Introduction

Learning To Rank (LTR) is a class of technique that applies supervised ML to solve Ranking problems. LTR does ranking based on a list of items. LTR doesn't consider the exact score each item gets, but considers more about the relative ordering among all the items unlike traditional ML which solves prediction problem (classification or regression) on a single instance at a time. The most common application of LTR is search engine ranking, but is useful anywhere one needs to produce a ranked list of items.

Lemur project was started by CIIR at UMass, Amherst and LTI at CMU. Lemur project is a collection of software tools, search engines, toolkit APIs for search and IR and LTR algorithm implementation. Lemur RankLib is a library of learning to rank algorithms. Currently eight popular algorithms have been implemented. RankLib is developed in Java programing language.

MART, RankNet, RankBoost, AdaRank, Coordinate Ascent, LambdaMART, ListNet, Random Forests

Description

Lemur RankLib also implements many retrieval metrics as well as provides many ways to carry out evaluation. With appropriate parameters for Random Forests, RankLib can also do bagging several MART/LambdaMART rankers.

Lemur RankLib does no document retrieval. To use RankLib, one needs to generate a query set, relevance information and feature values. Metrics used in generating an algorithm's model require a complete set of relevance judgements and not limited only to the documents in result ranked lists. The basic procedure is to use some initial set of rankings for a set of queries, complete with feature values.
One needs to come up with the set of features and their values. Feature values can be few or numerous. Some common ones are document TF-IDF, BM25 scores, document length, number of matching query terms, number of query terms in important sections of a document, such as a title or web page anchor text, in or out links, etc..

In general, for any LTR (Learning To Rank) algorithm, input file format is crucial. Many implementations of LTR algorithms follow LETOR format.
LETOR datasets (Learning to Rank for Information Retrieval).
Training data is also used as testing/validation data.
For the training data, the file format is used in the same standard as SVM-Rank and LETOR datasets.

LETOR is a package of benchmark data sets for research on LETOR. LETOR is a benchmark collection for research on learning to rank for Information Retrieval managed by Microsoft Research group. LETOR contains standard features, relevance judgements, data partitioning, evaluation tools and several baselines for the OHSUMED data collection and the .gov data collection.

RankLib usage provides a rich set of options to train, choose ranking algorithm, feature-file, metrics to optimize, save and test etc.. RankLib is developed in Java and typically is run using Java command line. The usage of RankLib internally provides each ranking algorithm specific parameters to choose. RankLib implements algorithms that use all 3 types of techniques, i.e., list-wise algorithms (AdaRank, LambdaMART), point-wise and and pair-wise techniques (MART, RankNet, RankBoost).

Conclusion

While RankLib implements AdaRank as well in Java, there are other popular implementations of LTR ranks such as AdaRank in Python (e.g., https://github.com/rueycheng/AdaRank).
There are some other implementations of LTR algorithms like
- LEROT
- Implementation of LambdaRank
- xapian-letor - part of xapian project developed at GSoC 2014
- MLR for Matlab
- ListMLE, ListNET
- SVM-MAP Implementation
- SVM Rank Implementation
- jforests
- IPython demo
- XGBoost

A part of Apache Solr (donated by Bloomberg) also implements Learning To Rank algorithm.

Majorly, the LTR algorithms are implemented in C/C++, Java and Python languages.

RankLib is one of very few packages that provides the implementation of multiple LTR algorithms. RankLib is a good use case to leverage in ElasticSearch implementation. While there are many LTR implementations, RankLib still remains the best due to maturity and proven correctness.

references:
https://sourceforge.net/p/lemur/wiki/RankLib/
https://lemurproject.org/ranklib.php
https://www.quora.com
https://en.wikipedia.org/wiki/Learning_to_rank