排序学习高级课程
——Pairwise与listwise方法

兰艳艳
中国科学院计算技术研究所

课程内容设置

- 排序学习简介及初级内容回顾
  - 排序学习框架
  - 排序学习评价准则
  - 单点型(pointwise)排序学习方法
- 点对型(pairwise)排序学习方法
  - RankSVM, RankBoost, RankNet
- 列表型(listwise)排序学习方法
  - ListMLE, ListNet, AdaRank
- 三类排序学习方法的比较

第一部分

排序学习简介及初级内容回顾

什么是排序学习？

排序学习（*Learning to Rank*）

研究问题　　研究方法

定义：
使用机器学习方法来研究排序问题的方式（研究方向）就称为排序学习。

(信息检索中的)排序问题

- 排序是很多实际应用的核心问题。

搜索

推荐系统

信息过滤

搜索引擎的内部机制

根据倒序表查到的相关文档池

$D=\{d_i\}$

查询词

q={中国科学院}

排序模型

文档与查询词的相关程度，以及文档（网页）本身的重要程度

文档的有序列表

传统排序方法(1)

- **相关性排序方法**(relevance ranking methods)
  - 布尔代数方法(Boolean Model)
    - 根据查询单词(term)在文档中出现的次数为基准。
    - 只能预测文档与查询是否相关，不能预测相关程度。
  - 向量模型(Vector Space Model)
    - 查询与文档均表达成欧式空间中向量的形式，通过两个向量的内积（相似性）来衡量相关程度。
    - 通常使用TF-IDF进行向量的表达，TF(Term Frequency)通过文档中term出现的频率反映相关性，IDF(Inverse Document Frequency)通过包含该term的文档的比例来反映该term的稀有或者普遍程度。举例：the brown cow

$$IDF(t) = \log \frac{N}{n(t)}$$

    - 向量表达中隐含的重要假设：term之间独立性

传统排序方法(2)

- 相关性排序方法(relevance ranking methods)
  - 隐语义索引(Latent Semantic Indexing)
    - 使用SVD分解(singular value decomposition)将原始向量空间转换到一个隐语义空间，然后在该空间中定义相似度。
  - 概率模型(Probabilistic Ranking Principle)
    - BM25 $$BM25(d,q)=\sum_{i=1}^{M}\frac{IDF(t_i)\cdot TF(t_i,d)\cdot(k_1+1)}{TF(t_i,d)+k_1\cdot(1-b+b\cdot\frac{LEN(d)}{avdl})},$$
    - 语言模型(Language Model)

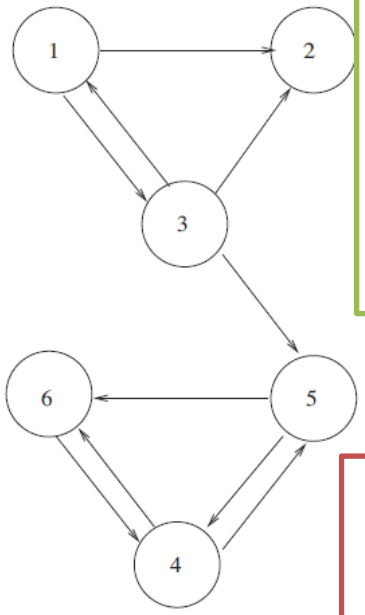
$$P(q|d)=\prod_{i=1}^{M}P(t_i|d) \qquad p(t_i|d)=(1-\lambda)\frac{TF(t_i,d)}{LEN(d)}+\lambda p(t_i|C),$$

传统排序方法(3)

- 重要性排序方法(importance ranking methods)
  - 出发点：垃圾网页，欺诈排序！(Spam)
  - PageRank $$PR(d_u) = \alpha \sum_{d_v \in B_u} \frac{PR(d_v)}{U(d_v)} + \frac{(1-\alpha)}{N},$$
  - 概率解释：Markov随机游走求遍历分布

传统排序方法(4)

- 例子： PageRank概率解释



$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$$\bar{\mathbf{P}} = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{e}^T/n = \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}$$

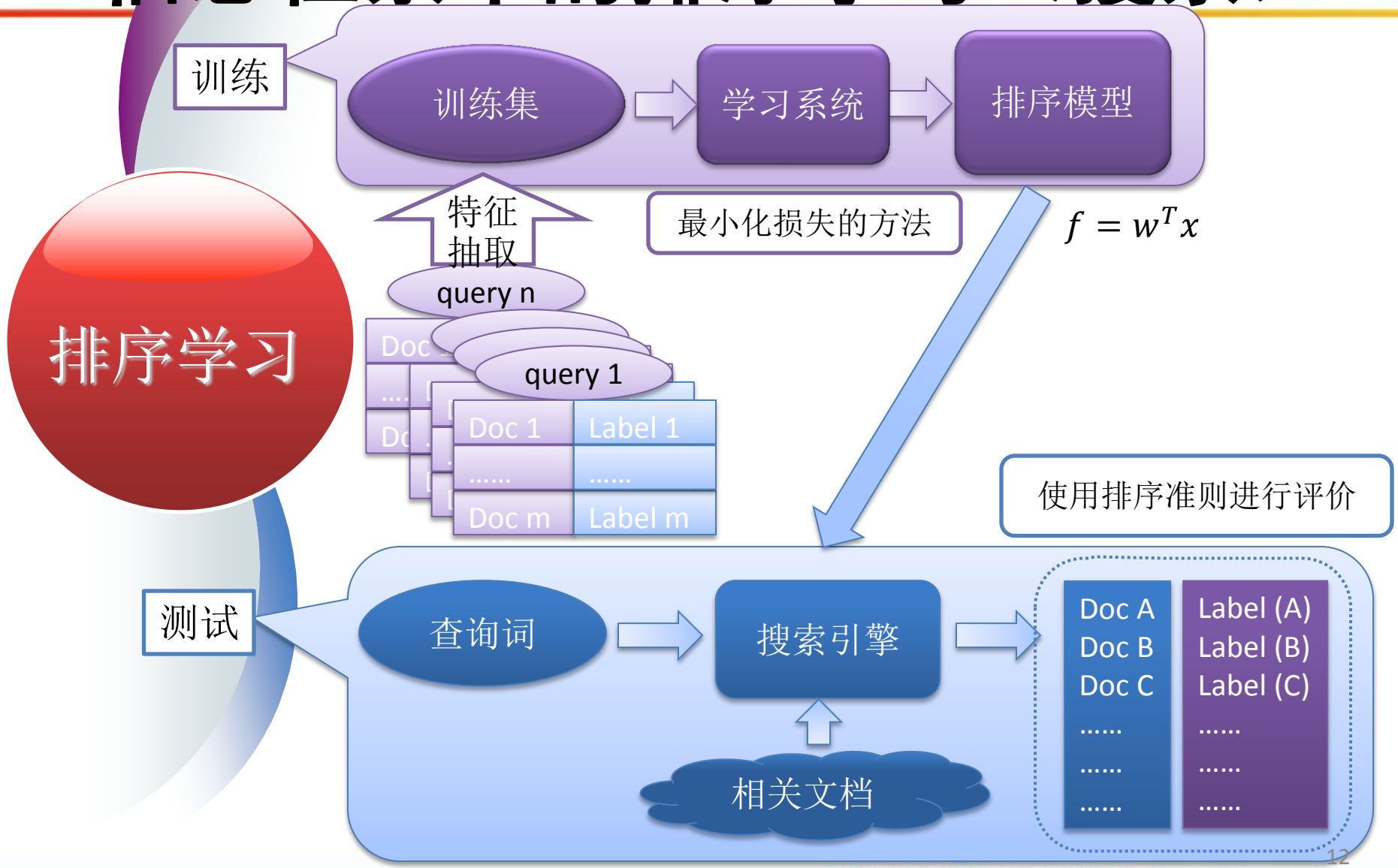
$$\pi^T\bar{\bar{\mathbf{P}}} = \pi^T$$

计算：幂方法

为什么要做排序学习？

- 两个问题：
  - 实际应用中该选择哪种方法？
    - 没有一个通用的结论，只能对特定数据集进行逐个实验，不科学，代价也较高。
  - 如何联合这些方法得到一个更好的方法？
    - 参数调节困难
    - 容易过拟合
- 机器学习可以解决问题：
  - 将每种因素(TF-IDF,BM25,PageRank)看做是一维特征
  - 参数通过学习的方式得到
  - 在不同数据上具有泛化能力

信息检索中的排序学习（搜索）



排序学习的评价准则(1)

- MAP(Mean Average Precision)
  - 针对二级标注的数据(1:相关,0:不相关)
    - P@k:前k个文档中相关文档的比例

$$P@k(\pi, l) = \frac{\sum_{t \le k} I_{\{l_{\pi^{-1}(t)}=1\}}}{k},$$

    - AP：平均的P@k

$$\mathrm{AP}(\pi, l) = \frac{\sum_{k=1}^{m} P@k \cdot I_{\{l_{\pi^{-1}(k)}=1\}}}{m_1},$$

- NDCG(Normalized Discounted Cumulative Gain)
  - 针对多级标注的数据(2:很相关, 1:相关, 0:不相关)

$$\mathrm{DCG}@k(\pi, l) = \sum_{j=1}^{k} G(l_{\pi^{-1}(j)})\eta(j), \quad G(z) = (2^z - 1), \quad \eta(j) = 1/\log(j+1)$$

$$\mathrm{NDCG}@k(\pi, l) = \frac{1}{Z_k} \sum_{j=1}^{k} G(l_{\pi^{-1}(j)})\eta(j).$$

排序学习的评价准则(2)

