Journal of Convergence Information Technology
Volume 5, Number 10. December 2010

Cost-Sensitive Support Vector Ranking for Information Retrieval

[1] Fengxia Wang, [2] Xiao Chang
[1, *2]*Department of Computer Science and Technology*
Xi'an Jiaotong University,710049, China
wangfengxia@gmail.com changxiao66@gmail.com
doi:10.4156/jcit.vol5. issue10.14

Abstract

In recent years, the algorithms of learning to rank have been proposed by researchers. However, in information retrieval, instances of ranks are imbalanced. After the instances of ranks are composed to pairs, the pairs of ranks are imbalanced too. In this paper, a cost-sensitive risk minimum model of pairwise learning to rank imbalanced data sets is proposed. Following this model, the algorithm of cost-sensitive supported vector learning to rank is investigated. In experiment, the standard Ranking SVM is used as baseline. The document retrieval data set is used in experiment. The experiment results show that the performance of cost-sensitive support vector learning to rank is better than Ranking SVM on two rank imbalanced data sets.

Keywords*: Cost-sensitive Learning, Learning to Rank, Imbalanced Data Set*

1. Introduction

Ranking is the central problem for information retrieval. The retrieval results can be rated by giving the grades to the results on the relevant to user's query. The similarity between user's query and document is used to rank the documents. The technique is proposed to eliminate the affixes and their effects on recognizing similar Persian documents [1]. Content-Time-based Ranking algorithm combines keywords, update time and content time of Web page into the ranking procedure [2]. In practice, the instances are ranked by mapped them to the score with ranking function. The task of learning to rank is to find a model on samples data, which can help to predict the order of new instances.

The problem of learning ranking function attracted much attention from machine learning community in recent years. This task is referred to as "Learning to Rank" in this field. "Learning to Rank" resides between multi-classes classification and metric regression in the area of supervised learning. In "Learning to Rank" problem, the samples are labeled with a set of discrete ranks which the size is larger than or equal to two. The task of learning to rank is to find a model on samples data set, which can predict the order of new instances. In information retrieval, for example, the retrieval results can be rated by giving the grades to the results on the relevant to user's query.

Many methods of "Learning to Rank" have been proposed. Most of these recent algorithms are based on the pairwise preference framework, in which instead of taking instances in isolation, instance pairs are used as instances in the learning process. The idea of pairwise preference learning to rank is to minimize the number of misordered pairs. The strategy of this approach is transfer ranking instances to classifying the pair of instance. The classification algorithm is the basis of pairwise approach of learning to rank. For example, supported vector machine is adopted to learn ranking function, which is called Ranking SVM [3,4]. Ranking SVM is a state-of-the-art method for learning to rank and has been empirically demonstrated to be effective. A boosting algorithm called RankBoost is developed in [5]. A probability loss of preference relation prediction is proposed and a two layers net is used to learn a ranking function in [6], which is named as RankNet. Nonlinear perceptron algorithm is proposed as a online algorithm of learning to rank [7]. Fidelity in physical field is employed as the loss function and a boosting algorithm is suggested to learn ranking function in [8]. Preference learning with Gaussian process is shown in [9]. A Learning to rank framework is proposed based on Bayesian perspective in [10]. Regularized least-squares approach is used to learn to ranking function in [11].

In practice, the data are imbalanced among ranks in many real world applications. In information retrieval, for example, the most relevant results occupy only a small part of candidates set. However,

the problem of imbalanced data set is not considered in most of the proposed algorithms of learning to rank. Only one approach is proposed to addressing the problem of imbalanced queries data with cost-sensitive supported vector learning approach [12]. However, the problem of imbalance among ranks is not considered in this approach. Furthermore, a binary classification model is used in it to learn ranking function, the new imbalance among preference pairs could be introduced into the model.

In this paper, the cost-sensitive support vector learning approach is proposed to learn the data set that is imbalanced among ranks. One class support vector model is employed to learn ranking function.

The performance of this approach proposed in this paper is compared with the standard Ranking SVM on document retrieval data set. The experimental results show that the performance of our approach is significant better than standard Ranking SVM to learn rank imbalanced data sets.

The rest of this paper is organized as follows. The problem analysis of imbalance among ranks is given in section 2. An approach of cost sensitive supported vector learning to rank imbalanced data is proposed in section 3. The experiments and results is given in section 4. The discussion is given in section 5. Section 6 is the conclusion of this paper.

