**IE 506: Machine Learning - Principles and Techniques** **Jan-May 2023**

# McRank: Learning to Rank Using Multiple Classification and Gradient Boosting

*Team Name: Alpha.ai* *Team Members: 22m1521,213192001*

**Abstract**

In this we presents a novel approach to learning to rank using a combination of multiple classification models and gradient boosting. The proposed method, called McRank, uses a set of binary classifiers to rank items and then combines their predictions using gradient boosting. The approach is evaluated on several benchmark datasets and compared to state-of-the-art learning to rank methods. The results show that McRank achieves competitive performance while being more computationally efficient than many existing approaches. Additionally, the paper provides insights into the design choices for McRank, including the number of classifiers and the choice of gradient boosting algorithm. Overall, this work demonstrates the effectiveness of combining multiple classification models and gradient boosting for learning to rank, and provides a promising direction for future research in this area.

## 1 Introduction

Ranking is a crucial problem in information retrieval, e-commerce, and online advertising systems. Traditional ranking algorithms rely on handcrafted features, which can be time-consuming and limit the performance of the system. In recent years, machine learning techniques have shown great promise in solving ranking problems. In this paper, author presents McRank, a learning-to-rank algorithm that uses multiple classification and gradient boosting to efficiently learn the ranking function. McRank learns a set of base classifiers, each of which is trained to distinguish between a pair of items, and then combines them using gradient boosting. We evaluate McRank on several benchmark datasets and show that it outperforms state-of-the-art learning-to-rank algorithms. Additionally, the author demonstrate the effectiveness of McRank on a large-scale commercial search engine, where it significantly improves the quality of the search results. The results suggest that McRank is a promising approach for learning-to-rank and can be applied to a wide range of applications. The URLs are arranged on the pages according to the search engine's proprietary ranking algorithm, which dynamically ranks the URLs in real-time based on the user's input query.The first page of URLs is particularly important because it is the page that most users will see and interact with, and it is crucial that the URLs on this page are relevant and useful to the user. This makes the ranking challenge a dynamic one because the URLs are ranked in real-time and must be adjusted based on the particular query that the user has entered. The three main kinds of learning to rank algorithms are pairwise approach, pointwise method, and listwise approach. RankNet[4], LambdaRank[5], and RankSVM models all employ pairwise methods. While LambdaRank typically takes into account a gradient of NDCG with ranknet loss, RankNet uses a cross entropy loss function.

## 2 Literature Survey

Several approaches have been proposed to address the learning to rank problem. One common approach is to use a single classification model to predict the relevance of documents. For example, Support Vector Machines (SVMs) have been widely used for this purpose. However, this approach does not consider the

inherent complexity of the ranking problem, as it assumes that the relevance of each document is independent of the others. As a result, it may not capture the complex interactions between the features.

To overcome these limitations, some researchers have proposed using multiple classification models to capture the complex interactions between features. For example, the RankNet algorithm proposed by Burges et al. (2005)[4] uses a neural network to predict the probability that one document is ranked higher than another. The LambdaRank algorithm proposed by Burges et al. (2007)[5] extends this approach by incorporating the idea of gradient boosting, which allows for the optimization of non-differentiable measures such as NDCG.

However, these approaches still have some limitations. For example, the RankNet algorithm assumes that the relevance of documents can be represented by a continuous function, which may not be true in practice. Additionally, the LambdaRank algorithm can suffer from overfitting due to its reliance on a large number of weak classifiers.

To address these limitations, Li et al. (2008) proposed the McRank algorithm, which uses multiple classification models to learn a ranking function. The McRank algorithm uses a modified version of gradient boosting, which allows for the optimization of non-differentiable measures such as NDCG. Additionally, the McRank algorithm uses a novel loss function that captures the pairwise relationship between documents. This loss function ensures that the McRank algorithm learns a ranking function that is consistent with the pairwise preferences of users.

Some of the previous work are given below-

**1** RankNet: A neural network model for learning to rank proposed by Burges et al. (2005)[4]. RankNet uses a neural network to learn a ranking function that predicts the probability of one document being ranked higher than another. The network is trained using a pairwise loss function that encourages the network to correctly rank pairs of documents. RankNet was one of the first neural network-based approaches for learning to rank, and its success paved the way for subsequent works like McRank.

**2** LambdaRank: A gradient boosting-based approach for learning to rank proposed by Burges et al. (2007)[5]. LambdaRank extends the pairwise ranking approach of RankNet by incorporating the idea of gradient boosting. The algorithm optimizes a loss function that is a weighted sum of pairwise loss functions, where the weights are determined by the gradient of the loss function. LambdaRank has been shown to outperform RankNet on several benchmark datasets.