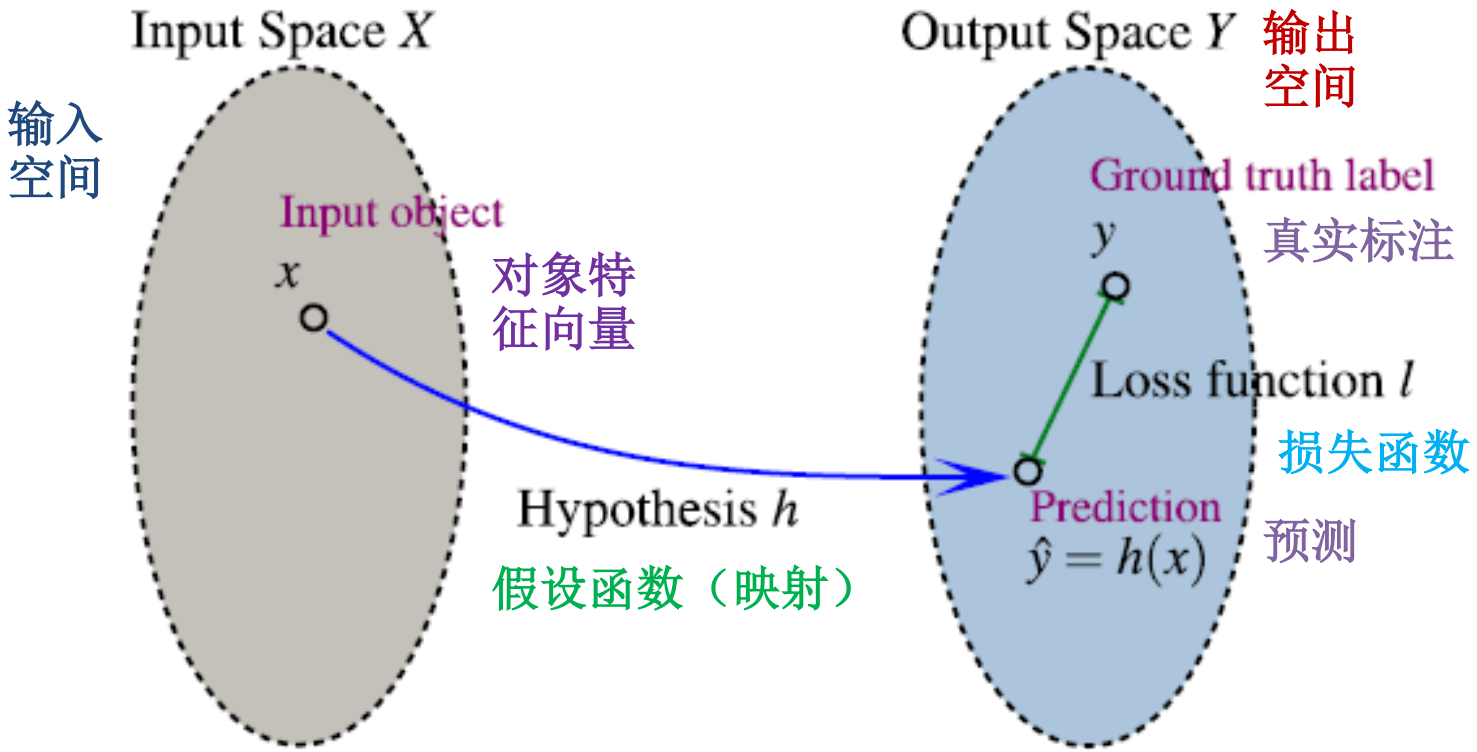
- 例子

Query = learning to rank

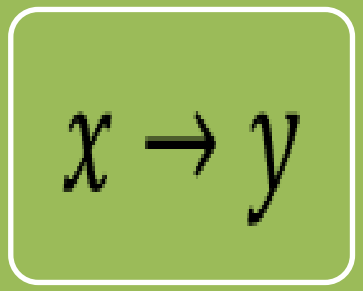
1. http://research.microsoft.com/~letor/	Relevant
2. http://www.learn-in-china.com/rank.htm	Irrelevant
3. http://web.mit.edu/shivani/www/Ranking-NIPS-05/	Relevant
... ...	

- MAP $P@1 = 1$ $P@2 = \frac{1}{2}$ $P@3 = \frac{2}{3}$ $AP = \frac{1}{2}(1 + \frac{2}{3}) = \frac{5}{6}$
- NDCG $DCG@3 = 1.5$ $Z_3 = 1.63$ $NDCG@3 = \frac{1.5}{1.63} = 0.92$

机器学习的基本框架

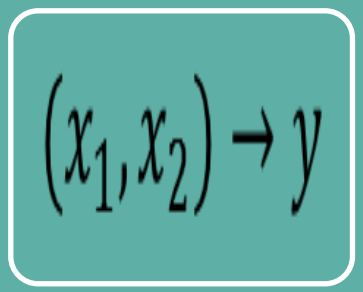


排序学习的基本方法



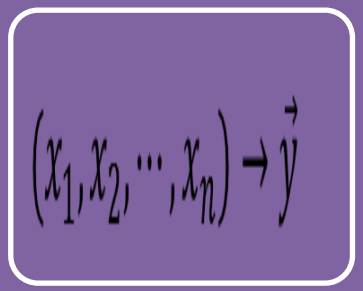
单点型(pointwise)排序学习算法

- 以单个文档为对象的回归，分类或序回归问题
- OC SVM, McRank



点对型(pairwise)排序学习算法

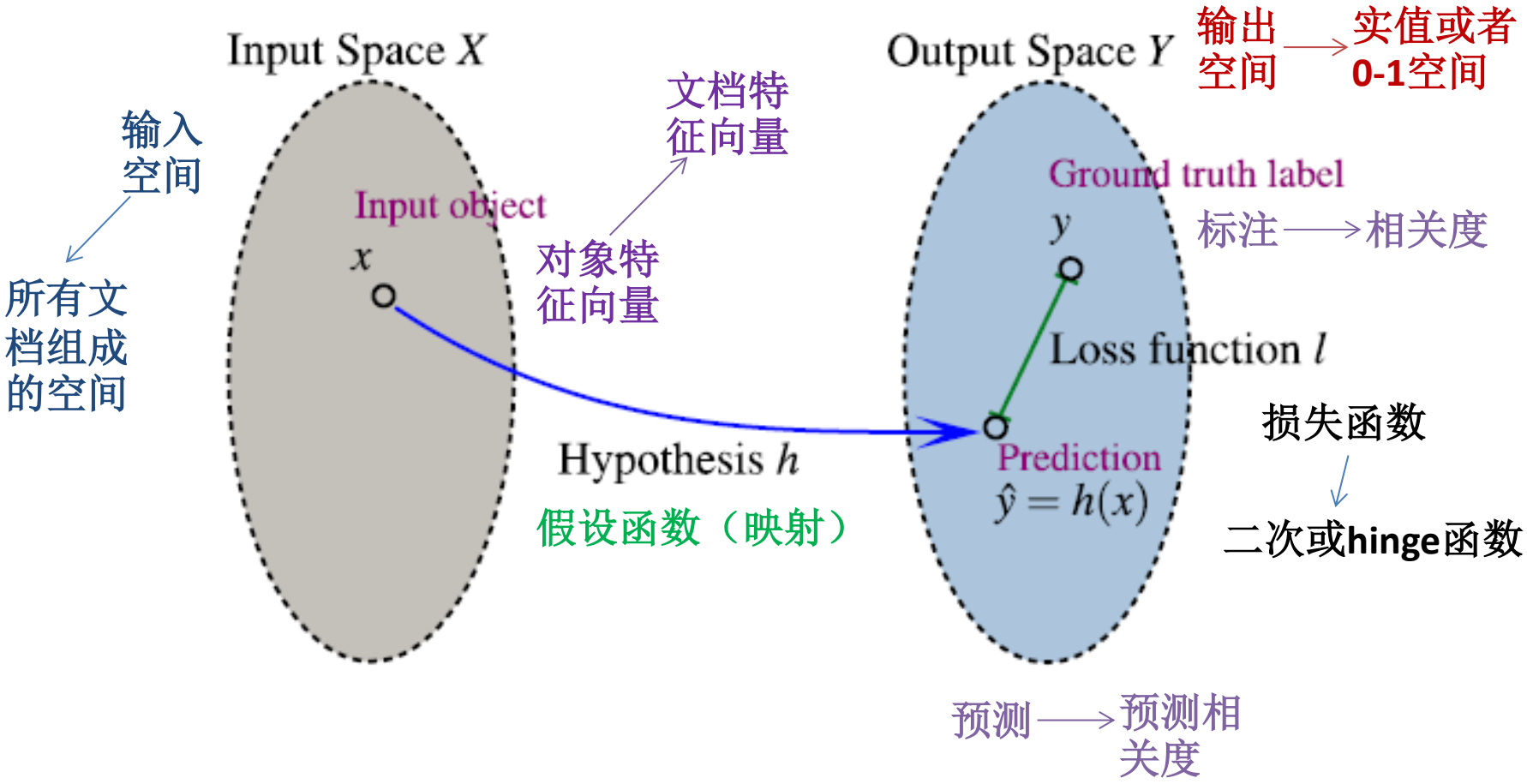
- 以点对为对象的分类问题
- RankSVM，RankBoost，RankNet，GBRank



列表型(listwise)排序学习算法

- 以列表为对象的排序问题
- ListMLE，ListNet，RankCosine，StructureSVM，SoftRank，AdaRank

单点型(pointwise)排序学习方法(1)

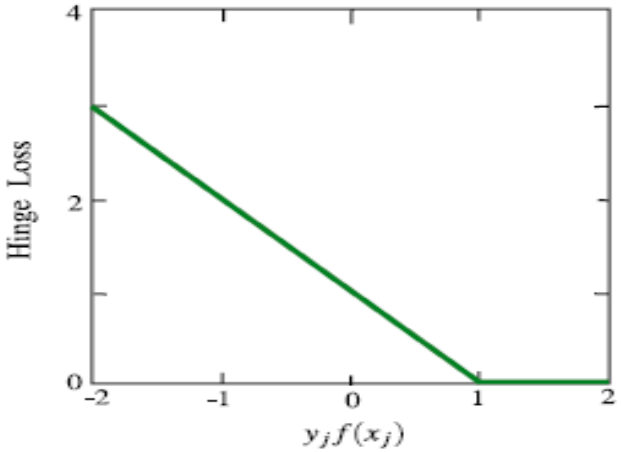


单点型(pointwise)排序学习方法(2)

- 回归模型
- 分类模型
  - 支持向量机(Support Vector Machine)

$$L(f; x_j, y_j) = (y_j - f(x_j))^2.$$

$$L(f; x_j, y_j) = (1 - y_j w^T x_j)_+$$



$$\min \frac{1}{2}\|w\|^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{m^{(i)}} \xi_j^{(i)}$$

$$\text{s.t.} \quad w^T x_j^{(i)} \le -1 + \xi_j^{(i)}, \quad \text{if } y_j^{(i)} = 0.$$

$$w^T x_j^{(i)} \ge 1 - \xi_j^{(i)}, \quad \text{if } y_j^{(i)} = 1.$$

$$\xi_j^{(i)} \ge 0, \quad j = 1, \ldots, m^{(i)}, i = 1, \ldots, n,$$

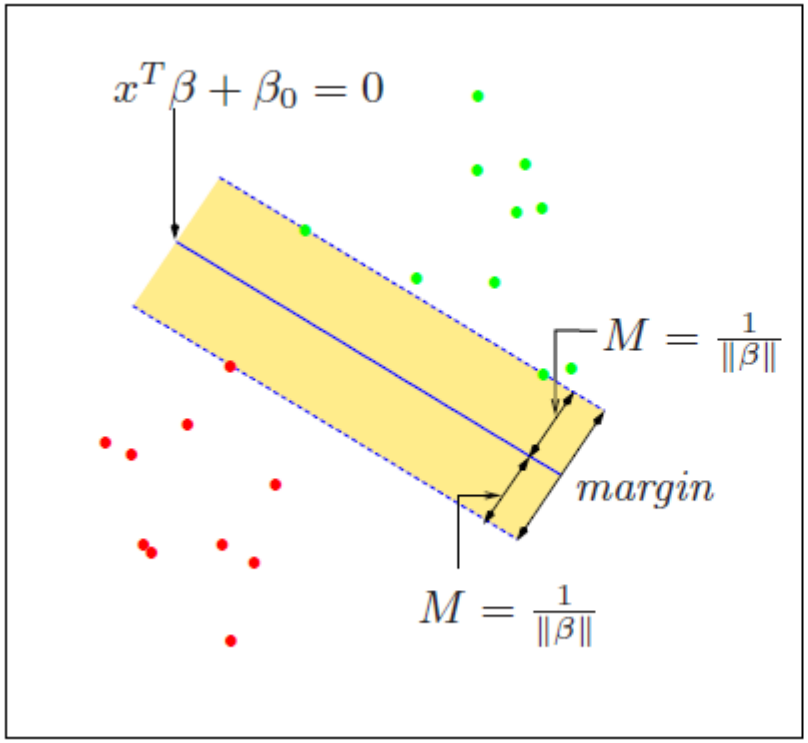
  - 逻辑回归(Logistic Regression)