2. Problem Analysis

2.1. Learning to Rank Statement

In preference learning problem, given an i.i.d training sample set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, each instance $\mathbf{x}_i \in \mathbb{R}^n$ is associated with a label y_i . $R = \{R_1, R_2, \ldots, R_k\}$ is a label space, the object in it can be ranked as $R_k \succ_R \cdots \succ_R R_2 \succ_R R_1$. ($\succ_R$ represents the order among ranks). R_i is the label of rank i.

Given π is a set of pairs generated by pairwise combining the elements in S , i.e. $\pi = \{\pi_i \mid i = 1, \ldots, M\}$. π_i is a couple (d_i, r_i) . Where d_i denotes a pair of instances $(\mathbf{x}, \mathbf{x}')$, and r_i denotes x in preference than x' or not. $\Upsilon(\cdot,\cdot)$ is a function of mapping the value of two instances $\mathbf{x}$ and $\mathbf{x}'$ to a preference label r which takes the form as

$$\Upsilon(y, y') = \begin{cases} +1, & y \succ y' \\ 0, & y = y' \\ -1, & y' \succ y \end{cases} \tag{1}$$

The goal of preference learning to rank is to learning a model from sample set π to rank the instances with right preference relation.

Assuming the model space of mapping object to real number is $\mathcal{H} = \{f : X \mapsto \mathbb{R}\}$. Each f in $\mathcal{H}$ creates an order $\succ_X$ in input space $X \subset \mathbb{R}^n$, according to the rule

$$\mathbf{x}_i \succ_X \mathbf{x}_j \Leftrightarrow f(\mathbf{x}_i) > f(\mathbf{x}_j) \tag{2}$$

which means that there is an unobservable latent function value $f(\mathbf{x}_i) \in \mathbb{R}$ associated with each training sample $\mathbf{x}_i$, and that the preference relation between any two instances dependents on the latent function values of them. The rules of deducing the preference label r_i^* of pair d_i takes the form

$$r_i^* = \begin{cases} +1, & f(\mathbf{x}) > f(\mathbf{x}') \\ -1, & f(\mathbf{x}) \le f(\mathbf{x}') \end{cases} \tag{3}$$

The task of learning to rank is to find a model f^* in space $\mathcal{H}$, which takes the minimum error of predicting the preference relation of the instances in training data set.

When r_i is not equal to r_i^*, the loss of the prediction error of model f can be denoted as the form $l_{\text{pref}}(d_i, r_i)$, where $l_{\text{pref}}(\cdot)$ is the loss function. The empirical risk of the model f predicting all pairs in set π is given as the form

$$\mathrm{R}_{\text{emp}}(f;\pi) = \sum_{\mathrm{i}=1}^{\mathrm{M}} l_{\text{pref}}(d_i, r_i) \Big/ \mathrm{M} \tag{4}$$

The goal of learning to rank is to find an optimal model f_{pref}^* which takes the minimum empirical risk of prediction error. The problem can be form as

$$f_{\text{pref}}^* = \underset{f}{\operatorname{argmin}}\ \mathrm{R}_{\text{emp}}(f;\boldsymbol{\pi}) \tag{5}$$

2.2. Rank Imbalance Analysis

The instance pair d_i can be labeled with the ordinal scale of one instance in the pair. The formal rule of labeling the pair is given in Definition 1.

Definition 1: A pair $d = \left(d^{(1)}, d^{(2)}\right)$ can be labeled with a ordinal scale r following the rule as the form

$$r' = \begin{cases} r_{d^{(1)}}, & \text{if } r_{d^{(1)}} \succ r_{d^{(2)}} \\ r_{d^{(2)}}, & \text{if } r_{d^{(1)}} \prec r_{d^{(2)}} \end{cases} \tag{6}$$

where $r_{d^{(i)}}$ is the label of i-th instance in the pair d. $\left(i \in \{1,2\}\right)$ The pair d can be called a pair of rank r'.

According to the label of the pairs in the set π, the empirical risk function $\mathrm{R}_{\text{emp}}(f;\boldsymbol{\pi})$ can be decomposed into $k-1$ sub-items as the form

$$\mathrm{R}_{\text{emp}}(f;\pi) = \frac{1}{\mathrm{M}} \sum_{j=2}^{k} \sum_{i=1}^{N_{\mathrm{j}}} l_{\text{pref}}(d_i, r_i) \tag{7}$$

where N_j denotes the number of instances of rank j $\left(j = 2,...,\mathrm{k}\right)$, k is the number of ranks.

