q} \frac{n_{dw}(k_1+1)}{n_{dw} + k_1\left(1 - b + b\frac{n_d}{\bar{n}_d}\right)} \max\left\{\log\frac{N - N_w + \frac{1}{2}}{N_w + \frac{1}{2}}, \varepsilon\right\}$$

n_d — длина документа d
$\bar{n}_d$ — средняя длина документов в коллекции
$b \in [0,1]$ управляет учётом длины документа (обычно $b = 0.75$)
$k_1 \geqslant 0$ ограничивает линейный рост TF (обычно $k_1 = 2$)
ε ограничивает снизу IDF (обычно $\varepsilon = 0$)

S.Robertson, H.Zaragoza. The probabilistic relevance framework: BM25 and beyond. 2009

PageRank — классический ссылочный признак

Документ d тем важнее,
чем больше ссылок других документов c на d,
чем важнее документы c, ссылающиеся на d,
чем меньше других ссылок имеют эти c.



Вероятность посетить страницу d, кликая по ссылкам случайно:

$$\mathrm{PR}(d) = (1-\delta)\frac{1}{N} + \delta \sum_{c \in D_d^{in}} \frac{\mathrm{PR}(c)}{|D_c^{out}|},$$

$D_d^{in} \subset D$ — множество документов, ссылающихся на d,
$D_c^{out} \subset D$ — множество документов, на которые ссылается c,
$\delta = 0.85$ — вероятность продолжать клики (damping factor),
N — число документов в коллекции D.

Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd. The PageRank citation ranking: bringing order to the Web. 1998.

Поведенческие признаки ранжирования

- CTR(d), CTR(q, d) — кликабельность, Click-Through Rate — отношение числа кликов к числу показов
- Вероятность единственного клика / последнего клика
- Средняя длительность посещения, частота посещений
- *Удовлетворённость пользователей* — вероятность завершить поиск после посещения страницы d
- *Глубина просмотра* — число страниц сайта, посещаемых пользователями через страницу d в течение одной сессии
- BrowseRank(d) — аналог PageRank(d), оценка доли времени, проводимого пользователями на странице d; страницы и ссылки образуют граф, как и в PageRank, но:
  — для каждого d дано распределение времени посещения,
  — для каждой ссылки дано число переходов пользователей.

Yuting Liu et al. BrowseRank: letting Web users vote for page importance. 2008

Оценивание качества поиска

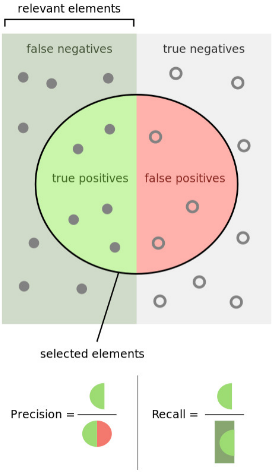
Precision — доля релевантных среди найденных
Recall — доля найденных среди релевантных

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \text{ — точность (precision)}$$

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \text{ — полнота (recall)}$$

$$F_1 = \frac{2PR}{P + R} \text{ — F1-мера}$$

TP (true positive) — найденные релевантные
FP (false positive) — найденные нерелевантные
FN (false negative) — ненайденные релевантные
TN (true negative) — не должен учитываться



Проблема: в «большом поиске» FN и Recall неизвестны

Точность, средняя точность, усреднённая средняя точность

Пусть $Y = \{0, 1\}$, $y(q, d)$ — релевантность,
$a(q, d)$ — оцениваемая функция ранжирования,
$d_q^{(i)}$ — i-й документ по убыванию $a(q, d)$.

Precision, точность — доля релевантных среди первых n:

$$P_n(q) = \frac{1}{n} \sum_{i=1}^{n} y(q, d_q^{(i)})$$

Average Precision, средняя P_n по позициям n релевантных $d_q^{(n)}$:

$$\text{AP}(q) = \sum_{n} y(q, d_q^{(n)}) P_n(q) \Bigg/ \sum_{n} y(q, d_q^{(n)})$$

Mean Average Precision — AP, усреднённая по всем запросам:

$$\text{MAP} = \frac{1}{|Q|} \sum_{q \in Q} \text{AP}(q)$$

Доля «дефектных пар»

Пусть $Y \subseteq \mathbb{R}$, $y(q,d)$ — релевантность,
$a(q,d)$ — оцениваемая функция ранжирования,
$d_q^{(i)}$ — i-й документ по убыванию $a(q,d)$.

