

# Методы обучения ранжированию (Learning to Rank)

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)»

МФТИ • 24 марта 2023

- 1 Постановка задачи и основные подходы**
  - Поточечный подход
  - Парный подход
  - Списочный подход
- 2 Ранжирование в поисковых системах**
  - Признаки ранжирования
  - Функционалы качества ранжирования
  - Ранжирование поисковой выдачи в Яндексе
- 3 Нейросетевые модели поиска**
  - Модель DSSM (Deep Structured Semantic Model)
  - Хэширование слов
  - Преимущества DSSM

## Определения и обозначения

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

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

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

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

## Примеры задач ранжирования

Ранжирование (Learning to Rank, LtR, L2R, LETOR)  
нужно везде, где система предоставляет пользователю выбор  
из большого числа вариантов

- 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 в системах машинного перевода (Machine Translation)

## Ранговая регрессия (Ordinal Regression)

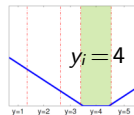
Обучающая выборка  $(x_i, y_i)_{i=1}^{\ell}$ , где  $y_i \in Y = \{1 < 2 < \dots < R\}$ .  
 Функция ранжирования с параметрами  $w$   
 и порогами  $b_0 = -\infty, b_1 \leq \dots \leq b_{R-1}, b_R = +\infty$ :

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

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

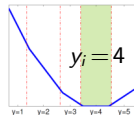
Сумма потерь по двум ближайшим порогам:

$$\sum_{i=1}^{\ell} \mathcal{L}(g(x_i, w) - b_{y_i-1}) + \mathcal{L}(b_{y_i} - g(x_i, w)) \rightarrow \min_{w, b}$$



Сумма потерь по всем порогам:

$$\sum_{i=1}^{\ell} \sum_{y=1}^R \mathcal{L}((b_y - g(x_i, w)) \text{ sign}(y - y_i)) \rightarrow \min_{w, b}$$

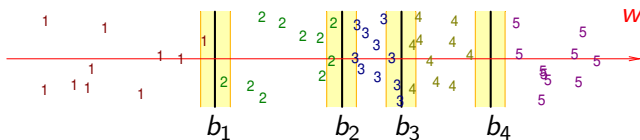


*J.D.M.Rennie, N.Srebro. Loss functions for preference levels: regression with discrete ordered labels. IJCAI-2005.*

## Ранговый SVM (Support Vector Ordinal Regression, SVOR)

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

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



Эквивалентная задача квадратичного программирования:

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{\ell} [y_i \neq 1] \xi_i^* + [y_i \neq R] \xi_i \rightarrow \min_{w,b,\xi} \\ \langle x_i, w \rangle \geq b_{y_i-1} + 1 - \xi_i^*, & \xi_i^* \geq 0, & y_i \neq 1; \\ \langle x_i, w \rangle \leq b_{y_i} - 1 + \xi_i, & \xi_i \geq 0, & y_i \neq R; \end{cases}$$

## Ранговый SVM (Support Vector Ordinal Regression, SVOR)

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

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

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, Sathya Keerthi. Support Vector Ordinal Regression. 2007.

## Попарный подход

Переход к гладкому функционалу качества ранжирования:

$$Q(w) = \sum_{i \prec j} \underbrace{[a(x_j, w) - a(x_i, w)]}_{\text{Margin}(i,j)} < 0 \\ \leq \sum_{i \prec j} \mathcal{L}(a(x_j, w) - a(x_i, w)) \rightarrow \min_w$$

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

$\mathcal{L}(M)$  — убывающая непрерывная функция отступа  $\text{Margin}(i, j)$ :

- $\mathcal{L}(M) = (1 - M)_+$  — RankSVM
- $\mathcal{L}(M) = \exp(-M)$  — RankBoost
- $\mathcal{L}(M) = \log(1 + e^{-M})$  — RankNet