**3** ListNet: A listwise approach for learning to rank proposed by Cao et al. (2007). ListNet is a neural network-based approach that learns a ranking function by directly optimizing the expected discounted cumulative gain (DCG) of the ranked list. The algorithm uses a cross-entropy loss function to encourage the network to produce lists that are consistent with the ideal ranking. ListNet has been shown to outperform RankNet and LambdaRank on several benchmark datasets.

These are some previous works that use either neural networks or gradient boosting. McRank combines these two techniques to produce a ranking algorithm that outperforms previous state-of-the-art approaches like RankNet, LambdaRank, and ListNet.

# 3 Methods and Approaches

This project solves the learning to rank problem using classification methods where as previous approaches used regression methods to need solve the ranking problem. The author proposed that regression based

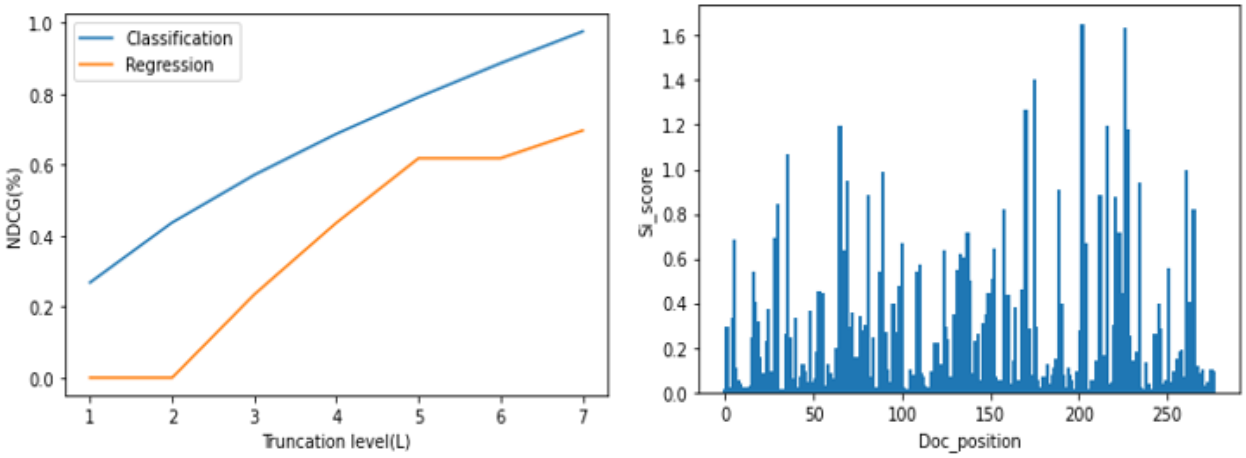
methods used with evaluation metrics i.e. DCG score getting bounded by regression errors is not sufficient. Its better to use learning to rank as a classification problem and bound DCG score by classification errors which makes the algorithm to better than previous existing approaches. The author employed a weighted average product of all the class probabilities with the relevance level to turn the classification issue into a ranking problem. Expected relevance is the term they gave to it.

$$S_i = \sum_{k=0}^{K-1} \hat{p_{i,k}} T(k) \tag{1}$$

where  is the class probability and T(k) is the relevance from 0 to 1.

## 3.1 Work done before mid-term project review

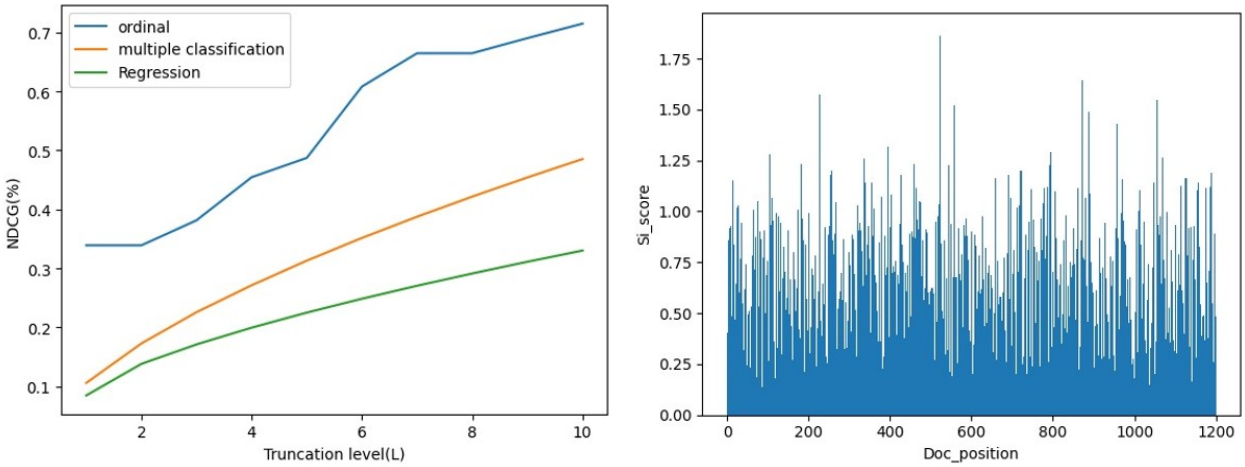
In the mid term we use two algorithm that is classifications and regression to know which one perform better in terms of ranking the data.