Доля инверсий порядка среди первых n документов:

$$\mathrm{DP}_n(q) = \frac{2}{n(n-1)} \sum_{i,j=1}^{n} \bigl[i<j\bigr] \Bigl[y(q,d_q^{(i)}) < y(q,d_q^{(j)})\Bigr]$$

Связь с коэффициентом ранговой корреляции (τ Кенделла):

$$\tau(a,y) = 1 - 2\cdot \mathrm{DP}_n(q)$$

Связь с AUC (area under ROC-curve) в задачах классификации с двумя классами $\{-1,+1\}$, $a\colon X \to \mathbb{R}$:

$$\mathrm{AUC}_n(q) = \frac{1}{\ell_-\ell_+} \sum_{i,j=1}^{n} \bigl[y_i < y_j\bigr] \bigl[a(x_i) < a(x_j)\bigr] = 1 - \frac{n(n-1)}{2\ell_-\ell_+} \mathrm{DP}_n(q)$$

DCG — Discounted Cumulative Gain

Пусть $Y \subseteq \mathbb{R}$, $y(q,d)$ — релевантность,
$a(q,d)$ — оцениваемая функция ранжирования,
$d_q^{(i)}$ — i-й документ по убыванию $a(q,d)$.

Дисконтированная (взвешенная) сумма выигрышей:

$$\mathrm{DCG}_n(q) = \sum_{i=1}^{n} \underbrace{G_q(d_q^{(i)})}_{\text{gain}} \cdot \underbrace{D(i)}_{\text{discount}}$$

$G_q(d) = (2^{y(q,d)} - 1)$ — бóльший вес релевантным документам
$D(i) = 1/\log_2(i+1)$ — бóльший вес в начале выдачи

Нормированная дисконтированная сумма выигрышей:

$$\mathrm{NDCG}_n(q) = \frac{\mathrm{DCG}_n(q)}{\max \mathrm{DCG}_n(q)}$$

$\max \mathrm{DCG}_n(q)$ — это $\mathrm{DCG}_n(q)$ при идеальном ранжировании

Яндекс pFound — модель поведения пользователя

Пусть $Y \subseteq [0, 1]$,
$y(q, d)$ — релевантность, оценка вероятности найти ответ в d,
$a(q, d)$ — оцениваемая функция ранжирования,
$d_q^{(i)}$ — i-й документ по убыванию $a(q, d)$.

Вероятность найти ответ в первых n документах
(по формуле полной вероятности):

$$\mathrm{pFound}_n(q) = \sum_{i=1}^{n} P_i \cdot y\bigl(q, d_q^{(i)}\bigr),$$

где P_i — вероятность дойти до i-го документа:

$$P_1 = 1;$$

$$P_{i+1} = P_i \cdot \bigl(1 - y\bigl(q, d_q^{(i)}\bigr)\bigr) \cdot \bigl(1 - P_{out}\bigr),$$

где P_{out} — вероятность прекратить поиск без ответа

Яндекс pFound — модель поведения пользователя

Параметры критерия pFound:

- $P_{out} = 0.15$ — вероятность прекратить поиск без ответа;
- $y(q, d)$ — оценка вероятности найти ответ в документе, вычисленная по кликовым данным пользователей:

оценка асессора		$y(q, d)$
5	Vital	0.61
4	Useful	0.41
3	Relevant+	0.14
2	Relevant−	0.07
1	Not Relevant	0.00

Гулин А., Карпович П., Расковалов Д., Сегалович И. Оптимизация алгоритмов ранжирования методами машинного обучения. РОМИП-2009.

