Efficiency/Effectiveness Trade-offs in Learning to Rank

Tutorial @ ECML PKDD 2018
http://learningtorank.isti.cnr.it

Claudio Lucchese
Ca' Foscari University of Venice
Venice, Italy

Franco Maria Nardini
HPC Lab, ISTI-CNR
Pisa, Italy

Publicly available Learning to Rank Datasets

- Istella Learning to Rank datasets, 2016, 2018
- Yahoo! Learning to Rank Challenge v2.0, 2011
- Microsoft Learning to Rank datasets (MSLR), 2010
- Yandex IMAT, 2009
- LETOR 4.0, April 2009
- LETOR 3.0, December 2008
- LETOR 2.0, December 2007
- LETOR 1.0, April 2007

Istella Learning to Rank dataset

- Data "used in the past to learn one of the stages of the Istella production ranking pipeline" [1,2].
- Istella LETOR full
  - 33,018 queries
  - 220 features per query-document pair
  - 10,454,629 labeled instances
  - Relevance judgments ranging from 0 (irrelevant) to 4 (perfectly relevant)
  - It comes splitted in train and test sets according to a 80%-20% scheme.
- Istella LETOR small: 3,408,630 labeled instances by sampling irrelevant pairs to an average of 103 examples per query.
- Istella LETOR extended: 26,791,447 documents for 10,000 queries produced by retrieving up to 5,000 documents per query according to the BM25F ranking score.

[1] http://blog.istella.it/istella-learning-to-rank-dataset/

[2] http://www.istella.it

MSLR and Yahoo!

- MSLR Web30K and Web10K
  - From Microsoft
  - Partitioned in five fold for easy cross-validation
  - Two sets, 10K and 30K queries, 136 features, 5-graded label
  - https://www.microsoft.com/en-us/research/project/mslr/
- Yahoo! Learning to Rank v2.0
  - From Yahoo!
  - Two datasets
  - Each dataset is divided in 3 sets: training, validation, and test.
  - Set 1: n. queries (train, valid, test) 19,944 2,994 6,983. n. features: 519
  - Set 2: n. queries (train, valid, test) 1,266 1,266 3,798. n. features: 596
  - https://webscope.sandbox.yahoo.com/

Software and Libraries

- Training Learning to Rank models
  - XGBoost, University of Washington
  - LightGBM, Microsoft
  - CatBoost, Yandex
  - QuickRank, ISTI-CNR
  - scikit-learn
  - jforests
- Evaluation of Learning to Rank solutions
  - RankEval

XGBoost

dmlc XGBoost

- Optimized distributed gradient boosting library
- Implements machine learning gradient boosting algorithms
  - Including DART
- Support for major distributed environment (Hadoop, SGE, MPI)
- Support for GPU
  - CUDA Accelerated Tree Construction
- XGBoost4J: Java/Scala API to export the core functionality of XGBoost library.
  - Enable its use within Spark, Flink and Dataflow
- Very popular on Kaggle
- https://github.com/dmlc/xgboost

T. Chen and C. Guestrin. *XGBoost: A Scalable Tree Boosting System*. In Proc. ACM SIGKDD, 2016.

LightGBM

- Fast, distributed, high performance gradient boosting framework based on decision tree
  - Part of the Microsoft Distributed Machine Learning Toolkit (DMTK)
- Parallel and GPU learning supported
- https://github.com/Microsoft/LightGBM
- Experiments on MSN (left) and Yahoo! LETOR (right) against XGBoost [1]

Metric	xgboost	xgboost_hist	LightGBM
ndcg@1	0.483956	0.488649	0.524255
ndcg@3	0.467951	0.473184	0.505327
ndcg@5	0.472476	0.477438	0.510007
ndcg@10	0.492429	0.496967	0.527371

Metric	xgboost	xgboost_hist	LightGBM
ndcg@1	0.719748	0.720223	0.732466
ndcg@3	0.717813	0.721519	0.738048
ndcg@5	0.737849	0.739904	0.756548
ndcg@10	0.78089	0.783013	0.796818