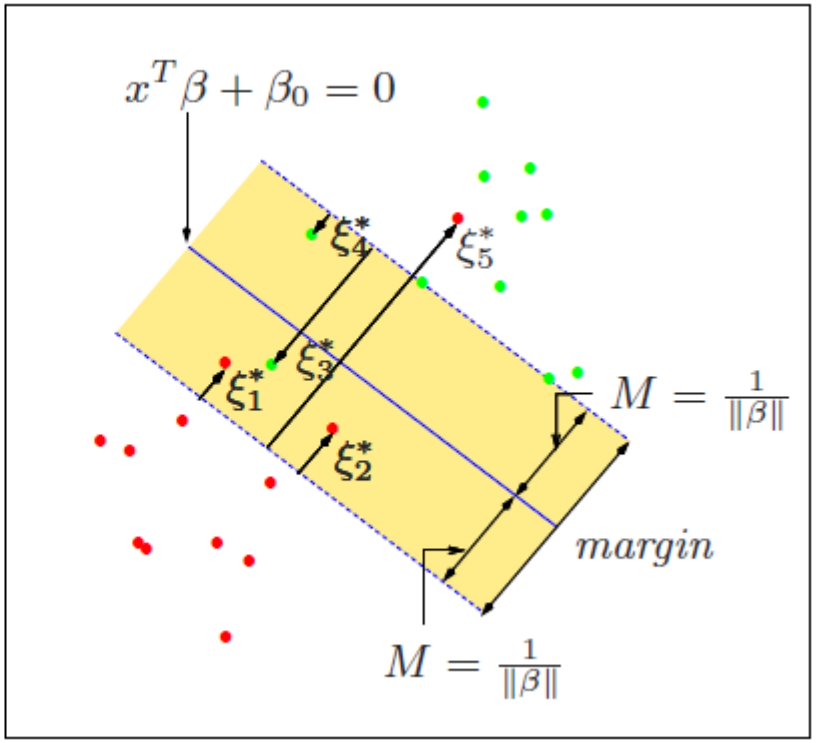
$$\log\left(\frac{P(R|x_j)}{1 - P(R|x_j)}\right) = c + \sum_{t=1}^{T} w_t x_{j,t}$$

求解：极大似然估计 $P(R|x_j) = \dfrac{1}{1 + e^{-c - \sum_{t=1}^{T} w_t x_{j,t}}}.$

SVM(Support Vector Machine)

- SVM的几何解释





单点型(pointwise)排序学习方法(3)

- 优点：
  - 直观，简单
  - 直接使用已有知识或算法来解决排序问题
- 缺点：
  - 仅考虑单个文档的相关度，而文档间的相对相关性没办法在学习过程中体现。
  - 学习过程与评价准则不符
    - 以单个文档为对象，不符合查询级别的评价
    - 文档的位置信息在算法中没有体现

接下来……

点对型(pairwise)排序学习方法

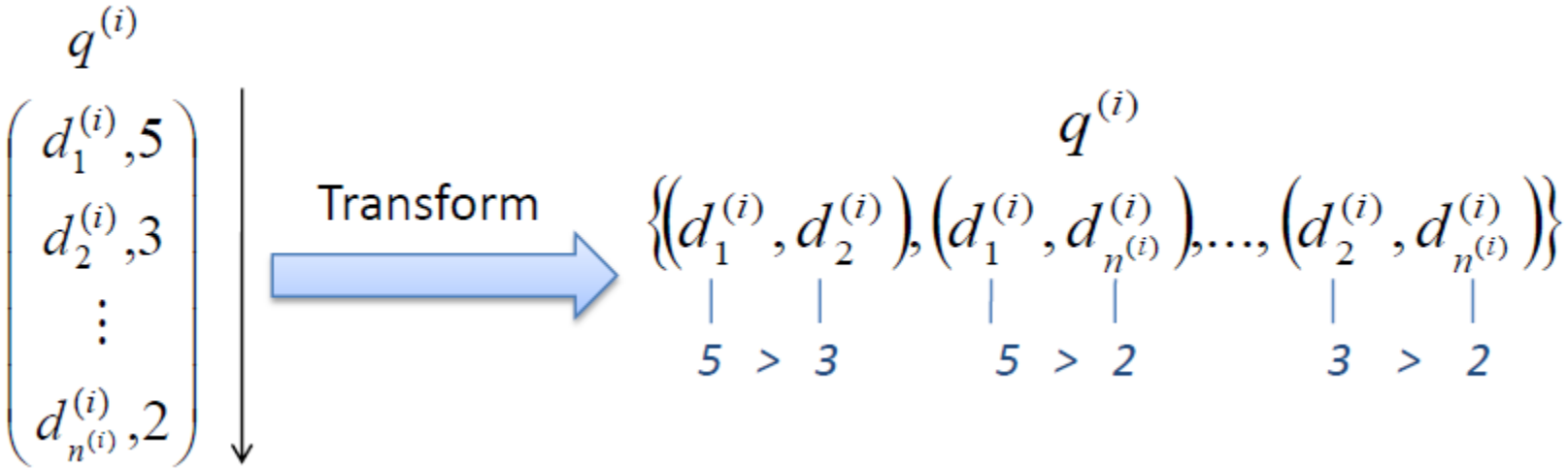
Break

第二部分

点对型(PAIRWISE)排序学习方法

Pairwise方法的核心思想(1)

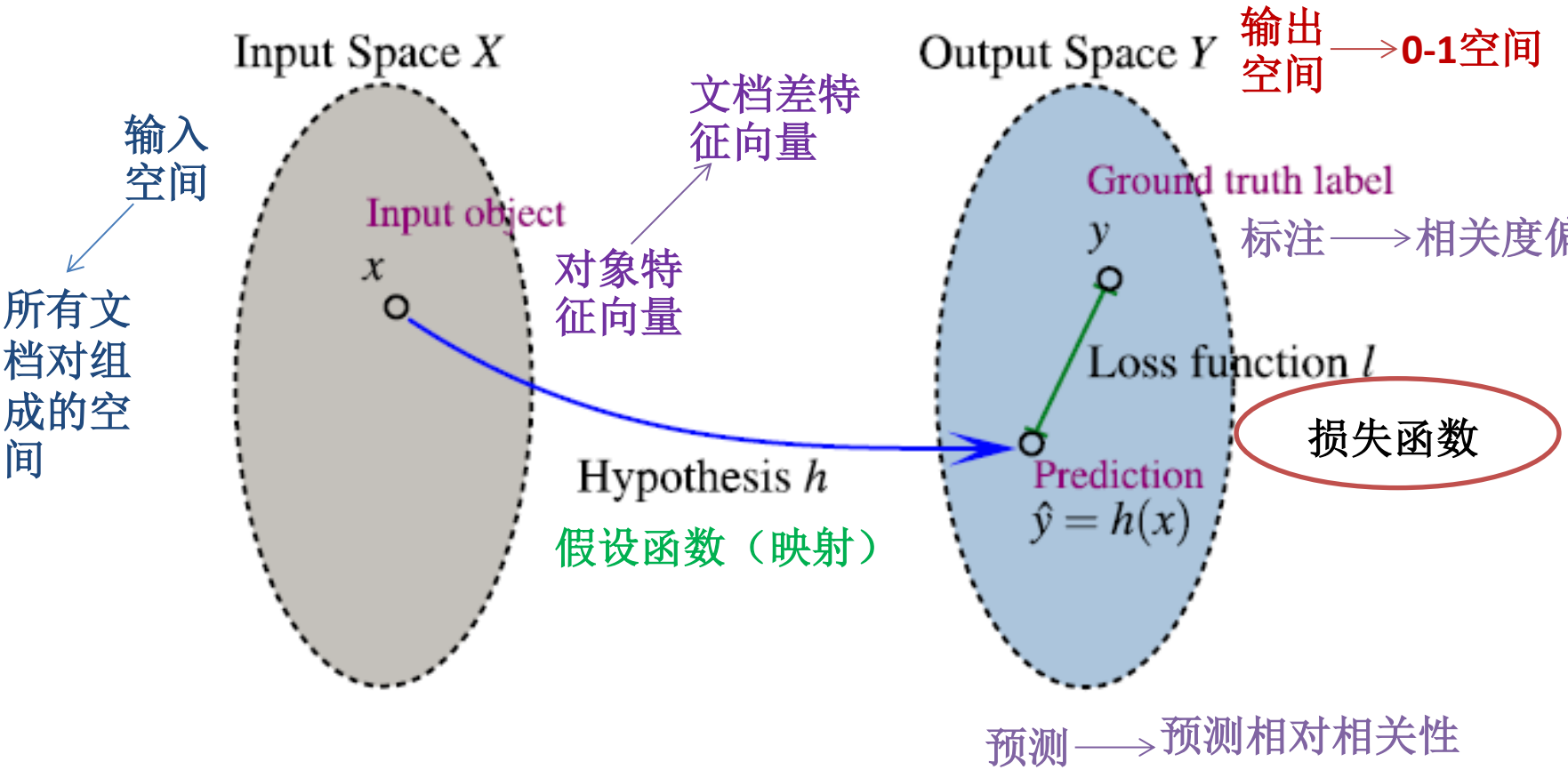
排序 ⇒ 两两文档之间的序



Pairwise方法的核心思想(1)

- 以两个文档形成的文档对为研究对象
- 以两个文档间的相对相关度（偏好顺序）为研究目标
- 将排序问题转化成为文档对上的分类问题
- 学习目标即是最小化训练集合中所有文档对上的分类错误率

点对型(pairwise)排序学习方法概述



RankSVM(1)

- 直接用支持向量机方法来进行两个文档上的分类(pairwise classification)

$$\min \frac{1}{2}\|w\|^2 + \lambda \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} \xi_{u,v}^{(i)}$$

$$\text{s.t.} \quad w^T\left(x_u^{(i)} - x_v^{(i)}\right) \geq 1 - \xi_{u,v}^{(i)}, \text{ if } y_{u,v}^{(i)} = 1,$$

$$\xi_{u,v}^{(i)} \geq 0, \quad i = 1, \ldots, n.$$

$$\min \frac{1}{2}\|w\|^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{m^{(i)}} \xi_j^{(i)}$$

$$\text{s.t} \quad w^T x_j^{(i)} \leq -1 + \xi_j^{(i)}, \quad \text{if } y_j^{(i)} = 0.$$

$$w^T x_j^{(i)} \geq 1 - \xi_j^{(i)}, \quad \text{if } y_j^{(i)} = 1.$$

$$\xi_j^{(i)} \geq 0, \quad j = 1, \ldots, m^{(i)}, i = 1, \ldots, n,$$

损失函数

$$L(f; x_u, x_v, y_{uv}) = \left(1 - y_{uv} w^T (x_u - x_v)\right)_+$$

$$L(f; x_j, y_j) = \left(1 - y_j w^T x_j\right)_+$$

Hinge Loss

RankSVM(2)

- RankSVM的几个优点：
  - 基于SVM的最大化margin的框架，具有良好的泛化能力
  - 基于SVM的框架，便于进行核函数操作，便于处理非线性情形
  - 具有快速实现的算法，在线算法包括 http://olivier.chapelle.cc/primal/和 http://www.cs.cornell.edu/People/tj/svm_light/svm_rank.html

RankBoost

- 用AdaBoost方法来进行两个文档上的分类(pairwise classification)

Algorithm 1: Learning algorithm for RankBoost

Input: training data in terms of document pairs

Given: initial distribution $\mathcal{D}_1$ on input document pairs.

For $t = 1, \ldots, T$

Train weak ranker f_t based on distribution $\mathcal{D}_t$.

Choose α_t

Update $\mathcal{D}_{t+1}(x_u^{(i)}, x_v^{(i)}) = \frac{1}{Z_t}\mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$

where $Z_t = \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} \mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$.

Output: $f(x) = \sum_t \alpha_t f_t(x)$.