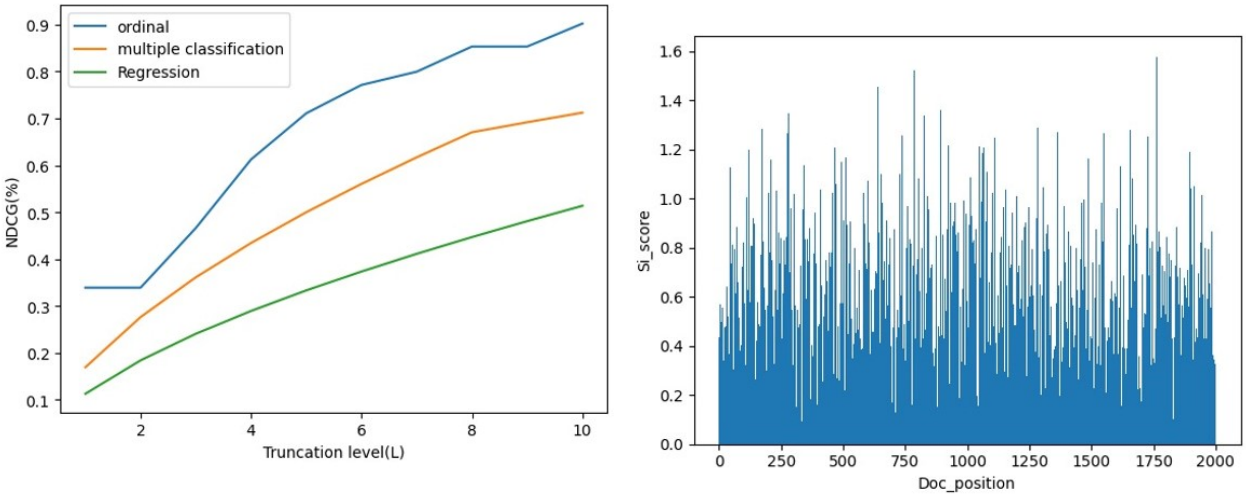

In the above plot we can clearly clearly see that classification perform better than regression, which is also mention by the author in the given paper.

## 3.2 Work done after mid-term project review

The above two figure we shown is shown for data which had 6000 column x 138 rows.We have use three algorithms for calculating the NDCG score at truncation level 10 and we can clearly see that multiple ordinal classification is perform better than multiple classification and Regression.And we have also Ranked the Documents based on there Si score which is shown in the right figure.




The below two figure we shown is shown for data which had 10000 column x 138 rows.We have use three algorithms for calculating the NDCG score at truncation level 10 and we can clearly see that multiple ordinal classification is perform better than multiple classification and Regression.And we have also Ranked the Documents based on there Si score which is shown in the right figure.




# 4 Data set Details

Each entry in the dataset corresponds to a pair consisting of a query and its corresponding URL. The first column in each row indicates the relevance label for the pair, while the second column contains the query ID. The remaining columns in the row represent the features associated with the query-url pair, with each pair being represented by a 136-dimensional feature vector.

The relevance label serves as an indicator of the degree to which the query-url pair is relevant, with higher values indicating a greater degree of relevance.