## Ranking SVM: метод опорных векторов для ранжирования

Постановка задачи SVM для попарного подхода:

$$Q(w) = \frac{1}{2} \|w\|^2 + C \sum_{i < j} \underbrace{\mathcal{L}(a(x_j, w) - a(x_i, w))}_{\text{Margin}(i,j)} \rightarrow \min_w$$

где  $a(x, w) = \langle w, x \rangle$  — линейная функция ранжирования

$\mathcal{L}(M) = (1 - M)_+$  — «шарнирная» функция потерь (hinge loss)

$M = \text{Margin}(i, j) = \langle w, x_j - x_i \rangle$  — отступ

Постановка задачи квадратичного программирования:

$$\begin{cases} \frac{1}{2} \|w\|^2 + C \sum_{i < j} \xi_{ij} \rightarrow \min_{w, \xi} \\ \langle w, x_j - x_i \rangle \geq 1 - \xi_{ij}, \quad i < j \\ \xi_{ij} \geq 0, \quad i < j \end{cases}$$

## RankNet: логистическая регрессия для ранжирования

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

$$Q(w) = \sum_{i \prec j} \mathcal{L}(a_j(w) - a_i(w)) \rightarrow \min_w$$

**Метод стохастического градиента:**

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

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

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

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

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

## 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  $x_i \leftrightarrow x_j$  в списке.

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

$$w := w - \eta \sum_j \lambda_{ij} \cdot |\Delta\tilde{Q}_{ij}| \cdot \left( [i \succ j] - [i \prec j] \right) \cdot \nabla_w a_i(w)$$

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.

$|\Delta\tilde{Q}_{ij}|$  изменяет величину смещения, сохраняя его направление.

---

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

## Задача ранжирования поисковой выдачи

$D$  — коллекция текстовых 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»:

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

$Y$  — упорядоченное множество рейтингов

$y: X \rightarrow Y$  — оценки релевантности, поставленные ассессорами:  
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  $q$ :

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

## Типы признаков для ранжирования поисковой выдачи

### Типы признаков

- функции только документа  $d$
- функции только запроса  $q$
- функции запроса и документа  $(q, d)$
- текстовые
  - слова запроса  $q$  встречаются в  $d$  чаще обычного
  - слова запроса  $q$  есть в заголовках или выделены в  $d$
- ссылочные
  - на документ  $d$  много ссылаются
  - документ  $d$  содержит много полезных ссылок
- кликовые
  - на документ  $d$  часто кликают
  - на документ  $d$  часто кликают по запросу  $q$

## 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  $q = \{w_1, \dots, w_k\}$  *чисто случайно*

Оценка релевантности запроса  $q$  документу  $d$ :

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

$\text{TF}(w, d) = n_{dw}$  — 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

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

$n_d$  — длина документа  $d$

$\bar{n}_d$  — средняя длина документов в коллекции

$b \in [0, 1]$  управляет учётом длины документа (обычно  $b = 0.75$ )

$k_1 \geq 0$  ограничивает линейный рост TF (обычно  $k_1 = 2$ )

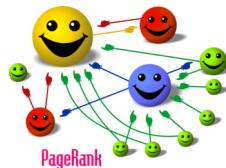
$\varepsilon$  ограничивает снизу IDF (обычно  $\varepsilon = 0$ )

---

*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 попасть на страницу  $d$ , если кликать случайно:

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

$D_d^{\text{in}} \subset D$  — множество документов, ссылающихся на  $d$ ,

$D_c^{\text{out}} \subset D$  — 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.*



## Оценивание качества поиска

Precision — доля релевантных среди найденных

Recall — доля найденных среди релевантных

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

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

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

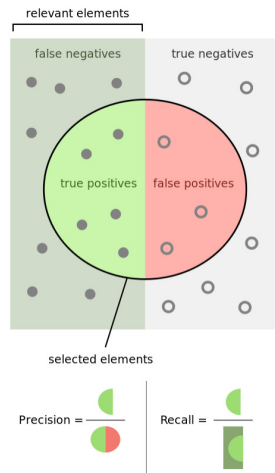
TP (true positive) — найденные релевантные