RankBoost中参数α_t的选择

- 依据一个定理 $\mathrm{rloss}_D(H) \le \prod_{t=1}^{T} Z_t$
- α_t选择的策略是最小化Z_t
  - 对任何给f_t，均可证明Z_t具有唯一最小解，可通过数值方法得到，例如线搜索方法。
  - 特别的，若函数f_t取值为0,1，则可直接得到解析解

$$\alpha_t = \frac{1}{2}\log\left(\frac{W_{t,-1}}{W_{t,+1}}\right). \quad W_{t,b} = \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} \mathcal{D}_t\left(x_u^{(i)}, x_v^{(i)}\right) I_{\{f_t(x_u^{(i)}) - f_t(x_v^{(i)}) = b\}}.$$

  - 对于取值在[0,1]中f_t，可最小化Z_t的近似值得到最优解

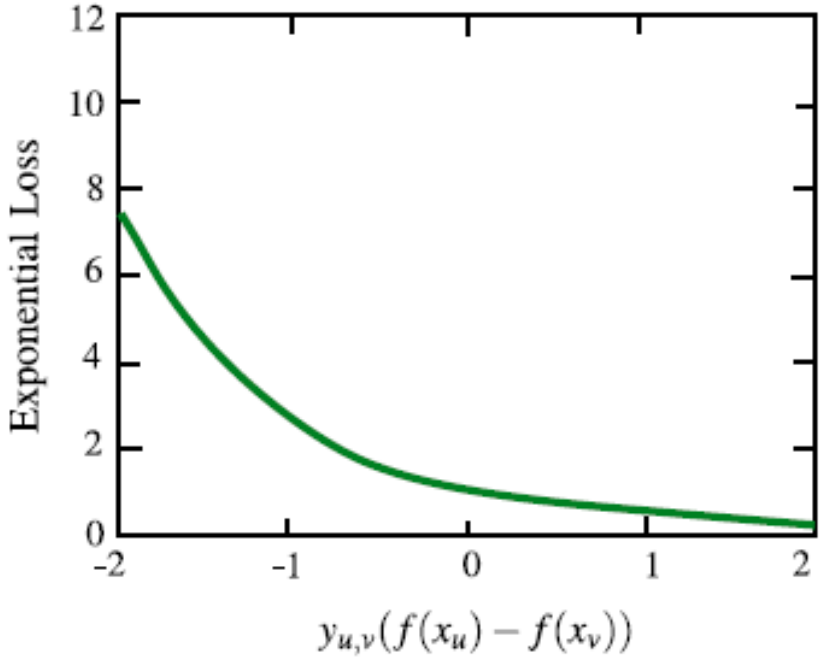
$$\alpha_t = \frac{1}{2}\log\left(\frac{1+r_t}{1-r_t}\right). \quad r_t = \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} \mathcal{D}_t\left(x_u^{(i)}, x_v^{(i)}\right)\left(f_t\left(x_u^{(i)}\right) - f_t\left(x_v^{(i)}\right)\right),$$

RankBoost的损失函数

- 指数损失函数(Exponential Loss)

$$L(f; x_u, x_v, y_{uv}) = \exp\{-y_{uv}(f(x_u) - f(x_v))\}$$

- 损失函数形状



RankNet(1)

- 损失函数：交叉熵(cross entropy)

$$L(f; x_u, x_v, y_{u,v}) = -\bar{P}_{u,v} \log P_{u,v}(f) - (1 - \bar{P}_{u,v}) \log\big(1 - P_{u,v}(f)\big).$$

$$\bar{P}_{u,v} = 1, \text{if } y_{u,v} = 1; \quad \bar{P}_{u,v} = 0$$

$$P_{u,v}(f) = \frac{\exp(f(x_u) - f(x_v))}{1 + \exp(f(x_u) - f(x_v))}.$$

根据标注估计的目标概率分布

根据打分函数估计的建模概率分布

RankNet的损失函数衡量的根据打分函数估计的概率分数与真实标注建模的概率分布之间的距离

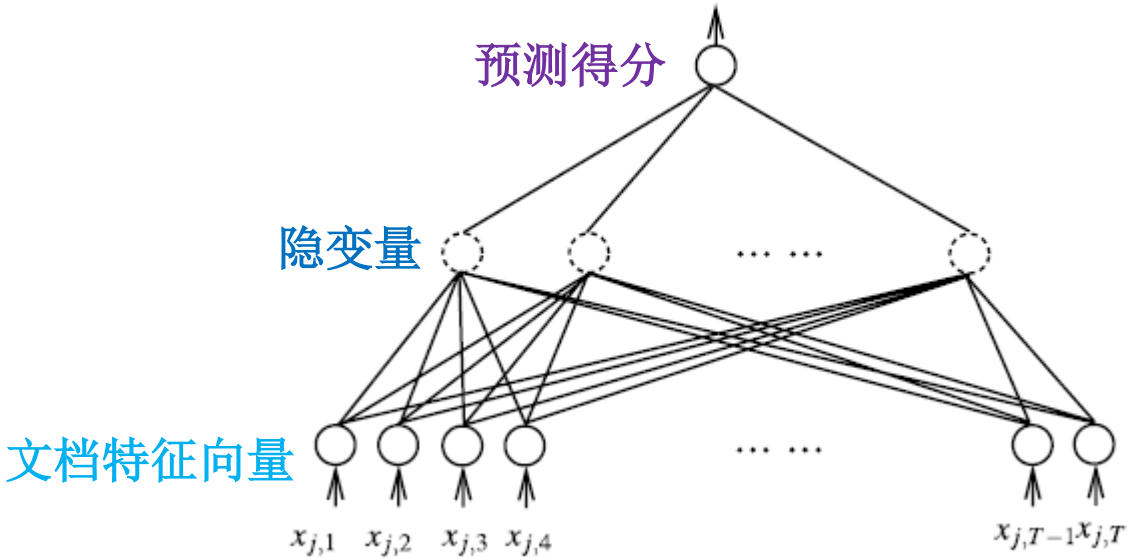
$$L(f; x_u, x_v, y_{u,v}) = -\bar{P}_{u,v} \log P_{u,v}(f) - (1 - \bar{P}_{u,v}) \log\big(1 - P_{u,v}(f)\big).$$

Pairwise的logistic regression

Logistic损失

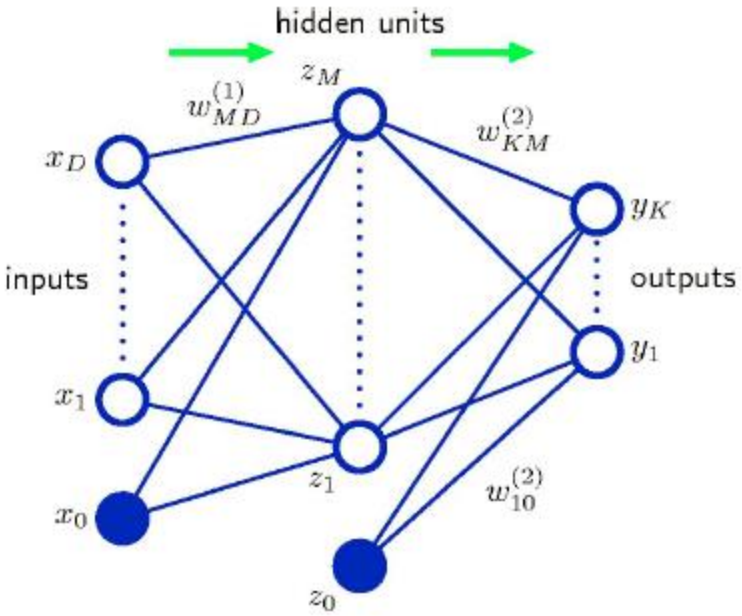
RankNet(2)

- 为什么叫做RankNet?



首先通过一个神经网络(neural network)来表达打分函数得到的预测得分，然后进入损失函数中进行梯度下降求解。

A Neural Network



Can be viewed as a generalization of linear models

$$y_k(\mathbf{x},\mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$

Pairwise方法小结(1)

- 目标是一致的，损失函数都是pairwise0-1损失的上界

$$L_{0-1}(f; x_u, x_v, y_{u,v}) = \begin{cases} 1, & y_{u,v}(f(x_u) - f(x_v)) < 0, \\ 0, & \text{otherwise.} \end{cases}$$

- 逼近的程度是不同的（不同的损失函数）

SVM → RankSVM

$$L(f; x_u, x_v, y_{uv}) = \left(1 - y_{uv} w^T (x_u - x_v)\right)_+$$

Hinge损失

AdaBoost → RankBoost

$$L(f; x_u, x_v, y_{uv}) = \exp\{-y_{uv}(f(x_u) - f(x_v))\}$$

指数损失

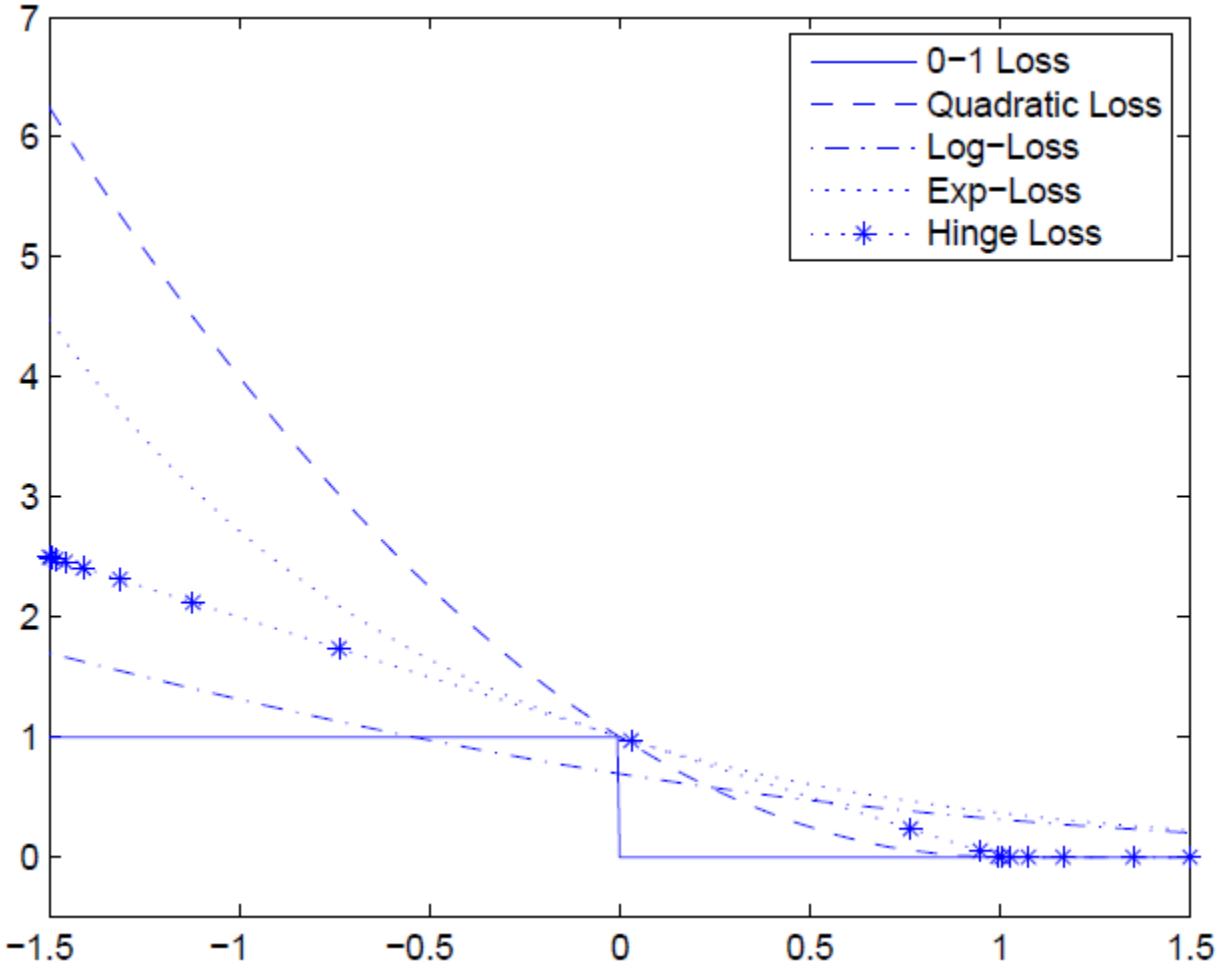
RankNet

$$L(f; x_u, x_v, y_{uv}) = -\log \frac{1}{1 + \exp\{-y_{uv}(f(x_u) - f(x_v))\}}$$

对数损失

Logistic Regression

不同损失函数比较



Pairwise方法小结(2)