In real world data set, the number of instances of ranks is different. Usually, the instance of the "important" rank is fewest, followed by 'possible important' rank, and the instance of the 'no important' rank is most. After these instances are combined into pairs and labeled following Definition 1. The pairs of ranks will also be imbalance. In this case, the empirical risk of the most important rank will occupy a smaller proportion in the total risk $\mathrm{R}_{\text{emp}}(f;\boldsymbol{\pi})$. It is possible that the optimal result f_{pref}^* of (5) will bias to the rank of occupying the larger proportion in the total risk. As a result, minimum prediction error could not be attained on the sample pairs of most important rank.

Given a sample set S, for example, the instances in it are labeled with three ordinal scales: 1, 2 and 3. Rank 3 is the most important rank, followed by rank 2 and rank 1 is "no important" rank. In the set S, 3, 2 and 1 instances are labeled with ordinal scale 1, 2 and 3 respectively. After the instances in it are combined into pairs, 6 and 5 pairs will be assigned to rank 2 and 3 respectively following Definition 1. In this case, the number of pairs of rank 3 is fewer than that of rank 2, i.e. $N_3 < N_2$. It is mean that there is a bias to the risk of pairs of rank 2 in risk function.

Unfortunately, the most of real world data sets are imbalanced. Therefore, researching the approach to improve the optimization result in this case should be a valuable work.

3. Proposed Cost-Sensitive Ranking Learning Approach

3.1. Cost-Sensitive Risk Model of Learning to Rank

The strategy of learning imbalance data is to modify the error cost of pairs of ranks. Following the Definition 1, the data set labeled with k ranks is split into k-1 pairs subsets. The error cost of pairs of a rank is adjusted by the proportion occupied by them in $\boldsymbol{\pi}$. The cost-sensitive risk model is written as the form

$$\mathrm{R}_{\mathrm{emp}}\left(f;\boldsymbol{\pi}\right)=\frac{1}{\mathrm{M}}\sum_{j=2}^{k}\eta_j\sum_{i=1}^{N_j}l_{\mathrm{pref}}\left(d_i,r_i\right) \tag{8}$$

where η_j is a cost parameter of pairs of rank j, which is used to adjust the error cost of pairs of rank j. The value of cost parameters η_r can be computed as the form

$$\eta_j=e_j\cdot\frac{N_m}{N_j},\ j=2,...,\mathrm{k} \tag{9}$$

where e_j is a enlargement factor to the cost of pairs of rank j, rank m is the rank with the most pairs. The value of m can be obtained following the form

$$\mathrm{m}=\underset{r}{\mathrm{argmax}}\left(N_r\right),\ \mathrm{r}=2,...,\mathrm{k} \tag{10}$$

The error risk of pairs of a rank in $\mathrm{R}_{\mathrm{emp}}\left(f;\boldsymbol{\pi}\right)$ can be adjusted by changing the value of e_j.

3.2. Cost-Sensitive One-Class Support Vector Learning to Rank

Following the rule given in (1), the pairs are labeled as two classes. However, the pairs can be assigned into only one class by changing the order of two instances in the couple. In this case, only the data of one class will be learned. Therefore, the cost-sensitive one-class support vector learning to rank is given as the form

$$\min_{w}\ \sum_{j=2}^{k}\sum_{i=1}^{N_j}\eta_j\left[1-\left\langle w,\Phi\left(d_i^{(1)}\right)-\Phi\left(d_i^{(2)}\right)\right\rangle\right]_{+}+\lambda\left\|w\right\|^2 \tag{11}$$

where $\Phi\left(\cdot\right)$ map a sample instance $d_i^{(n)}$ from input space into feature space .

The model in (11) is equal to a quadratic programming model which takes the form

$$\min_{w}\ \frac{1}{2}\left\|w\right\|^2+\sum_{m=1}^{\mathrm{M}}C_m\cdot\xi_m \tag{12}$$

Subject to

$$\left\langle w,\Phi\left(d_i^{(1)}\right)-\Phi\left(d_i^{(2)}\right)\right\rangle\geq 1-\xi_m,\quad m=1,\cdots,\mathrm{M} \tag{13}$$

$$\xi_m \geq 0, \quad m = 1, \cdots, \mathrm{M} \tag{14}$$

which is used in computation.

