Diversity-L2R

Thiago Vieira de Alcantara Silva

Last Modified at August 2nd, 2017

1 Introduction

Given a set of documents D and a set of queries Q the goal of Learning to Rank (L2R) is to learn a model that ranks D and any other documents, given any other query. In the "classic" version, we're just concerned with precision. But here we added another variable, diversity.
So, we try to optimize the F score between precision and diversity. Diversity defined as the number of different types found among the top k ranked documents.
The precision of each document can be forecasted using any L2R model; one can use methods like Random Forest, SVM, LambdaMART, etc.
And the types of each document can be defined using any method you want.

2 Formulation

The formulation minimizes the inverse of the F score.
The x_i variables represents whether a document is picked or not to be among the top k documents.
The y_j variables represents whether a document of type j is among the top k documents.

$$min \quad \frac{1}{\sum_i p_i x_i} + \frac{1}{\sum_j y_j}$$

$$\begin{aligned} s.t. \quad & \sum_i x_i = k \\ & y_j \leq \sum_i t_{ij} x_i \quad \forall j \\ & y \in \{0,1\} \qquad x \in \{0,1\} \end{aligned}$$

The formulation is very simple, but as you can see, there is a problem here. The objective function if not linear. But F^{-1} is monotonic in parts. So, one of the things that we can do is to brute force the possible values of $\frac{1}{\sum_j y_j}$ and binary search the rest of the objective function. And that is what we're going to do.

In order to do the binary search trick, three constants must be introduced θ, θ_1 and θ_2.
We state that $\theta = \theta_1 + \theta_2$.
θ_1 must be greater or equal to $\frac{1}{\sum_i p_i x_i}$.
θ_2 must be greater or equal to $\frac{1}{\sum_j y_j}$.
So the formulation follows:

$$
\begin{aligned}
& min \quad \theta \\
s.t. \quad & \sum_i x_i = k \\
& y_j \leq \sum_i t_{ij} x_i \quad \forall j \\
& \theta = \theta_1 + \theta_2 \\
& \theta_1 * \sum_i p_i x_i \geq 1 \\
& \theta_2 * \sum_j y_j \geq 1 \\
& y \in \{0,1\} \qquad x \in \{0,1\}
\end{aligned}
$$

Note that $\sum_j y_j$ can assume any integer value in $[1, k]$. And $\sum_i p_i x_i$ can assume any real value in $(0, k]$. So the algorithm works as follows:
For each possible value of θ_2, do a binary search in θ_1. We know that there is a point on the $(0, k]$ interval where there won't be a valid solution anymore. So the binary search will try to find the maximum value for $\sum_i p_i x_i$ that there is a valid solution.
Also note that the formulation will just serve to solve the decision problem of the binary search, "Given θ_1 and θ_2, is there a valid solution?"

3 Expected Input

Three integers: n, m and k.
n representing the number of documents
m representing the number of types of documents.
k representing the number of documents that will be selected.

n lines follows:
For each line there is a float p, the precision of the i-th document.

Another n lines follows:
For each line there is an integer x, the number of types assigned to the i-th document.
The integer x is followed by x other integers, the types of the document.

Everything here is 0-based.