- 优点：
  - 简单，直观，容易理解
  - 直接建模相对相关性，比pointwise方法更好的抓住了排序的本质
  - 效果好，被主流搜索引擎所使用
- 缺点：
  - 学习过程与评价准则不符
    - 以单个文档对为对象，不符合查询级别的评价
    - 文档的位置信息在算法中没有体现
  - 将排序打散成文档有序对和原始排序有差别
  - 文档对众多，运算慢

Pairwise方法的问题(1)

- 不同查询间的文档个数通常差异较大，导致pairwise方法的目标与优化准则偏离。
- 例子：
  - 两个查询
  - Pairwise错误率

  与查询级别准则对不同函数的评价有很大差别

		Case 1	Case 2
Document pairs of q_1	correctly ranked	770	780
	wrongly ranked	10	0
	Accuracy	98.72%	100%
Document pairs of q_2	correctly ranked	10	0
	wrongly ranked	0	10
	Accuracy	100%	0%
overall accuracy	document level	98.73%	98.73%
	query level	99.36%	50%

Pairwise方法的问题(2)

- 无法衡量不同位置上的pairwise错误率对结果的影响
  - 在pairwise分类错误率下是同等的
  - 然而在与位置相关的评价准则下则是不同的
- 例子：

p: *perfect*, g: *good*, b: *bad*

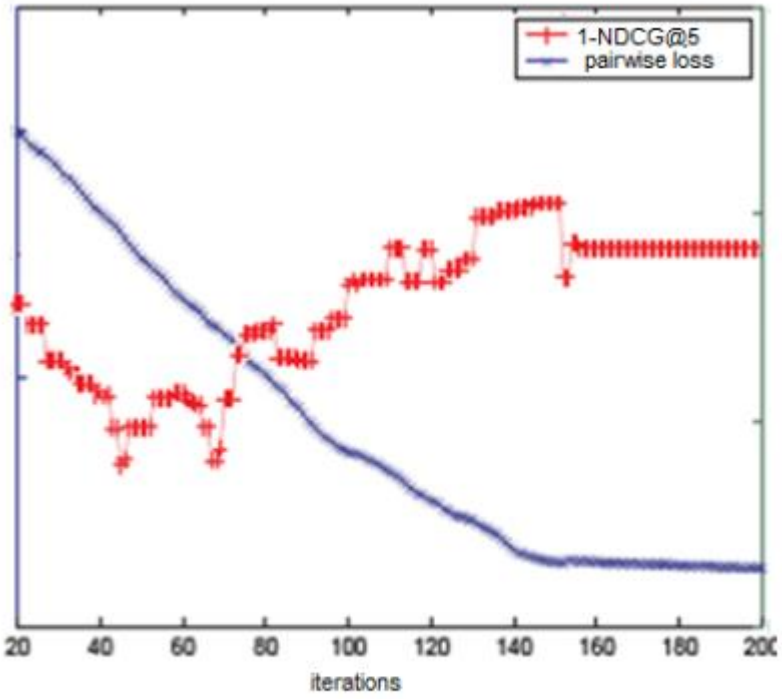
Ideal: p g g b b b b

ranking 1: g p g b b b b one wrong pair **Worse**

ranking 2: p g b g b b b one wrong pair **Better**

Pairwise方法的问题(3)

- 前者所述两个问题带来的直接后果：
  pairwise分类错误率与
  评价准则存在严重偏差



Pairwise loss vs. (1-NDCG@5)
TREC Dataset

解决问题的方式(1)

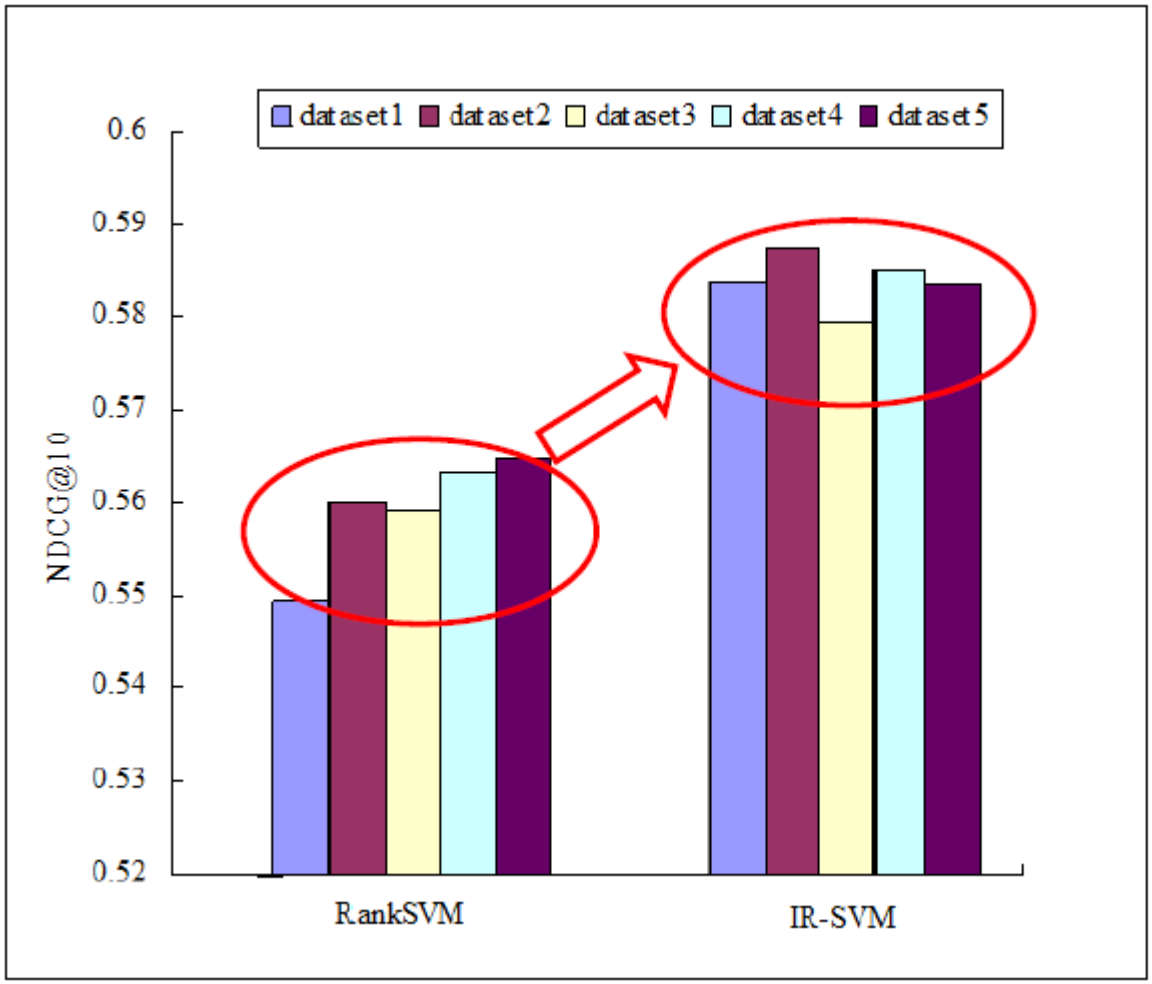
- 在优化目标中对于不同查询加上文档pair数的归一化因子进行运算

$$\min_{\vec{w}} L(\vec{w}) = \sum_{i=1}^{l} \mu_{q(i)} \left[1 - z_i \left\langle \vec{w}, \vec{x}_i^{(1)} - \vec{x}_i^{(2)} \right\rangle \right]_+ + \lambda \left\| \vec{w} \right\|^2$$

Query-level normalizer

$$\mu_{q(i)} = \frac{\max_j \#\{\text{instance pairs associated with } q(j)\}}{\#\{\text{instance pairs associated with } q(i)\}}$$

解决问题的方式(1)实验效果



Web Data

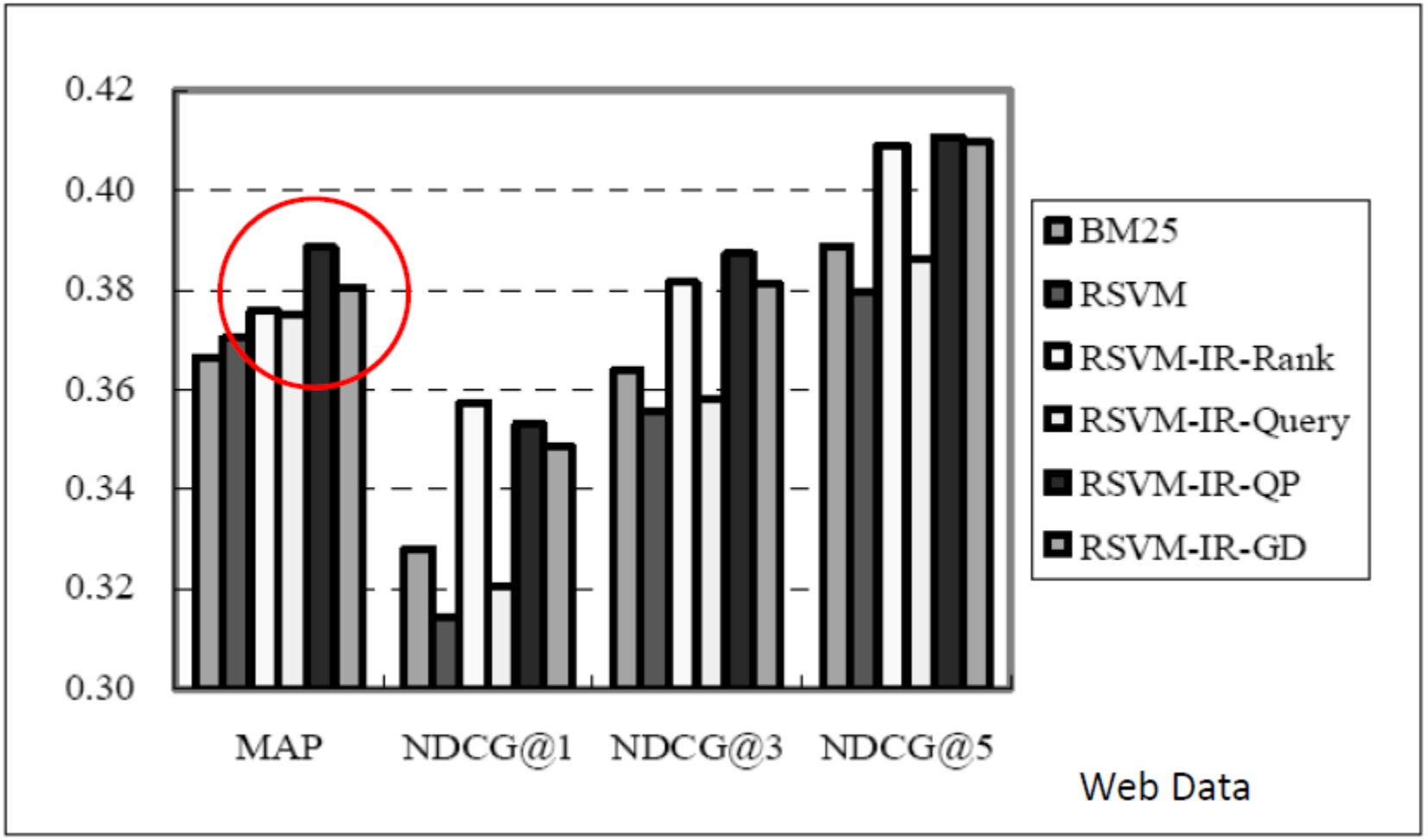
解决问题的方式(2)

- 在重点位置上(top)加入更强的权重进行优化学习

$$\min_{\vec{w}} L(\vec{w}) = \sum_{i=1}^{l} \tau_{k(i)} u_{q(i)} \left[1 - z_i \left\langle \vec{w}, \vec{x}_i^{(1)} - \vec{x}_i^{(2)} \right\rangle \right]_+ + \lambda \left\| \vec{w} \right\|^2$$

Implicit Position Discount:
Larger weight for more critical type of pairs
Learned from a labeled set

解决问题的方式(2)实验效果



接下来.....