О ранжировании поисковой выдачи в Яндексе

(По моим устаревшим сведениям)

- Более 50 000 новых оценок асессоров ежемесячно
- За 8 лет придумано и проверено более 2000 признаков
- Pair-wise подход лучше, чем point-wise и list-wise
- Наряду с данными асессоров (explicit relevance feedback) используются большие, но менее надёжные данные о поведении пользователей (implicit relevance feedback)

Технологии:

- **MatrixNet**: модель ранжирования — градиентный бустинг над ODT (небрежными решающими деревьями)
- **CatBoost**: свободно доступный аналог MatrixNet, хорошо работающий с категориальными признаками

Постановка задачи для DSSM (Deep Structured Semantic Model)

Дано: Q — множество запросов
D_q^+ — множество кликнутых документов (clickthrough data)

Найти: вероятностную модель релевантности документов

$$p(d|q) = \operatorname*{SoftMax}_{d\in D_q} \gamma R(q,d) = \frac{\exp\bigl(\gamma R(q,d)\bigr)}{\sum\limits_{d'\in D_q} \exp\bigl(\gamma R(q,d')\bigr)},$$

$R(q,d) = \cos(u_q, u_d)$ — косинусная близость эмбедингов u_q, u_d;
D_q содержит по 4 случайных некликнутых документа вместе с каждым кликнутым $d \in D_q^+$ (Negative Sampling).

Критерий максимума правдоподобия:

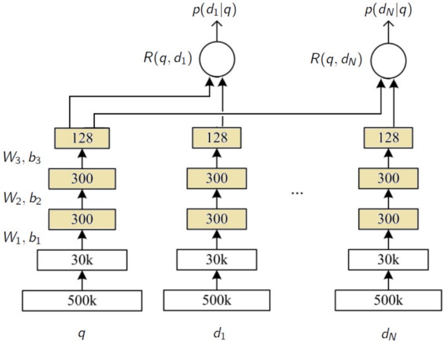
$$\sum_{q\in Q} \sum_{d\in D_q^+} \log p(d|q) \;\to\; \max_{\Omega},$$

max по параметрам кодировщика $u_d = f(d, \Omega)$

Нейросетевой кодировщик в DSSM

Трёхслойная *сиамская* нейронная сеть с параметрами $\Omega = (W_1, b_1, W_2, b_2, W_3, b_3)$

$$\begin{aligned} u_d &= \mathrm{th}(W_3 v_d^2 + b_3) \\ v_d^2 &= \mathrm{th}(W_2 v_d^1 + b_2) \\ v_d^1 &= \mathrm{th}(W_1 x_d + b_1) \\ x_d &= \mathrm{WordHash}(d) \end{aligned}$$



Хэширование слов (word hashing): документ d представляется вектором частот не слов n_{dw}, а буквенных триграмм: WordHash (дармолюб) = {_да, арм, рмо, мол, олю, люб, юб_}

Po-Sen Huang, et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. 2013.

Преимущества DSSM

- Благодаря Word Hashing:
  — сокращается число векторов x_d с 500k до 30k,
  — схожие по написанию слова имеют близкие векторы,
  — появляется возможность обрабатывать новые слова,
  — а также слова с опечатками
- В отличие от других эмбедингов, которые обучаются реконструировать данные без учителя, DSSM обучается с учителем, по большим данным о кликах пользователей
- Поэтому он опережает по качеству поиска как частотные модели (TF-IDF, BM25), так и векторные (PLSA, LDA, DAE)

Po-Sen Huang, et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. 2013.

Резюме в конце лекции

- Ранжирование — особый класс задач машинного обучения:
  — по обучающей выборке похоже на классификацию,
  — по функции ранжирования похоже на регрессию.
- Критерий качества ранжирования зависит от приложения. Наилучшего универсального критерия не существует.
- Три подхода: поточечный, попарный, списочный. Теоретически списочный должен быть наилучшим. Однако в Яндексе долгое время лучше работал попарный.
- Со временем становится всё труднее создавать и улучшать признаки ранжирования, «большой поиск» перешёл на глубокие нейронные сети для ранжирования.

Tie-Yan Liu. Learning to Rank for Information Retrieval. 2011
Hang Li. A Short Introduction to Learning to Rank. 2011
Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, Hang Li. Listwise approach to learning to rank: theory and algorithm. 2008