Learn to Rank
ICDM 2013 Challenge
Ranking Hotel Search Queries

Saurabh Agarwal, Luke Styles, Saurabh Verma

1 Introduction

Learning to Rank (LeToR) is an important class of machine learning problems that focuses on finding an optimal sequence of documents as a function of some user query. In this project we consider hotel search and click-through data provided by the popular travel website Expedia.com, with the goal of developing a model that will provide a list of hotels ranked by highest likelihood of customer purchase. In this work, we propose to use the framework from the logistic regression binary classification algorithm to describe a straightforward linear model for ranking.

2 Related Work

Important families of LeToR algorithms include point-wise and pairwise approaches [1]. The point-wise approach to ranking applies proven machine learning regression and classification techniques to the ranking problem. This family takes as an input a feature vector for a single document, and outputs a relevance score for the given query. A well-known example is the MART family of boosting algorithms, which in effect use regression trees to perform gradient descent in the function space [2, 3].

The pair-wise approach predicts the relative desirability between pairs of documents, and accordingly the input for a pairwise algorithms is two feature vectors corresponding to documents to be compared. In contrast to the point-wise approach, no absolute relevance score is given. Ranking proceeds by comparing relative relavance between pairs of documents to arrive at a final ordering. Many methods may be developed by applying different loss functions to perform ranking in this formulation [4]. RankNet is an important pair-wise LeToR algorithm which incorporates cross entropy as its objective function[5, 6]. LamdaMart is another pairwise approach which combines Lamda and MART algorithm [7, 8].

3 Proposed Method

In our case, we consider a pointwise approach and convert commonly used classification algorithms for ranking. We propose a modified model of logistic regression and random forest for classification.

3.1 Logistic Regression

Logistic regression is a widely used and well-studied classification model [9]. As a linear classifier, the method trains a weight vector to give the posterior probability of class membership given some feature vector. Here we use a similar formulation, instead seeking the probability of a hotel being purchased (event R) for a given user query. Given features of the jth document, the log-odds of purchase for the document X_j is given by,

$$\log\left(\frac{\Pr(R|X_j)}{1-\Pr(R|X_j)}\right) = c + \sum_{i=1}^{d} w_i x_{ij} \tag{1}$$

where c is a constant offset. Thus, the probabality of relevance of document to a query is equal to,

$$\Pr(R|X_j) = \frac{1}{1+\exp(-c-\sum w_i x_{ij})} \tag{2}$$

and the parameters w_i are predicted by Maximum Liklihood Estimation on the training data. In our formulation, we propose to use the posterior probability directly as the relevance score for a test query.

As relatively few queries result in purchases, it is common in purchase click-through data for the target class to be represented in a small minority of training instances, and this is seen in the Expedia hotel booking data. A popular technique used to mitigate this hazard is to perform training on a sampled subset of the training set, with the sampling biased to provide proportionally more examples of purchases.

We utilized an alternative approach to handle the class imbalance problem which was shown to work well on the provided dataset [10]. We adjust the weight of positive class by assinging a value α in cross entropy function derived from the MLE process. This modified version of the cross entropy error (CCE) then becomes,

$$CEE = -\sum_{n=1}^{N} \log[y_n^{\alpha\mathbb{1}(t_n=1)} \dot{} (1-y_n)^{\mathbb{1}(t_n=0)}] \tag{3}$$

$$= -\sum_{n=1}^{N} [\alpha \log y_n + (1-t_n)\log(1-y_n)] \tag{4}$$

With $\mathbb{I}$ being the indicator function and $t_n = 1$ referring to the occurance of a purchase.

This modification preserves convexity in the objective function, which may be optimized by one of many iterative methods including gradient descent. By adding the bias term α we are able to handle the class imbalance while using the entireity of the majority class in training.

3.2 Random Forest

Random forests comprise a popular family of classifiers known for an ability to handle heterogeneous and noisy datasets [11]. Ensemble methods such as bagged classifiers and forests of decision trees may be extended to the ranking problem by the use of probabilistic estimation trees (PETs). PETs use the train impurity of the decision tree nodes nodes to estimate the posterior probability of class membership for a query document [12]. In a random forest of PETs, the posterior may be estimated as the unweighted average or the maximum estimation of the ensemble's weak classifiers. As with the logistic model, this may be used for ranking as a confidance score of a purchase for a query.