Proposition 1: The problems in (11) and (12)~(14) are equivalent, when $C_m = \frac{\eta_j}{2\lambda}$ where the pair d_i belongs to rank j.

The Lagrange method is used to solve the quadratic programming problem (12)~(14). The Lagrange dual form of problem (12)~(14) takes the form

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i=1}^{\mathrm{M}}\sum_{j=1}^{\mathrm{M}}\alpha_i\alpha_j\left\langle \Phi\left(d_i^{(1)}\right)-\Phi\left(d_i^{(2)}\right), \Phi\left(d_j^{(1)}\right)-\Phi\left(d_j^{(2)}\right)\right\rangle \tag{15}$$

Subject to

$$0 \leq \alpha_i \leq C_i, \ i = 1, \cdots, \mathrm{M} \tag{16}$$

$$\sum_{i=1}^{\mathrm{M}}\alpha_i = 1 \tag{17}$$

where C_i is the upper limit of the value of α_i. The upper limit C_i of Lagrange coefficient of the pairs of a rank r increases with the growth of the enlargement factor e_r. It is possible that a larger value is assigned to α_i that corresponds to a large C_i.

According to the Lagrange method, the ranking function can be expressed as the form

$$f^*(\mathbf{x}) = \sum_{i=1}^{m} a_i^* \left\langle \Phi(\mathbf{x}), \Phi\left(d_i^{(1)}\right)-\Phi\left(d_i^{(2)}\right)\right\rangle \tag{18}$$

where a_i^* can be seemed as the weight of a sample pair d_i, m is the number of support sample pairs. If a Lagrange coefficient α_i obtains a larger value in the optimization process means that the pair d_i corresponding to it is an important pair in the prediction model.

4. Experiments and Results

4.1. Experiment Setting

In experiments, the performance of the approach proposed in this paper is compared with standard Ranking SVM. The standard Ranking SVM is named as RankSVM. The algorithm of cost-sensitive support vector learning ranking function from imbalanced data set is named as CSRankSVM.

The linear kernel function is employed in two algorithms in experiments, which takes the form $\kappa\left(\mathbf{x}, \mathbf{x}'\right) = \mathbf{x}^{\mathrm{T}} \square \mathbf{x}'$. The trade-off parameter λ is set to 0.5 for two algorithms.

CSRankSVM and RankSVM are trained and tested on the document retrieval data set OHSUMED.

The normalized discounted cumulative gain (NDCG) [13] is used as evaluation measure, which has been widely used by researchers in recent years. NDCG can be used to evaluate the performance of ranking method on the data set that is labeled with more than two ranks.

$$\mathrm{NDCG@k} = \mathrm{N(k)}\sum_{\mathrm{j}=1}^{\mathrm{k}}\frac{2^{\mathrm{r(j)}}-1}{\log(1+\mathrm{j})}$$

where $\mathrm{N(k)}$ is the NDCG at k-th position of ideal ranking list. It is used as a normalization factor of the NDCG at k of ranking list of prediction result.

4.2. Experiments on Document Retrieval Data

OHSUMED [14] is a data set for document retrieval research. It has been processed and included in LETOR [15], a Benchmark data set build by MSAR for ranking algorithm research. This data set has been used in information filtering task of TREC 2000. The relevance judgments of documents in OHSUMED are either 'd' (definitely relevant), 'p' (possibly relevant), or 'n' (not relevant). Rank 'n' has the largest number of documents, followed by 'p' and 'd'. The original OHSUMED collection consists of 348,566 records from 270 medical journals. There are 106 queries. For each query, there are a number of documents associated.

The OHSUMED has been collected into a Benchmark data set LETOR for ranking algorithm research. In this data set, each instance is represented as a vector of features, determined by a query and a document. Every vector consists of twenty-five features. The value of features has been computed.

The twenty folds experimental data set is obtained by running following strategy twenty times: selecting the instances of two queries randomly as training data, the instances other queries as test data. The evaluation results are average results of running experiments on twenty folds.

The average instances number of three ranks in twenty folds is given in Table 1. The rank 2, 1 and 0 denote the rank of 'definitely relevant', 'possibly relevant' and 'not relevant' respectively. The rank 0 has the most instances, followed by rank 1 and 2.

