# LearningToRank 算法介绍

July 1, 2020

## 1 RankNet [1] [2]

### 1.1 符号定义

输入的特征向量对：$x_i, x_j$；

对应的标注：$U_i, U_j$，$U_i \triangleright U_j$ 代表 $U_i$ 应该排在 $U_j$ 前面。

Rank 的打分函数记作 $s = f(x; w)$，模型的参数为 $w$。比较样本 $i, j$ 的打分函数记作：$S_{ij}$，并且 $S_{ij}$ 的取值空间为：$\{+1, -1, 0\}$，其中，$+1$ 代表 $i$ 的排序比 $j$ 靠前；

### 1.2 代价函数和梯度

RankNet 和 LambdaRank 同属于 pairwise 方法。对于某一个 query，pairwise 方法并不关心某个 doc 与这个 query 的相关程度的具体数值，而是将对所有 docs 的排序问题转化为求解任意两个 docs 的先后问题，即：根据 docs 与 query 的相关程度，比较任意两个不同文档 $i$ 和 $j$ 的相对位置关系，并将 query 更相关的 doc 排在前面。

RankNet 巧妙的借用了 sigmoid 函数来定义样本 $i$ 比样本 $j$ ($U_i \triangleright U_j$) 更相关的概率为：

$$P_{ij} = P(U_i \triangleright U_j) = \frac{1}{1 + e^{-\sigma(s_i - s_j)}}$$

$\sigma$ 为待学习的参数，$\sigma(x) = wx + b$。

若 $i$ 比 $j$ 更相关，则 $P_{ij} > 0.5$ ，反之 $P_{ij} < 0.5$。

所以，记 $\bar{P}_{ij}$ 为真实的概率（取值范围为 $[0, 1]$），则有：

$$\bar{P}_{ij} = \frac{1}{2}(1 + S_{ij})$$

RankNet 使用交叉熵函数作为损失函数，单个样本对的交叉熵损失函数（loss）为：

$$C_{ij} = -\sum_{i=1}^{N} \bar{y_{ij}} \log y_{ij} = -[\bar{P}_{ij} \log P_{ij} + (1 - \bar{P}_{ij}) \log (1 - P_{ij})]$$

代入公示后，可以求得对于单个样本对的交叉熵损失函数具体表达式为

$$
\begin{aligned}
C_{ij} &= -\bar{P}_{ij} \log P_{ij} (1 - \bar{P}_{ij}) \log 1 - P_{ij} && (1)\\
&= -\frac{1}{2}(1 + S_{ij}) \cdot \log \frac{1}{1 + e^{-\sigma(s_i - s_j)}} - [1 - \frac{1}{2}(1 + S_{ij})] \cdot \log [1 - \frac{1}{1 + e^{-\sigma(s_i - s_j)}}] && (2)\\
&= -\frac{1}{2}(1 + S_{ij}) \cdot \log \frac{1}{1 + e^{-\sigma(s_i - s_j)}} - \frac{1}{2}(1 - S_{ij}) \cdot [-\sigma(s_i - s_j) + \log \frac{1}{1 + e^{-\sigma(s_i - s_j)}}] && (3)\\
&= \frac{1}{2}(1 - S_{ij}) \cdot \sigma(s_i - s_j) + \log[1 + e^{-\sigma(s_i - s_j)} && (4)
\end{aligned}
$$

所以 $C_{ij}$ 关于任一待优化参数 $w_k$ 的偏导数为

$$\frac{\partial C_{ij}}{\partial w_k} = \frac{\partial C_{ij}}{\partial s_i} \frac{\partial s_i}{\partial w_k} + \frac{\partial C_{ij}}{\partial s_j} \frac{\partial s_j}{\partial w_k}$$

使用随机梯度下降法（SGD）对参数进行优化：

$$w_k \to w_k - \eta \frac{C_{ij}}{\partial w_k} = w_k - \eta (\frac{\partial C_{ij}}{\partial s_i} \frac{\partial s_i}{\partial w_k} + \frac{\partial C_{ij}}{\partial s_j} \frac{\partial s_j}{\partial w_k})$$

## 1.3 应用

根据上面的推导，给定两个样本 $i$ 和 $j$ ，可通过 $s = f(x)$ 来比较它们排序的得分：

$$P_{ij} = P(U_i \rhd U_j) = \frac{1}{1 + e^{-\sigma(s_i - s_j)}}$$

但是实际应用时，分别计算各自如下得分即可：

$$P_i = \frac{1}{1 + e^{-\sigma(s_i)}}$$

$$P_j = \frac{1}{1 + e^{-\sigma(s_j)}}$$

原因如下：

如果 $U_i \rhd U_j$，那么有：

$$P_{ij} = P(U_i \rhd U_j) \to 1 \tag{5}$$

$$\Rightarrow \frac{1}{1+e^{-\sigma(s_i - s_j)}} \to 1 \tag{6}$$

$$\Rightarrow e^{-\sigma(s_i - s_j)} \to 0 \tag{7}$$

$$\Rightarrow w(x_i - w_j) + b \to \infty \tag{8}$$

$$\Rightarrow wx_i >> wx_j \tag{9}$$

$$\Rightarrow \frac{1}{1+e^{-\sigma s_i}} > \frac{1}{1+e^{-\sigma s_j}} \tag{10}$$

$$\Rightarrow P_i > P_j \tag{11}$$

因此，预测时不需要两两比较计算 $P_{ij}$，直接对各个样本 $i$ 计算 $P_i$，然后排序即可。

# References


[1] C. J. Burges, "From ranknet to lambdarank to lambdamart: An overview," Tech. Rep. MSR-TR-2010-82, June 2010. [Online]. Available: https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/

[2] "浅谈 learning to rank 中的 ranknet 和 lambdarank 算法." [Online]. Available: https://zhuanlan.zhihu.com/p/68682607