列表型(listwise)排序学习方法

Break

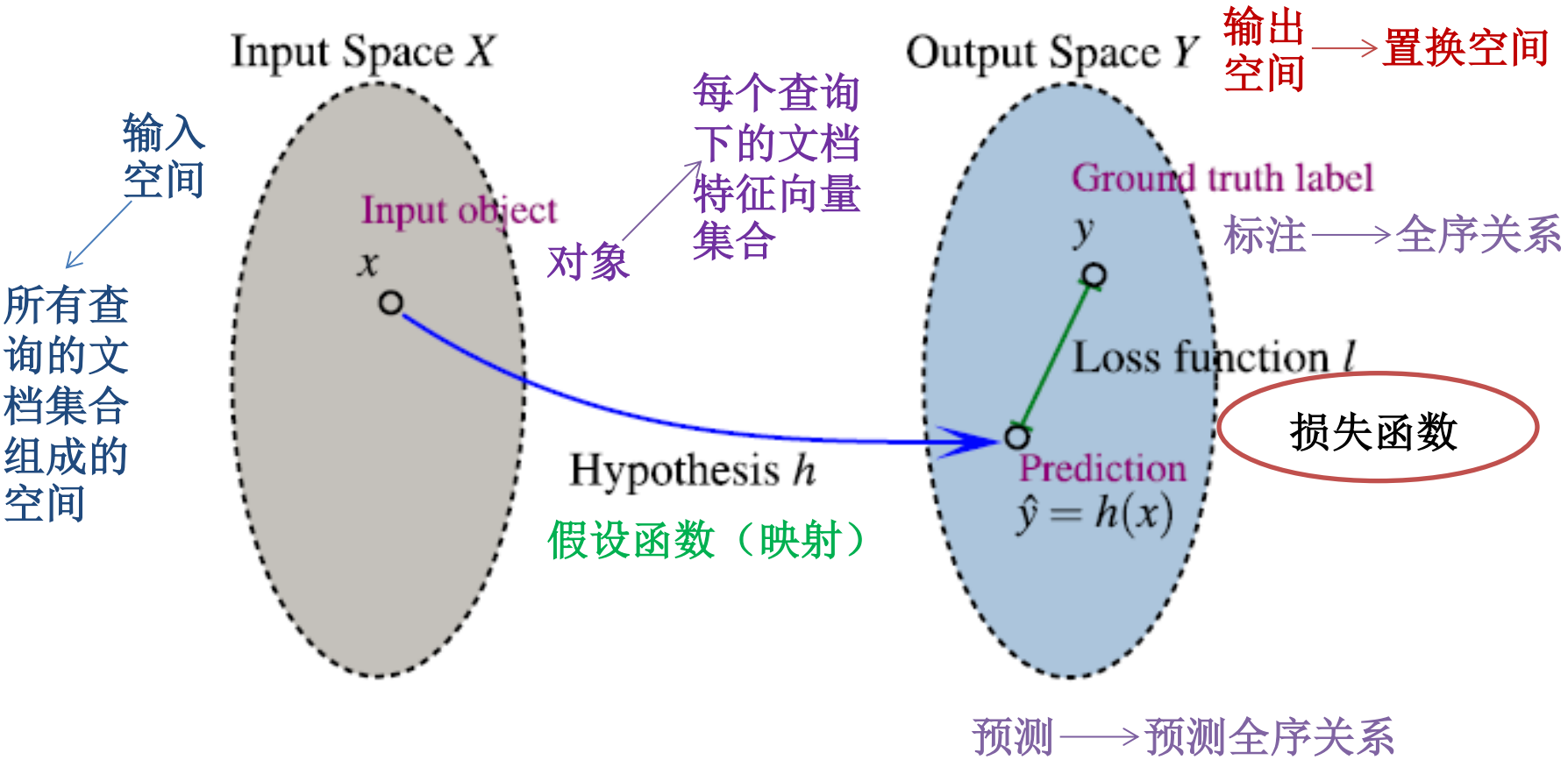
第三部分

列表型(LISTWISE)排序学习方法

listwise方法的核心思想

- 以每个查询下所有文档的集合为研究对象
- 以文档的全序关系(permutation)为研究目标
- 将排序问题转化成为所有文档与其全序关系的映射问题(permutation prediction)
- 学习目标是最小化训练集合中所有查询上的预测错误率

列表型(listwise)排序学习方法概述



两类列表型排序学习方法

- 优化替代损失的列表型排序学习方法
  - ListMLE，ListNet，StrctRank，BoltzRank
- 直接优化评价准则的列表型排序学习方法
  - 优化评价准则的一个连续可导的近似函数
    - SoftRank, AppRank, SmoothRank
  - 优化评价准则的一个连续可到的上界
    - SVMmap，SVMNDCG，PermuRank
  - 使用优化技巧直接优化评价准则
    - AdaRank，RankGP

ListMLE与ListNet的初衷(1)

- 以往众多距离的衡量并不尽如人意
- 一个简单的例子：
  - 函数f　$f(A)=3,\ f(B)=0,\ f(C)=1$　ACB
  - 函数h　$h(A)=4,\ h(B)=6,\ h(C)=3$　BAC
  - 标注g　$g(A)=6,\ g(B)=4,\ g(C)=3$　ABC

  对应列表
  - 问题：f和h哪个函数更好（更接近g）？
    - 单点型：f<h
    - 点对型：f=h
    - Cosine距离：f<h
    - NDCG等准则：f>h

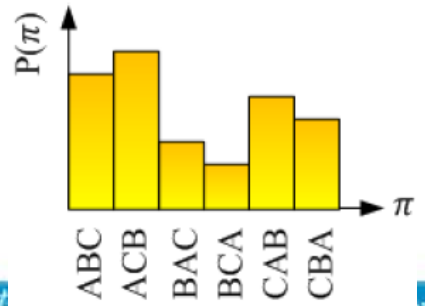
矛盾！

ListMLE与ListNet的初衷(2)

- 建模的方式不合理！
  - 单点型：单个的得分
  - 点对型：两两偏好关系
  - Cosine距离：分数形成的向量
- 如何建模一个排序以及得分？

将每个排序根据他的分数
建模成为一个概率

f: $f(A)=3, f(B)=0, f(C)=1$;
Ranking by f: ABC



Plackett-Luce概率模型

- 建模了对给定一个文档集合，其全序关系的产生概率

$$P(\pi|\mathbf{s}) = \prod_{j=1}^{m} \frac{\varphi(s_{\pi^{-1}(j)})}{\sum_{u=1}^{m} \varphi(s_{\pi^{-1}(u)})},$$

- 例子

$$P_f(\mathrm{ABC}) = \frac{\varphi(f(\mathrm{A}))}{\varphi(f(\mathrm{A}))+\varphi(f(\mathrm{B}))+\varphi(f(\mathrm{A}))} \cdot \frac{\varphi(f(\mathrm{B}))}{\varphi(f(\mathrm{B}))+\varphi(f(\mathrm{C}))} \cdot \frac{\varphi(f(\mathrm{C}))}{\varphi(f(\mathrm{C}))}$$

P(A ranked No.1)

P(B ranked No.2 | A ranked No.1)
= P(B ranked No.1)/(1- P(A ranked No.1))

P(C ranked No.3 | A ranked No.1, B ranked No.2)

Plackett-Luce模型的性质

- 连续，可微并关于得分是凸函数
- 若得分与标注得到的序一致，任意更换两个文档的顺序，则概率都将下降。
- 如果ψ是指数函数，那么具有平移不变性

$$P_s(\pi) = \prod_{j=1}^{n} \frac{\phi(s_{\pi(j)})}{\sum_{k=j}^{n} \phi(s_{\pi(k)})} = P_{\lambda+s}(\pi) = \prod_{j=1}^{n} \frac{\phi(\lambda + s_{\pi(j)})}{\sum_{k=j}^{n} \phi(\lambda + s_{\pi(k)})}$$

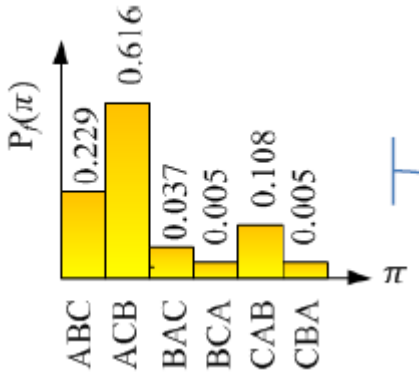
- 如果ψ是线性函数，那么具有尺度不变性

$$P_s(\pi) = \prod_{j=1}^{n} \frac{\phi(s_{\pi(j)})}{\sum_{k=j}^{n} \phi(s_{\pi(k)})} = P_{\lambda s}(\pi) = \prod_{j=1}^{n} \frac{\phi(\lambda s_{\pi(j)})}{\sum_{k=j}^{n} \phi(\lambda s_{\pi(k)})}$$

两个概率分布之间的距离：K-L

- 例子

f: $f(\mathrm{A}) = 3, f(\mathrm{B})=0, f(\mathrm{C})=1;$ ↔
Ranking by f: ABC

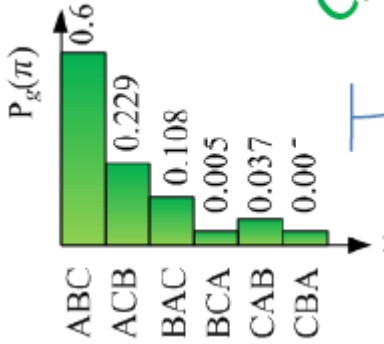


K-L距离度量下

Closer!

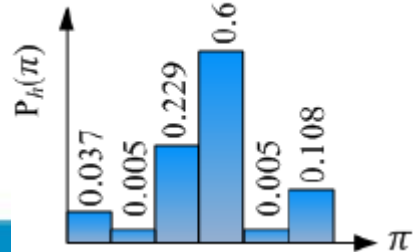
$dis(f,g) = 0.46$

g: $g(\mathrm{A}) = 6, g(\mathrm{B})=4, g(\mathrm{C})=3;$ ↔
Ranking by g: ABC



$dis(g,h) = 2.56$

h: $h(\mathrm{A}) = 4, h(\mathrm{B})=6, h(\mathrm{C})=3;$ ↔
Ranking by h: ACB



ListNet(1)

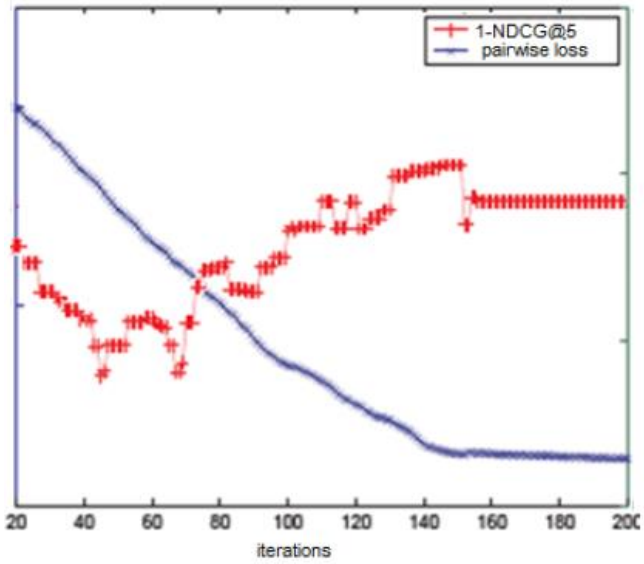
- 基于P-L模型，使用K-L Divergence作为损失函数进行学习

损失函数 $L(f;\mathbf{x},\Omega_y)=D\left(P_y(\pi)\middle\|P\left(\pi\middle|\left(f(w,\mathbf{x})\right)\right)\right)$