[1] https://github.com/Microsoft/LightGBM/wiki/Experiments

CatBoost

- Machine learning framework by Yandex based on gradient boosting over decision trees.
- Features
  - Support for both numerical and categorical features.
  - Data visualization tools included.
  - Support for the training of oblivious trees.
- Implementations: Python, R, Command-line
- Jupyter notebook tutorials available on GitHub
- https://catboost.yandex/

CatBoost

	CatBoost	LightGBM	XGBoost	H2O
	Tuned	Tuned	Tuned	Tuned
Adult	**0.26974**	0.27602 +2.33%	0.27542 +2.11%	0.27510 +1.99%
Amazon	**0.13772**	0.16360 +18.80%	0.16327 +18.56%	0.16264 +18.10%
Click prediction	**0.39090**	0.39633 +1.39%	0.39624 +1.37%	0.39759 +1.72%
KDD appetency	**0.07151**	0.07179 +0.40%	0.07176 +0.35%	0.07246 +1.33%
KDD churn	**0.23129**	0.23205 +0.33%	0.23312 +0.80%	0.23275 +0.64%

QuickRank

- A parallel C++ suite of Learning to Rank algorithms
- The algorithms currently implemented are:
  - **GBRT** [Fried01], **LambdaMART** [Wu10],
  - **Oblivious GBRT/LambdaMART** [Sega10]
  - **CoordinateAscent** [Metz07], **LineSearch** [Luen84], **RankBoost** [Freu03]
  - **DART** [Rash15], **X-DART** [Lucc17],
  - **CLEAVER** [Lucc16], **X-CLEAVER** [Lucc18]
- Available under Reciprocal Public License 1.5
- http://quickrank.isti.cnr.it/

jforests

- jforests is a Java library that implements tree-based learning algorithms.
- Provides "Risk-Sensitive" LambdaMART
- https://github.com/yasserg/jforests
- Available under Apache License 2.0
- Maintained by The University of Glasgow.

Y. Ganjisaffar, R. Caruana, C. Lopes, *Bagging Gradient-Boosted Trees for High Precision, Low Variance Ranking Models*. In Proc. ACM SIGIR, 2011.

RankEval

- An Evaluation and Analysis Framework for Learning-to-Rank Solutions
- Functionalities:
  - Effectiveness Analysis
  - Statistical Analysis
  - Topological Analysis
  - Feature Analysis
- Available for Python 2 and 3
- Support for several formats: LightGBM, XGBoost, QuickRank, scikit-learn, jforests, CatBoost
- Available Jupyter Notebooks to ease its use
- Available under Mozilla Public License 2.0
- http://rankeval.isti.cnr.it/, https://github.com/hpclab/rankeval

References

[Fried01] J. H. Friedman. *Greedy function approximation: a gradient boosting machine*. Annals of Statistics, pages 1189–1232, 2001.

[Wu10] Q. Wu, C. Burges, K. Svore, and J. Gao. *Adapting boosting for information retrieval measures*. Information Retrieval, 2010.

[Sega10] I. Segalovich. *Machine learning in search quality at yandex*. Invited Talk, ACM SIGIR, 2010.

[Metz07] Metzler, D., Croft, W.B. *Linear feature-based models for information retrieval*. Information Retrieval 10(3), pages 257–274, 2007.

[Luen84] D. G. Luenberger. *Linear and nonlinear programming*. Addison Wesley, 1984.

[Freu03] Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. *An efficient boosting algorithm for combining preferences*. The Journal of machine learning research, 4, 933-969 (2003).

[Rash15] K.V. Rashmi and R. Gilad-Bachrach. *Dart: Dropouts meet multiple additive regression trees*. Journal of Machine Learning Research, 38 (2015).

[Lucc16] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, F. Silvestri, S. Trani. *Post-Learning Optimization of Tree Ensembles for Efficient Ranking*. ACM SIGIR, 2016.

[Lucc17] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego and S. Trani. *X-DART: Blending Dropout and Pruning for Efficient Learning to Rank*. ACM SIGIR, 2017.

[Lucc18] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, F. Silvestri, S. Trani. *X-CLEaVER: Learning Ranking Ensembles by Growing and Pruning Trees*. ACM TIST 2018.