Below are first rows from MSLR-WEB10K dataset:

0 qid : 10002 1:0.007477 2:0.000000 3:1.000000 4:0.000000 5:0.007470 6:0.000000 7:0.000000 8:0.000000 9:0.000000 10:0.000000 11:0.471076 12:0.000000 13:1.000000 14:0.000000 15:0.477541 16:0.005120 17:0.000000 18:0.571429 19:0.000000 20:0.004806 21:0.768561 22:0.727734 23:0.716277 24:0.582061 25:0.000000 26:0.000000 27:0.000000 28:0.000000 29:0.780495 30:0.962382 31:0.999274 32:0.961524 33:0.000000 34:0.000000 35:0.000000 36:0.000000 37:0.797056 38:0.697327 39:0.721953 40:0.582568 41:0.000000 42:0.000000 43:0.000000 44:0.000000 45:0.000000 46:0.007042

The dataset is organized into columns, with the first column containing labels, the second column containing pairs of query and their corresponding IDs, and the third column onwards containing pairs of query features and document features.

# 5 Experiments

The dataset was acquired from the Microsoft website for the experiment's purposes. The datasets utilised were a subset of the MSLR 30K dataset called MSLR 10K and MSLR 6K. The section above contains information about the dataset. After the dataset was collected, several preprocessing tasks were carried out, such as removing colons from the feature vectors, converting categorical numerical mix features, such as query id, into numerical features, and also changing the datatypes from object into integer for further processing. The dataset was then divided into train and test datasets and was trained using the train dataset.On both datasets, we applied three different models. The models that were employed were the regression-based model, the ordinal classification model, and the multi-classification model. The results were then used to compare classification-based models to earlier models that relied on regression. Gradient-based boosting strategies were used to all models. In the 10k dataset, there are 1000 rows, and in the 6K dataset, there are 6000 rows, and there are 136 characteristics. There are 5 degrees of relevance: 0, 1, 2, 3, and 4. The boosting tree algorithm has three primary inputs. J is the number of terminal nodes in each tree, M is the total number of iterations, and v is the shrinkage factor. We have assigned the value for J=100, M=1000 and v=0.01 in our dataset for these hyperparemeters. Additionally, we determined the average ndcg score result at a truncation level of 10, and the output is provided in the result section.

## 5.1 The Boosting Tree Algorithm for Learning Class Probabilities

For multiple classification, we consider the following common surrogate loss function:

$$\sum_{i=1}^{N} \sum_{k=0}^{K-1} -\log(p_{i,k}) \mathbf{1} y_i = k, \tag{2}$$

where $N$ is the number of data points, $K$ is the number of classes, $y_i \in \{0, 1, \ldots, K-1\}$ is the class label of data point $i$, and $pi, k$ is the predicted probability of data point $i$ belonging to class $k$.

Algorithm 1 implements a boosting tree algorithm for learning class probabilities $p_{i,k}$, and we use basically the same implementation later for regression as well as multiple ordinal classification.

## 5.2 Multiple Ordinal Classification

For multiple ordinal classification, it is common practice to learn the cumulative probabilities $Pr(y_i \leq k)$ rather than the class probabilities $Pr(y_i = k) = p_{i,k}$. In this study, a method similar to the cumulative logits approach used in statistics [cite1] is proposed.

**Algorithm 1** The boosting tree algorithm for multiple classification, taken from [9, Algorithm 6], although the presentation is slightly different.

0: $\tilde{y}_{i,k} = 1$, if $y_i = k$, and $\tilde{y}_{i,k} = 0$ otherwise.
1: $F_{i,k} = 0$, $k = 0$ to $K-1$, $i = 1$ to $N$
2: For $m = 1$ to $M$ Do
3: For $k = 0$ to $K-1$ Do
4: $p_{i,k} = \exp(F_{i,k}) / \sum_{s=0}^{K-1} \exp(F_{i,s})$
5: $\{R_{j,k,m}\}_{j=1}^{J} = J$-terminal node regression tree for $\{\tilde{y}_{i,k} - p_{i,k},\ \mathbf{x}_i\}_{i=1}^{N}$
6: $\beta_{j,k,m} = \frac{K-1}{K} \frac{\sum_{\mathbf{x}_i \in R_{j,k,m}} \tilde{y}_{i,k} - p_{i,k}}{\sum_{\mathbf{x}_i \in R_{j,k,m}} (1-p_{i,k}) p_{i,k}}$
7: $F_{i,k} = F_{i,k} + \nu \sum_{j=1}^{J} \beta_{j,k,m} 1_{\mathbf{x}_i \in R_{j,k,m}}$
8: End
9: End

The proposed method involves partitioning the training data into two groups: $y_i \geq 4$ and $y_i \leq 3$, thereby converting the problem into a binary classification task. The same boosting tree algorithm used for multiple classification can then be applied to learn $Pr(y_i \leq 3)$. This process is repeated for $Pr(y_i \leq 2)$, $Pr(y_i \leq 1)$, and $Pr(y_i \leq 0)$, separately.