The expected gain used in building a decision tree may be selected from a large variety of functions. This study used information gain as the splitting criterion, which is defined as the difference in information entropy between a decision rule and its consequents:

$$InfGain = E(Parent) - E(Child|Rule) \tag{5}$$

and the information entropy function E for a node containing objects x_i is defined as:

$$E(Node) = -\sum_i P(x_i) \log P(X_i) \tag{6}$$

In this study, random forests were implemented as a baseline simple classifier against which the proposed logistic regression-based approach could be compared.

4 Experiment and Results

4.1 Overview of Dataset

The training dataset was provided as part of the 2013 ICDM competition "Personalize Expedia Hotel Purchases" [13]. It contained nearly 10 million historical hotel search results representing approximately 400 thousand unique search queries on the popular travel booking website Expedia.com. The data represented actual customer transaction data selected over a window of the year 2013.

The data was similar to what would be recieved by a ranking program somewhere in the website back end. As such, it was decidedly heterogeneous, including statistics on the purchase history of the unique user; categorical and continuous features describing the resultant hotel; and output of other Expedia algorithms including estimated probability that the hotel appears in internet searches. The data included a large amount of missing values and many clearly erroneous outliers. Notably, the queries were not sparsely populated, which makes the problem somewhat different in character than the archetypal web search ranking problem. Potential target features included both click and purchase indicator values.

4.2 Data Preprocessing

A hold-out set representing 10% of the training data was retained for evaluation. As a way to minimize correlation between searches in close proximity, those searches whose ID numbers were 1 modulus 10 were reserved.

We consider both categorical as well as numerical features for our problem. Missing values and and those representing large multiples of the standard deviation of a feature were replaced by the feature mean to minimize outlier bias.

In the case of random forest, a subset of the training data was used which was sampled so as to provide a one-to-one ratio between purchase and non-purchase target variables. This was required in order to prevent all rules from classifying examples as the majority class.

4.3 Model Training

For logistic regression we used all features, categorical and numerical, to train and rank. Categorical features were split into asymmetric binary attributes. Weight parameter α is adjusted on each run to find the lowest test error and comes best for $\alpha = 10$. The convex objective function was optimized to maximize the likelihood using gradient descent.

In the case of random forests we first split the data in 172 pieces by the country of the hotel being searched for. Then we balanced each piece using the procedure mentioned in section 4.1. For each peice we create a random forest of 200 PETs with height two. Thus, a unique model was trained for each of the 172 destination countries. The unweighted average of the base classifiers was used as the query ranking.

4.4 Evaluation Metric

Learning to rank methods require evaluation metrics that can account not only for class membership of the test set, but also relative ranking of the result. For our study, we use Normalized Discounted Cumulative Gain (NCDG) at position 38. NCDG is a commonly used benchmark in information retrieval [14], and was the measure used to determine the winner of the official Expedia learning to rank contest.

NDCG is defined in terms of Discounted Cumulative Gain (DCG). For a sequnce of documents $X = \{x_1, \ldots, x_n\}$ ranked in order of decreasing predicted relevance, and a relevance function $f(x)$, DCG at position $m < n$ is defined as

$$DCG_m(X) = \sum_{i=1}^{m} = \frac{2^{f(x_i)} - 1}{\log_2(i+1)} \tag{7}$$

Normalized DCG simply divides by the best possible DCG for X, which is desirable because different test sets may have different best-case DCGs. Hence, NDCG returns a score that falls between zero and one, with one being a perfect score - that is, one where documents display monotonically decreasing relevence scores.

The relevance score used in our evaluation was also that provided in the official contest:

$$f(x) = \begin{cases} 5 & \text{if hotel booked} \\ 1 & \text{if hotel clicked} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The relevance function highly rewards hotel purchases, and marginally prefers clicks over no user action.

4.5 Results

The NDCG at 38 results are as follows. We compare our proposed logistic model and random forest baseline to the winning ensemble method and a simple random ordering of the queries. We emphasize that the winning algorithm is the NDCG@38 score publicly reported on the Kaggle leaderboards, and was not generated by the authors.