Table 1. Average instance number of ranks of twenty folds training data of OHSUMED

	Rank 0	**Rank 1**	**Rank 2**
Average Instance Number	38	4	3

Following the Definition 1, instance of three ranks are combined to the pairs of two ranks. The average number of pairs of rank 1 of twenty folds is 159.5. The average number of pairs of rank 2 of twenty folds is 138.3. The pairs of rank 2 are fewer than that of rank 1. All of the pairs of Rank 1 and Rank 2 are assigned to one class.

The training pairs are fallen into two most relevant ranks. The error risk of prediction can be decomposed into two parts. A two dimensions enlargement factor vector $\mathbf{E} = (e_1, e_2)$ is used in this experiment. e_1 is the enlargement factor of rank 1. e_2 is enlargement factor of rank 2, i.e. 'definitely relevant' rank. Document retrieval problem is focused on the prediction precision of 'definitely relevant' rank. In experiment, therefore, e_1 is set to one and e_2 is adjusted from one to five. The experimental results are given in Figure 1. It can be seen that when e_2 is set to from 1 to 5 the value of NDCG of CSRankSVM keep higher than that of RankSVM significant.

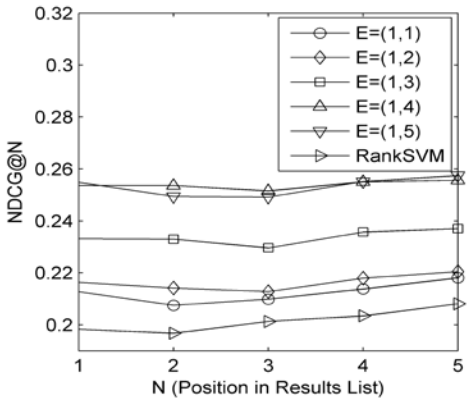

Figure 1. The NDCG evaluation results of RankSVM and CSRankSVM running on OHSUMED. To CSRankSVM e_2 is set to from 1 to 5. The NDCG (y-axis) versus position of prediction results list (x-axis).

The value of NDCG is increased when e_2 is set to from one to four. When, however, the value of e_2 is changed from four to five, the change of the value of NDCG is not monotony.

When e_2 is increased from one to four, the value of NDCG increasing with the augment of e_2 is significant in Figure 2. When e_2 is set to larger than four the change of NDCG is not significant.

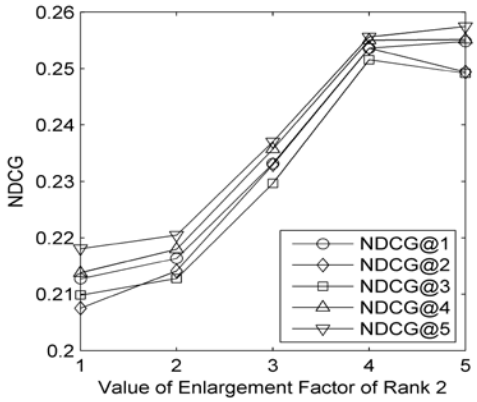

Figure 2. The NDCG evaluation results of CSRankSVM running on OHSUMED. NDCG at 1 to 5 (y-axis) versus e_2 (x-axis) of CSRankSVM.

5. Discussion

The real world data set used in the experiments is imbalanced. In Figure 1 it is clear that the learning results of CSRankSVM are better than standard RankSVM, where e_2 is set to one. In this condition, the number of loss items of any two ranks in risk is equal.

From Figure 2 we can see that the evaluation results do not increase monotony with the increase of e_2 . When e_2 is large than three the performance of algorithm will can not be improved significantly.

In experiment, it can be observed that the weight a_i of the sample pair d_i is not monotony increase with the increase of e_2 . When e_2 is larger than a certain value, the higher value can not be assigned to a_i , even though the higher value is set to the upper limit of a_i by enlarging e_2 . This means that the prediction model will not be improved significantly when e_2 is larger than a certain value.

According to the analysis given above, it can be concluded that the performance of algorithm of learning imbalance data set can be improved when all of elements in vector E are set to one. If the value of e_i is set to larger than one the risk function will have a bias to rank i . The instance of a rank could be predicted more precision by enlarging the enlargement factor.

In this paper, the effect of the cost sensitive model of learning to rank imbalanced data sets is given. In practice, the optimal setting to enlargement factor vector E can be found by adopting cross-validate strategy.