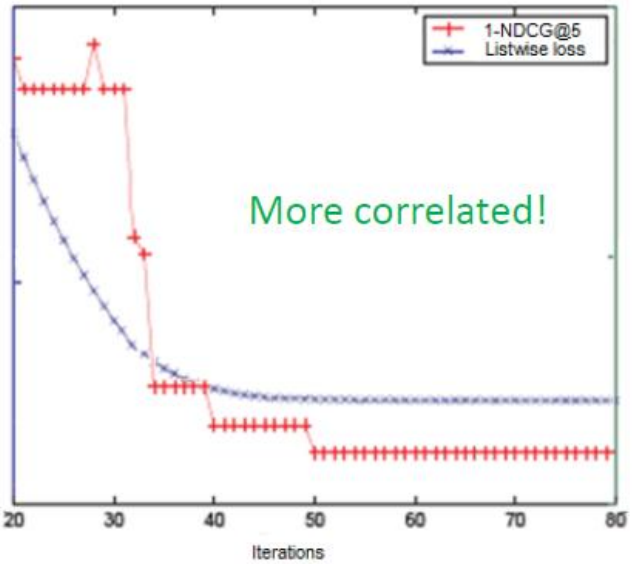
- Net：使用神经网络为模型
- 求解：目标是凸函数，使用梯度下降求解。
- 局限：计算过于复杂！文档数的指数级别！
- Top-k ListNet：在计算K-L距离时只算前k个上的概率分布，然而仍然计算比较复杂！实际中通常只用Top-1的ListNet进行计算。

ListNet(2)

- 优点：纠正了pairwise方法的缺陷，使得损失更加符合评价准则



Pairwise (RankNet)



Listwise (ListNet)

ListMLE

- 基于P-L模型，使用极大似然方法进行学习

损失函数 $$L(f;\mathbf{x},\pi_y) = -\log P\left(\pi_y|f(w,\mathbf{x})\right).$$ 似然损失 (likelihood loss)

- 与ListNet相比，大大降低了复杂程度，在实际应用中非常高效。

AdaRank

- 将评价准则纳入到Boosting框架中进行优化

Algorithm 1: Learning algorithms for AdaRank

Input: the set of documents associated with each query

Given: initial distribution $\mathcal{D}_1$ on input queries

For $t = 1, \ldots, T$

Train weak ranker $f_t(\cdot)$ based on distribution $\mathcal{D}_t$.

Choose $\alpha_t = \frac{1}{2} \log \frac{\sum_{i=1}^{n} \mathcal{D}_t(i)(1+M(f_t, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}))}{\sum_{i=1}^{n} \mathcal{D}_t(i)(1-M(f_t, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}))}$

Update $\mathcal{D}_{t+1}(i) = \frac{\exp(-M(\sum_{s=1}^{t} \alpha_s f_s, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}))}{\sum_{j=1}^{n} \exp(-M(\sum_{s=1}^{t} \alpha_s f_s, \mathbf{x}^{(j)}, \mathbf{y}^{(j)}))}$.

Output: $\sum_t \alpha_t f_t(\cdot)$.

AdaRank的收敛性

- 一个定理标明只要 $e^{-\delta_{min}^t}\sqrt{1-\varphi(t)^2}<1$ ，训练误差将会随时间下降

THEOREM 1. *The following bound holds on the ranking accuracy of the AdaRank algorithm on training data:*

$$\frac{1}{m}\sum_{i=1}^{m} E(\pi(q_i,\mathbf{d}_i,f_T),\mathbf{y}_i) \geq 1-\prod_{t=1}^{T} e^{-\delta_{min}^t}\sqrt{1-\varphi(t)^2},$$

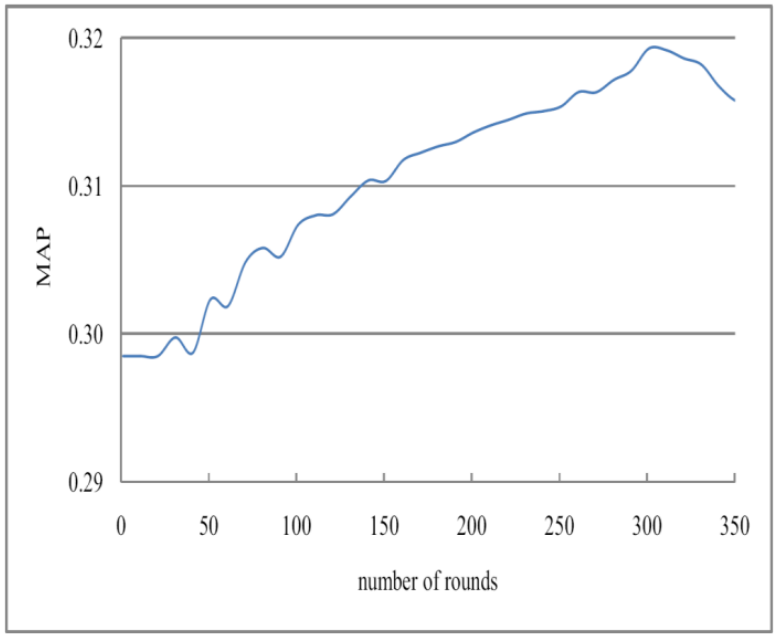
where $\varphi(t)=\sum_{i=1}^{m} P_t(i)E(\pi(q_i,\mathbf{d}_i,h_t),\mathbf{y}_i)$, $\delta_{min}^t=\min_{i=1,\cdots,m}\delta_i^t$, *and*

$$\delta_i^t = E(\pi(q_i,\mathbf{d}_i,f_{t-1}+\alpha_t h_t),\mathbf{y}_i) - E(\pi(q_i,\mathbf{d}_i,f_{t-1}),\mathbf{y}_i) - \alpha_t E(\pi(q_i,\mathbf{d}_i,h_t),\mathbf{y}_i),$$

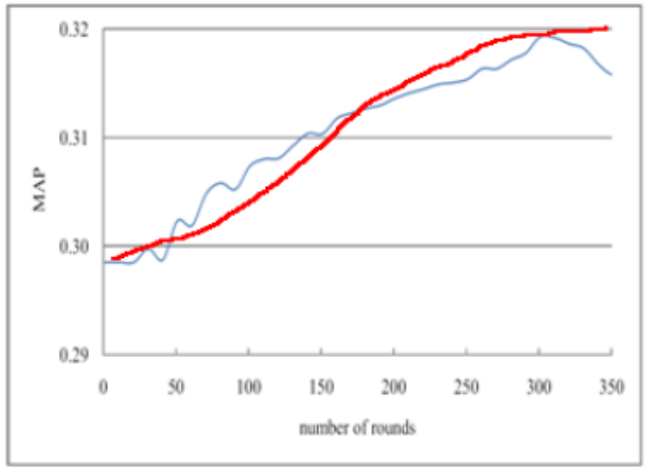
for all $i=1,2,\cdots,m$ *and* $t=1,2,\cdots,T$.

实际操作AdaRank

- 不一定满足定理条件，因此AdaRank并不总是能收敛



- 解决办法：当选择新一轮弱函数，首先检测一下是否满足定理条件，如果不满足则选择次好的函数。



SoftRank(1)

- 思想：为了避免排序得到位置，将概率的思想引入到排序的过程中。

分数（随机变量）的分布

$$p(s_j) = N\left(s_j | f(x_j), \sigma_s^2\right).$$

两个文档间相对关系的概率

$$P_{u,v} = \int_0^{\infty} N\left(s | f(x_u) - f(x_v), 2\sigma_s^2\right) ds.$$

通过递归过程求解位置概率

$$P_j^{(u)}(r) = P_j^{(u-1)}(r-1) P_{u,j} + P_j^{(u-1)}(r)(1 - P_{u,j}).$$

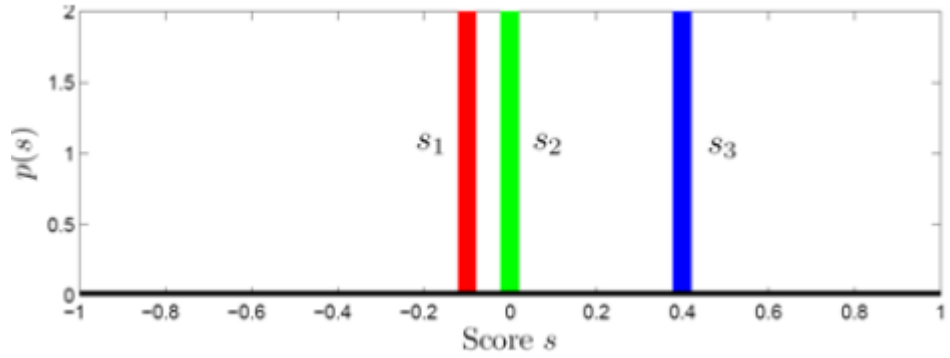
损失函数（概率化NDCG）

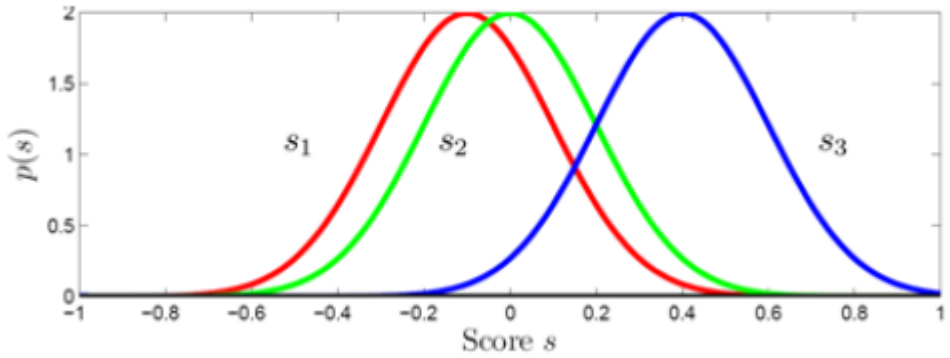
$$L(f; \mathbf{x}, \mathbf{y}) = 1 - \frac{1}{Z_m} \sum_{j=1}^{m} \left(2^{y_j} - 1\right) \sum_{r=1}^{m} \eta(r) P_j(r).$$

SoftRank(2)

- 使用高斯得到打分函数的概率分布

$$p(s_j) = N\left(s_j \mid f(x_j), \sigma_s^2\right).$$





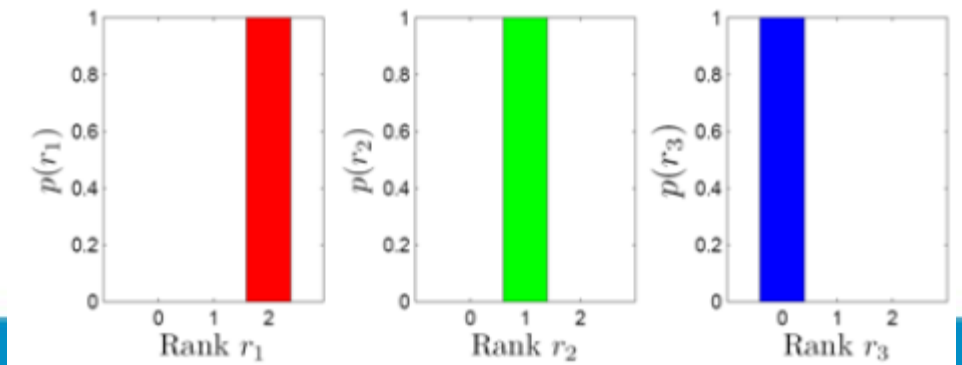
SoftRank(3)