Next, the class probabilities $p_{i,k} = Pr(y_i = k) = Pr(y_i \leq k) - Pr(y_i \leq k-1)$ are inferred, and the Expected Relevance is used to compute the ranking scores and sort the URLs.

The approach for both rankers based on multiple classification and multiple ordinal classification is referred to as McRank.

### 5.3 Regression-based Ranking Using Boosting Tree Algorithm

The boosting tree algorithm can be adapted for regression with minor adjustments. The input data comprises pairs of response values and feature vectors, denoted as $(y_i, x_i)_{i=1}^{N}$, where $y_i \in 0, 1, 2, 3, 4$. To perform regression, [cite6] recommended regressing the feature vectors $x_i$ on the response values $2^{y_i - 1}$.

Algorithm 2 presents the least-square boosting tree algorithm, which is similar to Algorithm 3 but replaces the least absolute deviation (LAD) loss with the least square loss. Although the LAD boosting tree algorithm was also implemented, the performance was found to be considerably worse than the least-square tree boost.

**Algorithm 2** The boosting tree algorithm for regressions. After we have learned the values for $S_i$, we use them directly as the ranking scores to order the data points within each query.

0: $\tilde{y}_i = 2^{y_i} - 1$
1: $S_i = \frac{1}{N} \sum_{s=1}^{N} \tilde{y}_s,\ i = 1$ to $N$
2: For $m = 1$ to $M$ Do
5: $\{R_{j,m}\}_{j=1}^{J} = J$-terminal node regression tree for $\{\tilde{y}_i - S_i,\ \mathbf{x}_i\}_{i=1}^{N}$
6: $\beta_{j,m} = \text{mean}_{\mathbf{x}_i \in R_{j,m}} \tilde{y}_i - S_i$
7: $S_i = S_i + \nu \sum_{j=1}^{J} \beta_{j,m} 1_{\mathbf{x}_i \in R_{j,m}}$
9: End

# 6 Results

As shown in Table 1, we obtained NDCG results for several algorithms at a truncation threshold of 10. From the table, we can see that multiple ordinal categorization outperforms other algorithms.

| DATASET | ORDINAL | CLASSIFICATION | REGRESSION |
|---|---|---|---|
| MSLR WEB-10K | 90.2 | 71.2 | 51.3 |
| SLR WEB-6K | 71.47 | 47.3 | 35.34 |

Table 1

# 7 Future Work

The model's interpretability is a concern since McRank is a black-box model. The method of improving the model's interpretability has room for improvement. In addition to being utilised in web search engines and recommendation systems, the McRank algorithm may also be applied in other fields such as e-commerce, banking, and healthcare. Scalability is an issue since McRank is taught on very big datasets. Future research in this area might focus on creating parallelization methods and scalable algorithms to solve the issue.

# 8 Conclusion

The key difficulty is converting classification to ranking score when a learning to rank problem is framed as a classification problem. For this, the boosting approach indicated above is used to learn the class probabilities, and the class probabilities are then transformed into a numerical score using a scoring function. These scores may also be sorted, and they can be used to rank the query. Additionally, the ranking process is enhanced by using ordinal classification to further improve classification problems. To sum up, the use of multiple categorization and gradient boosting for machine learning to rank (MCRank) is a promising strategy that has demonstrated to deliver precise and effective ranking functions. This algorithm's potential applications are many, and there are several avenues for more study and refinement. Overall, MCRank is a strong and adaptable method that has the ability to completely change how we rank and access information. It uses multiple categorization and gradient boosting. Exciting developments in machine learning and other fields will definitely result from its continuing growth and improvement.

## References

[1] A. Agresti. Categorical Data Analysis. John Wiley  Sons, Inc., Hoboken, NJ, second edition, 2002.

[2] L. Brieman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. 1983.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In WWW, pages 107–117, 1998.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In ICML, pages 89–96, 2005.

[5] C. Burges, R. Ragno, and Q. Le. Learning to rank with nonsmooth cost functions. In NIPS, pages 193–200, 2007.

[6] D. Cossock and T. Zhang. Subset ranking using regression. In COLT, pages 605–619, 2006.

[7] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research, 4:933–969, 2003.

[8] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. The Annals of Statistics, 28(2):337–407, 2000.

[9] J. Friedman. Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 29(5):1189–1232, 2001.