Algorithm	NDCG Score
LambdaMART, DNN, etc "Kitchen Sink" Ensemble	0.54
Random Forests Benchmark	0.51
Proposed Logistic Regression	0.49
Random Ordering	0.35

Our results show that the proposed algorithm is not only outperformed by the winning algorithm and the proprietary Expedia ranker, but also the basline random forests benchmark. A silver lining is that the method still performs significantly better than a random ordering.

5 Summary

5.1 Conclusion

We have shown that logistic logression provides a framework that may be intuitively and simply extended to attack learning to rank problems, and that it may produce results significantly higher in quality than a random ordering. Regrettably, we were unable to match the performance of even a simple off-the-shelf classifier such as random decision forests of probability estimation trees. We believe this result shows the importance of preprocessing and feature selection in large and heterogeneous datasets of questionable data quality. Indeed, it is the authors' opinion that the better relative performance of the random forest ranker is due to its built-in robustness to heterogeneity in the feature space.

The large size of the training data was found to be a suprisingly difficult challenge as well. With datasets as large as provided in this competition, knowledge of system and hardware implementation and limitations were essential. Data size was also a limiting factor in the development cycle of the project, as model training took well over twenty four hours on available resouces.

5.2 Future Work

The reported results represent an excellent starting point for future explorations in large LeToR problems. A logical next step would be to test the stated hypothesis that the lag in performance of the logistic ranker is due to insufficient preprocessing. Extensive grooming of the data should be performed to see if the performance of the logistic classifer can be made to approach that of a random forest. Machine learning techniques for feature selection, such as sparse group lasso, may also be explored as potential ways to improve performance.

An alternative route of study that could also prove interesting would involve investigating the behavior of different optimization algorithms on the modified CCE function. Because logistic regression is an additive model, there is a great variety of potential techniques in the literature meant to resolve various resource constraints, a particularly interesting recent example being the alternating direct method of multipliers (ADMM).

References

[1] Liu, Tie-Yan. 2009. *Learning to Rank for Information Retrieval.* Springer, New York, NY.

[2] Friedman, Jerome H. Greedy Function Approximation: A Gradient Boosting Machine. *IMS 1999 Reitz Lecture*, (1999, mod. 2000, 2001). Url: `http://statweb.stanford.edu/~jhf/ftp/trebst.pdf`

[3] Freund, Yoav et. al. An Efficient Boosting Algorithm for Combining Preferences. *Journal of Machine Learning Research*, 4 (2004), 933-969.

[4] Sculley, David., Combined Regression and Ranking. *ACM SIDKGG*, 16th Proceedings of (2010), 979-988.

[5] Burges, Christopher. From RankNet to LambdaRank to LambdaMART: An Overview. Microsoft Research Technical Report, 23 June 2010. URL: `http://research.microsoft.com/apps/pubs/default.aspx?id=132652`

[6] Burges, Christopher, et. al. Learning to Rank using Gradient Descent. *22nd International Conference on Machine Learning*, 2005. URL: `http://research.microsoft.com/en-us/um/people/cburges/papers/ICML_ranking.pdf`

[7] Burges, Christopher, et. al. Learning to Rank Using an Ensemble of Lambda-Gradient Models. *Journal of Machine Learning Research: Workshop and Conference Proceedings*, 14 (2011), 25-35. URL: `http://research.microsoft.com/en-us/um/people/cburges/papers/YahooChallenge.pdf`

[8] Qin, Tao, et. al. LETOR: A Benchmark Collection for Research on Learning to Rank for Information Retrieval. *Information Retrieval* (2010).

[9] Bishop, Christopher. 2006. *Pattern Recognition and Machine Learning.* Springer, New York, NY.

[10] Liu, Xudong, et. al. Combination of Diverse Ranking Models for Personalized Expedia Hotel Searches. URL: `http://arxiv.org/pdf/1311.7679.pdf`

[11] Breiman, Leo. Random Forests. *Machine Learning*, 50 (2001), 5-32.

[12] Beygelzimer, Alina, et. al. Conditional Probability Tree Estimation Analysis and Algorithms. *UAI Conference of Uncertainty in Artificial Intelligence*, 25th Proceedings of (2009), 51-58.

[13] Website: ”Personalize Expedia Hotel Searches - ICDM 2013”, *Kaggle, Inc.* URL: `http://www.kaggle.com/c/expedia-personalized-sort`

[14] Qin, Tao, et. al. Query-level loss functions for information retrieval. *Information Processing and Management*, 44 (2008), 838-855.