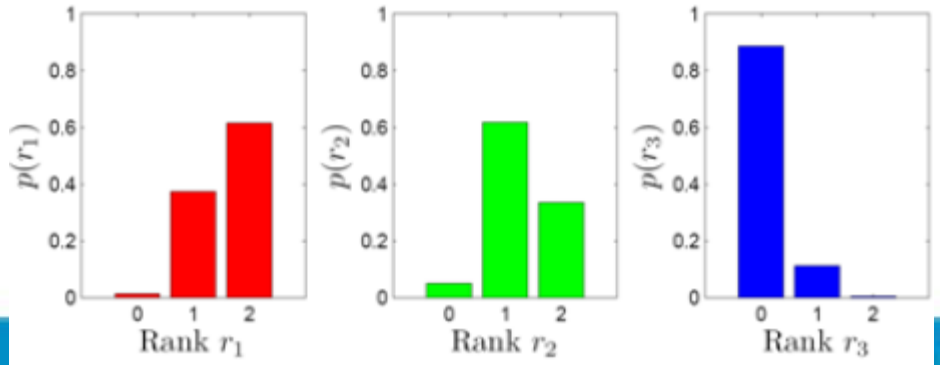
- 由分数的概率分布得到位置的概率分布

$$p(s_j) = N\left(s_j | f(x_j), \sigma_s^2\right).$$

$$P_{u,v} = \int_0^{\infty} N\left(s | f(x_u) - f(x_v), 2\sigma_s^2\right) ds.$$

$$P_j^{(u)}(r) = P_j^{(u-1)}(r-1) P_{u,j} + P_j^{(u-1)}(r)(1 - P_{u,j}).$$





SoftRank(4)

- 最后优化目标期望的NDCG

$$L(f;\mathbf{x},\mathbf{y}) = 1 - \frac{1}{Z_m}\sum_{j=1}^{m}\left(2^{y_j}-1\right)\sum_{r=1}^{m}\eta(r)P_j(r).$$

- 通过梯度下降的方式求解

SVMmap(1)

- 使用结构SVM(structure SVM) 直接优化AP

$$\min \frac{1}{2}\|w\|^2 + \frac{\lambda}{n}\sum_{i=1}^{n}\xi^{(i)}$$

$$\text{s.t.} \quad \forall \mathbf{y}^{c(i)} \neq \mathbf{y}^{(i)},$$

$$w^T\Psi\left(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}\right) \geq w^T\Psi\left(\mathbf{y}^{c(i)}, \mathbf{x}^{(i)}\right) + 1 - \mathrm{AP}\left(\mathbf{y}^{c(i)}, \mathbf{y}^{(i)}\right) - \xi^{(i)}.$$

联合特征映射

$$\Psi(\mathbf{y}, \mathbf{x}) = \sum_{u,v:y_u=1,y_v=0} (x_u - x_v),$$

$$\Psi\left(\mathbf{y}^c, \mathbf{x}\right) = \sum_{u,v:y_u=1,y_v=0} \left(y_u^c - y_v^c\right)(x_u - x_v).$$

SVMmap(2)

- 难点：约束集合中置换指数量级！
- 解决办法
  - 第一步：从无约束开始，只在当前可行域中进行优化计算
  - 第二步：使用第一步中得到的函数找最大违背约束
  - 第三步：如果当前得到约束要比最大违背约束违背更多，则将其加入到可行域中
- 通过该办法，可将迭代次数降低到多项式时间内。

Algorithm 1 Cutting plane algorithm for solving OP 1 within tolerance ϵ.

Input: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$, C, ϵ
$\mathcal{W}_i \leftarrow \emptyset$ for all $i = 1, \ldots, n$
repeat
  for $i = 1, \ldots, n$ **do**
    $H(\mathbf{y}; \mathbf{w}) \equiv \Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y}) - \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y}_i)$
    compute $\hat{\mathbf{y}} = \operatorname{argmax}_{\mathbf{y} \in \mathcal{Y}} H(\mathbf{y}; \mathbf{w})$
    compute $\xi_i = \max\{0, \max_{\mathbf{y} \in \mathcal{W}_i} H(\mathbf{y}; \mathbf{w})\}$
    if $H(\hat{\mathbf{y}}; \mathbf{w}) > \xi_i + \epsilon$ **then**
      $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{\hat{\mathbf{y}}\}$
      $\mathbf{w} \leftarrow$ optimize (3) over $\mathcal{W} = \bigcup_i \mathcal{W}_i$
    end if
  end for
until no $\mathcal{W}_i$ has changed during iteration

SVMmap(3)

- 找最大违背约束的算法：

$$\text{Violation} \triangleq 1 - \text{AP}(\mathbf{y}^c, \mathbf{y}) + w^T \Psi(\mathbf{y}^c, \mathbf{x}).$$

  - 按照相关度顺序排的方式找到一个最好的排序
  - 从最好的排序开始，交换任意两个相邻的相关和不相关文档来得到最大违背约束
  - 算法复杂度是O(nlogn)

Listwise方法小结(1)

- 优点：
  - 充分考虑学习与评价的一致性
    - 以查询下文档集合为对象，符合查询级别的评价
    - 文档的位置信息在算法中有体现
  - 很多数据上实验效果更好
- 缺点：
  - 模型较复杂
  - 很多算法训练过程也更复杂

Listwise方法小结(2)

- 各种损失函数的比较

ListNet $L(f;\mathbf{x},\Omega_y)=D\left(P_y(\pi)\,\|\,P\left(\pi|\left(f(w,\mathbf{x})\right)\right)\right)$

ListMLE $L(f;\mathbf{x},\pi_y)=-\log P\left(\pi_y|f(w,\mathbf{x})\right).$

与置换级别0-1损失有关

SoftRank $L(f;\mathbf{x},\mathbf{y})=1-\frac{1}{Z_m}\sum_{j=1}^{m}\left(2^{y_j}-1\right)\sum_{r=1}^{m}\eta(r)P_j(r).$

SVMmap $\max_{\mathbf{y}^c\neq\mathbf{y}}\left[1-M\left(\mathbf{y}^c,\mathbf{y}\right)+w^T\Psi\left(\mathbf{y}^c,\mathbf{x}\right)-w^T\Psi\left(\mathbf{y},\mathbf{x}\right)\right]_+$

AdaRank $1-M(f,\mathbf{x},\mathbf{y}).$

与评价准则有关

三类排序学习方法的比较(1)

Table 1.2 Summary of approaches to learning to rank

Category	Pointwise		
	Regression	Classification	Ordinal regression
Input space	Single document x_j		
Output space	Real value y_j	Non-ordered category y_j	Ordered category y_j
Hypothesis space	$f(x_j)$	Classifier on $f(x_j)$	$f(x_j)$ + thresholding
Loss function	$L(f; x_j, y_j)$		

Category	Pairwise	Listwise	
	–	Non-measure-specific	Measure-specific
Input space	Document pair (x_u, x_v)	Set of documents $\mathbf{x} = \{x_j\}_{j=1}^{m}$	
Output space	Preference $y_{u,v}$	Ranked list π_y	
Hypothesis space	$2 \cdot I_{\{f(x_u) > f(x_v)\}} - 1$	$sort \circ f(\mathbf{x})$	
Loss function	$L(f; x_u, x_v, y_{u,v})$	$L(f; \mathbf{x}, \pi_y)$	

三类排序学习方法的比较(2)

	Pointwise	Pairwise	Listwise
简便性	***	**	*
建模合理性	*	**	***
与评价准则的符合程度	*	**	***
实验效果	*	***	***
计算复杂度	***	*	*
实际应用	*	***	*
综合评价	*	***	**

参考文献(1)

- *R. Herbrich, T. Graepel, et al. Support Vector Learning for Ordinal Regression, ICANN1999.*
- *T. Joachims, Optimizing Search Engines Using Clickthrough Data, KDD 2002.*
- *Y. Freund, R. Iyer, et al. An Efficient Boosting Algorithm for Combining Preferences, JMLR 2003.*
- *R. Nallapati, Discriminative model for information retrieval, SIGIR 2004.*
- *J. Gao, H. Qi, et al. Linear discriminant model for information retrieval, SIGIR 2005.*
- *D. Metzler, W. Croft. A Markov random field model for term dependencies, SIGIR 2005.*
- *C.J.C. Burges, T. Shaked, et al. Learning to Rank using Gradient Descent, ICML 2005.*
- *I. Tsochantaridis, T. Joachims, et al. Large Margin Methods for Structured and Interdependent Output Variables, JMLR, 2005.*
- *F. Radlinski, T. Joachims, Query Chains: Learning to Rank from Implicit Feedback, KDD 2005.*
- *I. Matveeva, C.J.C. Burges, et al. High accuracy retrieval with multiple nested ranker, SIGIR 2006.*
- *C.J.C. Burges, R. Ragno, et al. Learning to Rank with Nonsmooth Cost Functions , NIPS 2006*
- *Y. Cao, J. Xu, et al. Adapting Ranking SVM to Information Retrieval, SIGIR 2006.*
- *Z. Cao, T. Qin, et al. Learning to Rank: From Pairwise to Listwise Approach, ICML 2007.*
- *T. Qin, T.-Y. Liu, et al, Multiple hyperplane Ranker, SIGIR 2007.*
- *T.-Y. Liu, J. Xu, et al. LETOR: Benchmark dataset for research on learning to rank for information retrieval, LR4IR 2007.*
- *M.-F. Tsai, T.-Y. Liu, et al. FRank: A Ranking Method with Fidelity Loss, SIGIR 2007.*
- *Z. Zheng, H. Zha, et al. A General Boosting Method and its Application to Learning Ranking Functions for Web Search, NIPS 2007.*

参考文献(2)

- *Z. Zheng, H. Zha, et al, A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgment, SIGIR 2007.*
- *M. Taylor, J. Guiver, et al. SoftRank: Optimising Non-Smooth Rank Metrics, LR4IR 2007.*
- *J.-Y. Yeh, J.-Y. Lin, et al. Learning to Rank for Information Retrieval using Generic Programming, LR4IR 2007.*
- *T. Qin, T.-Y. Liu, et al, Query-level Loss Function for Information Retrieval, Information Processing and Management, 2007.*
- *X. Geng, T.-Y. Liu, et al, Feature Selection for Ranking, SIGIR 2007.*
- *Y. Yue, T. Finley, et al. A Support Vector Method for Optimizing Average Precision, SIGIR 2007.*
- *Y.-T. Liu, T.-Y. Liu, et al, Supervised Rank Aggregation, WWW 2007.*
- *J. Xu and H. Li, A Boosting Algorithm for Information Retrieval, SIGIR 2007.*
- *F. Radlinski, T. Joachims. Active Exploration for Learning Rankings from Clickthrough Data, KDD 2007.*
- *P. Li, C. Burges, et al. McRank: Learning to Rank Using Classification and Gradient Boosting, NIPS 2007.*
- *C. Cortes, M. Mohri, et al. Magnitude-preserving Ranking Algorithms, ICML 2007.*
- *H. Almeida, M. Goncalves, et al. A combined component approach for finding collection-adapted ranking functions based on genetic programming, SIGIR 2007.*
- *T. Qin, T.-Y. Liu, et al, Learning to Rank Relational Objects and Its Application to Web Search, WWW 2008.*
- *R. Jin, H. Valizadegan, et al. Ranking Refinement and Its Application to Information Retrieval, WWW 2008.*
- *F. Xia. T.-Y. Liu, et al. Listwise Approach to Learning to Rank – Theory and Algorithm, ICML 2008.*
- *Y. Lan, T.-Y. Liu, et al. On Generalization Ability of Learning to Rank Algorithms, ICML 2008.*

推荐几本教材

- 排序学习：
  - 刘铁岩“Learning to Rank for Information Retrieval”
  - 李航“Learning to Rank for Information Retrieval and Natural Language Processing”
- 机器学习
  - Christopher M. Bishop, “Pattern Recognition and Machine Learning”
  - Hastie, Tribshirani, Friedman, “The Element of Statistical Learning”
  - 李航，“统计学习方法”

Q&A lanyanyan@ict.ac.cn