6. Conclusions and Future Work

In this paper, a cost-sensitive risk minimum model is proposed to learning rank function from rank imbalanced data sets. In this model, the enlargement factors are used to adjust the error cost of ranks. Following this model, a cost-sensitive support vector learning approach is developed. The performance of the approach proposed in this paper is compared with standard Ranking SVM in experiment. The experimental results on document retrieval data set show that the performance of our approach is better than that of standard Ranking SVM to learn imbalanced data sets.

In this paper, a simple way is proposed to improve the Ranking SVM to learn rank imbalanced data sets more efficient. In the future work, some advanced techniques [16] [17] which have been used in SVM

classification imbalanced data sets will be employed to learning ranking function from rank imbalanced data set.

7. References

[1] Kashefi Omid, Mohseni Nina, Minaei Behrouz, "Optimizing document similarity detection in Persian information retrieval", Journal of Convergence Information Technology, vol. 5, no. 2, pp: 101-106, 2010.
[2] Jin Peiquan, Li Xiaowen, Chen Hong, Yue Lihua, "CT-Rank: A Time-aware Ranking Algorithm for Web Search", Journal of Convergence Information Technology, vol. 5, no. 6, 2010.
[3] Herbrich Ralf, Graepel Thore, Obermayer Klaus, "Support vector learning for ordinal regression", In the 9th International Conference on Artificial Neural Networks (ICANN), pp: 97-102, 1999.
[4] Joachims Thorsten, "Optimizing Search Engines using Clickthrough Data", In the ACM Conference on Knowledge Discovery and Data Mining, pp: 133-142, 2002.
[5] Freund Yoav, Iyer Raj, Schapire Robert E., Singer Yoram, "An efficient boosting algorithm for combining preferences", In the 15th International Conference on Machine Learning, pp, 1998.
[6] Burges Chris, Shaked Tal, Renshaw Erin, Lazier Ari, Deeds Matt, Hamilton Nicole, Hullender Greg, "Learning to rank using gradient descent", pp: 89-96, 2005.
[7] Chen Xue-Wen, Wang Haixun, Lin Xiaotong, "Learning to rank with a novel kernel perceptron method", In International Conference on Information and Knowledge Management, pp: 505-512, 2009.
[8] Tsai Ming-Feng, Liu Tie-Yan, Qin Tao, Chen Hsin-Hsi, Ma Wei-Ying, "FRank: A Ranking Method with Fidelity Loss", In The 30th Annual International ACM SIGIR Conference, pp, 2007.
[9] Chu Wei, Ghahramani Zoubin, "Preference Learning with Gaussian Processes", In 22nd International Conference on Machine Learning, pp: 137-144, 2005.
[10] Kuo Jen-Wei, Cheng Pu-Jen, Wang Hsin-Min, "Learning to rank from Bayesian decision inference", In International Conference on Information and Knowledge Management, pp: 827-835, 2009.
[11] Pahikkala T., Tsivtsivadze E., Airola A., Boberg J., Salakoski T., "Learning to rank with pairwise regularized least-squares", In the 30th International Conference on Research and Development in Information Retrieval -Workshop on Learning to Rank for Information Retrieval, pp: 27-33, 2007.
[12] Xu J., Cao Y., Li H., Huang Y. L., "Cost-sensitive learning of SVM for ranking", In Europe Conference on Mahcine Learning 2006, pp: 833-840, 2006.
[13] Kekalainen Jaana, "Binary and graded relevance in IR evaluations: comparison of the effects on ranking of IR systems", Inf Process Manage, vol. 41, no. 5, pp: 1019-1033, 2005.
[14] Hersh W., Buckley C., Leone T. J., Hickam D., "OHSUMED: an interactive retrieval evaluation and new large test collection for research", pp: 192-201, 1994.
[15] Liu T. Y., Xu J., Qin T., Xiong W., Li H., "Letor: Benchmark dataset for research on learning to rank for information retrieval", In SIGIR 2007 Workshop on Learning to Rank for Information Retrieval, 2007.
[16] Raskutti B., Kowalczyk A., "Extreme re-balancing for SVMs: a case study", ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp: 60-69, 2004.
[17] Tao Q., Wu G. W., Wang F. Y., Wang J., "Posterior probability support vector Machines for unbalanced data", IEEE Transactions on Neural Networks, vol. 16, no. 6, pp: 1561-1573, 2005.