FP (false positive) — найденные нерелевантные

FN (false negative) — не найденные релевантные

TN (true negative) — не должен учитываться

Недостаток: 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  $d_q^{(n)}$ :

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

*Mean Average Precision* — AP, усреднённая по всем запросам:

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

## 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:

$$DP_n(q) = \frac{2}{n(n-1)} \sum_{i,j=1}^n [i < j] [y(q, d_q^{(i)}) < y(q, d_q^{(j)})]$$

Связь с коэффициентом ранговой корреляции ( $\tau$  Кенделла):

$$\tau(a, y) = 1 - 2 \cdot DP_n(q)$$

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

$$AUC_n(q) = \frac{1}{l_- l_+} \sum_{i,j=1}^n [y_i < y_j] [a(x_i) < a(x_j)] = 1 - \frac{n(n-1)}{2l_- l_+} DP_n(q)$$

## 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:*

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

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

*Нормированная дисконтированная сумма выигрышей:*

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

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

## Яндекс рFound — 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):

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

где  $P_i$  — вероятность дойти до  $i$ -го документа:

$$P_1 = 1;$$

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

где  $P_{out}$  — вероятность прекратить поиск без ответа

## Яндекс rFound — модель поведения пользователя

Параметры критерия rFound:

- $P_{out} = 0.15$  — 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

- Более 50 000 новых оценок ассессоров ежемесячно
- За 8 лет придумано и проверено более 2000 признаков
- Pair-wise подход лучше, чем point-wise и list-wise
- Наряду с данными ассессоров (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
- **FML** (Friendly Machine Learning): среда для тестирования алгоритмов машинного обучения, включая ранжирование

## Постановка задачи для DSSM (Deep Structured Semantic Model)

**Дано:**  $Q$  — множество запросов

$D_q^+$  — множество кликнувших документов (clickthrough data)

**Найти:** вероятностную модель релевантности документов

$$p(d|q) = \text{SoftMax}_{d \in D_q} \gamma R(q, d) = \frac{\exp(\gamma R(q, d))}{\sum_{d' \in D_q} \exp(\gamma R(q, d'))},$$

$R(q, d) = \cos(u_q, u_d)$  — косинусная близость эмбедингов  $u_q, u_d$ ;

$D_q$  содержит по 4 случайных некликнувших документа вместе с каждым кликнувшим  $d \in D_q^+$  (Negative Sampling).

**Критерий** максимума правдоподобия:

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

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



## Нейросетевой кодировщик в DSSM

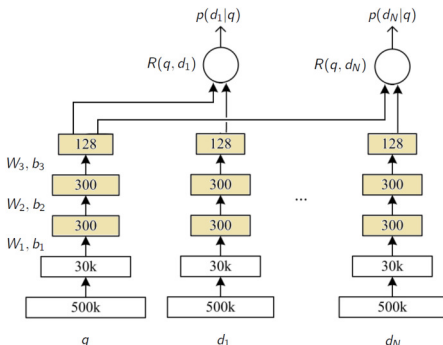
Трёхслойная *сиамская* нейронная сеть с параметрами  $\Omega = (W_1, b_1, W_2, b_2, W_3, b_3)$

$$v_d^2 = \text{th}(W_3 v_d^1 + b_3)$$

$$v_d^1 = \text{th}(W_2 v_d^0 + b_2)$$

$$v_d^0 = \text{th}(W_1 x_d + b_1)$$

$$x_d = \text{WordHash}(d)$$



**Хэширование слов (word hashing):** документ  $d$  представляется вектором частот не слов  $n_{dw}$ , а буквенных триграмм:

$\text{WordHash}(\text{дармолюб}) = \{ \_ \text{да}, \text{арм}, \text{рмо}, \text{мол}, \text{олю}, \text{люб}, \text{юб\_} \}$

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

## Преимущества DSSM

- Благодаря Word Hashing:
  - сокращается размерность 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.

---

*Tie-Yan Liu*. Learning to Rank for Information Retrieval. 2011.

*Hang Li*. A Short Introduction to Learning to Rank. 2011.