Machine Learned Ranking (Pointwise)

Authors: Mingjie Ye, Manuel Scurti

1. Introduction

Machine-learned ranking or **Learning to rank** is the the application of machine learning to rank in information retrieval systems.

Generally, the input space is features of documents-query pairs and the output space is an ordered list or ordered list for one particular query.

Compared with pairwise and listwise, pointwise generally converts the ranking problem to classification, regression or ordinal classification. It considers only a single document in the loss function to predict how relevant it is for this query. The final ranking result is achieved by sorting the result scores.

It's simple because we could use many classification or regression algorithms, however, the group structures are ignored.

Overall, the algorithms of pointwise are based on regression and classification, which can be achieved by almost every methods we use in data analysis including machine learning methods, such as Ranking SVM, RankBoost, RankNet, ListMLE, random forest, McRank, logistic regression. While most learning to rank methods learn the ranking function by minimizing the loss functions which is the ranking measures (such as NDCG and MAP) that are used to evaluate the performance of the learned ranking function.

Generally, people use sigmoid cross entropy for binary label in pointwise method.

2. Training set

We build our training set based on loinc_dataset.xlsx. We are using three queries: “glucose in blood”, “bilirubin in plasma” and “White blood cells count”. The documents consist of three features with binary relevance (relevant/irrelevant).

We define the features values as 1 for relevant and 0 for irrelevant manually based on semantic meaning. Therefore each query-docs pair has features $x = [f1, f2, f3], \qquad f1, f2, f3 \in 0, 1$.

The labels of each document-query pair are defined manually as multi-level ratings from 0 to 3 (the larger the integer, the more relevant) added by three feature values. $Label = f1 + f2 + f3 \qquad Label \in 0, 1, 2, 3$. Therefore, we could train this dataset with features and labels.

For each document, we have 20 documents partitioned into five parts with 5 docs denoted as S1, S2, S3, S4, S5, for five-fold cross validation. In each fold, we use three parts for training, one part for validation and one for test.

The training set is used to learn ranking models. The validation set is used to tune the hyper parameters of the learning algorithms. The test set is used to evaluate the performance of the model.

Folds	Training Set	Validation Set	Test Set
Fold1	{S1,S2,S3}	S4	S5
Fold2	{S2,S3,S4}	S5	S1
Fold3	{S3,S4,S5}	S1	S2
Fold4	{S4,S5,S1}	S2	S3
Fold5	{S5,S1,S2}	S3	S4

In order to implement the ranking algorithm, we convert the dataset into LETOR format:

```
   r qid docid C1 C2 C3
1: 0   1     1  0  0  0
2: 1   1     2  1  0  0
3: 2   1     3  1  1  0
4: 0   1     4  0  0  0
5: 0   1     5  0  0  0
6: 1   1     6  1  0  0
```

Each row corresponds to a query-document pair. The first column "r" is relevance label of the pair, the second column is query id, the third column is document id and the following columns are features.

Abbreviations of the features:

```
Bld: Blood
Ser: Serum
Plas: Plasma
Plr fld: Pleural fluid (thoracentesis fld)
Synv fld: Synovial fluid (Joint fluid)
MCnc, CCnc, or SCnc for mass, catalytic, and substance
concentrations(moles) respectively.
Ncnc: cell counts, property number concentration
NFr: property number fraction
PrThr: Presence or Threshold
```

3. Algorithms

Pointwise uses regression or classification methods to train the dataset. The common algorithms are Subset Ranking, McRank, Prank, random forest and SVM.

Before training, we must prepare our dataset which contains two main parts: feature and label.

3.1 Feature

Features of document-query pair could depend only on the document, only on the query or both document and query. Some examples of features are:

TF, TF-IDF, BM25, PageRank, spam HITS ranks and so on.

It is important to select and optimize good features to measure the pairs.

3.2 Label

The label can be defined manually which is relatively close to real situation. But it needs too much time and resource if the dataset is huge.

Besides, the label could achieved by users' behaviors like click-through data especially in search engine.

The types of labels could be binary judgment (relevant/irrelevant), muti-level ratings (Perfect > Excellent > Good > Fair > Bad) in pointwise.

4. Summary of the logistic inference model in Gey's paper

Here, we are going to summary the logistic inference model in Gey's paper. Gey proposed a new probabilistic text and document search method based on logistic regression. This logistic inference method estimates probability of relevance for documents with respect to a query which represents the user’s information need. Documents are then ranked in descending order of their estimated probability of relevance to the query.

The logistic inference model has six elementary clues: query absolute frequency (QAF), relative frequency in the query (QRF), DAF, relative frequency in the document (DRF), inverse document frequency (IDF) and relative frequency of the term in all documents (RFAD). The complete formula for logodds of relevance, given the presence of term t_j is then:

$$Z_{t_j} = \log O(R \mid t_j) = c_0 + c_1 log(QAF) + c_2 log(QRF)$$

$$+ c_3 log(DAF) + c_4 log(DRF) + c_5 log(IDF) + c_6 log(RFAD)$$

After achieving these parameters by regression, they compared this Model with Vector Space Model and used Cranfield, CACM and CISI dataset to evaluate the performances of the model. The steps are:

- For each query take the ranking of relevant documents by each method and compute an average precision value over all ranks.
- Compute the difference in average precision between the two methods and apply T-test to test the significantly different.
- Apply a T-test to these difference under the null hypothesis that the methods perform identically and hence their mean difference should be zero and their standard deviation not significantly different.

Conclusions

Pointwise approaches look at a single document at a time in the loss function. They essentially take a single document and train a classifier / regressor on it to predict how relevant it is for the current query. The final ranking is achieved by simply sorting the result list by these document scores. For pointwise approaches, the score for each document is independent of the other documents that are in the result list for the query. The possible drawback of this approach, compared to other techniques, could be that the position of documents in the ranked list is not modelled by the loss function. Furthermore, sampling is essential, because if the number of unrelevant documents are way more that the relevant ones, we could achieve a model that might not emphasise enough the relevant documents.

References

[1] Gey F.C. (1994) Inferring Probability of Relevance Using the Method of Logistic Regression. In: Croft B.W., van Rijsbergen C.J. (eds) SIGIR '94. Springer, London [2] Tao, Q, Tie-Yan, L. (2010). Microsoft Learning to Rank Datasets, https://www.microsoft.com/en-us/research/project/mslr/ [3] C4 LEARN TO RANK, https://rstudio-pubs-static.s3.amazonaws.com/93221_bba686cb152f4e37abfb6b37ddc8d400.html [4] Wikipedia: Learning to rank, https://en.wikipedia.org/wiki/Learning_to_rank