RankNet 方法就是使用交叉熵作为损失函数，学习出一些模型（例如神经网络、决策树等）来计算每个 pair 的排序得分，学习模型的过程可以使用梯度下降法。

RankNet 使用神经网络对文档进行打分，$f(x_1)$表示样本x_1的分数，如果$f(x_1) > f(x_2)$，则表示x_1排名高于x_2（用$x_1 \rightarrow x_2$表示）。

我们可以利用函数 f 计算得到样本x_i比样本x_j排名更高的概率，如下所示。

$$o_i = f(x_i)$$

$$o_{ij} = f(x_i) - f(x_j)$$

$$P_{ij} = \frac{e^{o_{ij}}}{1 + e^{o_{ij}}} = \frac{1}{1 + e^{-o_{ij}}} \quad \#\ P_{ij} 表示 x_i 排名高于 x_j 的概率$$

另外还需要 x_i 比 x_j 排名更高 (即 x_i 比 x_j 更加相关) 的真实概率，在数据集中有参数 S_{ij}。

如果 x_i 相关性比 x_j 更高，则 S_{ij}= 1；

如果 x_i 相关性比 x_j 低，则 S_{ij} = -1；

如果 x_i 相关性和 x_j 一样，则 S_{ij} = 0。

我们可以通过下面的公式计算 x_i 比 x_j 排名更高的真实概率。

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

RankNet 的损失函数采用了 cross entrophy，根据 损失函数进行梯度下降优化神经网络的参数 (即函数 f)，损失函数如下所示。

$$C_{ij} = -\overline{P}_{ij} log P_{ij} - (1 - \overline{P}_{ij}) \log(1 - P_{ij})$$

下图是不同真实概率下，损失函数取值和 $(o_i\text{-}o_j)$ 的关系。