ISSN 0005-1179 (print), ISSN 1608-3032 (online), Automation and Remote Control, 2025, Vol. 86, No. 4, pp. 358–363.

Using Ensembles with Enhanced Divergence in Forecast Space in Recommender Systems

O. V. Senko[*,a], **A. A. Dokukin**[*,b], **and F. A. Melnik**[**,c]

[*] *Federal Research Center "Computer Science and Control", Russian Academy of Sciences, Moscow, Russia*
[**] *M.V. Lomonosov Moscow State University, Moscow, Russia*
e-mail: [a] *senkoov@mail.ru,* [b] *dalex@ccas.ru,* [c] *melnik.tedor@gmail.com*

Received November 29, 2024
Revised January 9, 2025
Accepted January 14, 2025

Abstract—The paper considers the divergent decision forest method based on achieving a higher divergence in the forecast space compared to the standard random decision forest. It is based on including a new tree T_x in the ensemble at each step, which is constructed based on minimizing a special functional, which is the difference between the squared error of T_x and the squared divergence of the forecasts T_x and the current ensemble. The method is a development of similar previously developed methods that are intended for predicting numerical variables. The paper presents the results of applying the divergent decision forest method to solving classification problems that arise when creating recommender systems. The paper investigates the dependence of the forecast efficiency on the tree depth and one of the key parameters of the algorithm that regulates the contribution of two components to the minimized functional. Studies have shown that the accuracy of the proposed technology significantly exceeds the accuracy of the random decision forest and is close to the accuracy of the CatBoost method.

Keywords: ensemble method, machine learning, recommender systems

DOI: 10.31857/S0005117925040062

1. INTRODUCTION

Methods based on the use of ensembles of regression or decision trees are one of the main branches of modern machine learning and are actively used in solving various applied problems [1]. Two popular directions of ensemble algorithms can be distinguished: random forests [2] and gradient boosting [3]. For both technologies, methods have been developed to solve both automatic classification problems and numerical variable prediction problems. When predicting a target numerical variable Y based on features $X_1, \ldots, X_n$, training is performed using the sample $S = \{s_1 = (y_1, x_1), \ldots, s_m = (y_m, x_m)\}$, where y_j and x_j are the values of the target variable Y and the vector of feature values $X_1, \ldots, X_n$ of the object s_j, respectively. Classification problems with two disjoint classes can also be viewed as binary numeric variable prediction problems with training from a similar type of sample. Let us formulate the main differences between gradient boosting-based methods and random forests. In the random forest method, individual trees are built independently. In this case, the tree at step k is aimed at predicting the target variable Y and is constructed using the sample S_k obtained by the bagging procedures [4] and the random subspace method [5]. The bagging procedure comes down to sample selection with replacement from S. The random subspace method assumes the use of a random subset of the original feature set $X_1, \ldots, X_n$ of a fixed size when training each new random forest tree. The output forecast of $\hat{A}_k$, consisting of k trees, is calculated as the average forecast over all trees included in the ensemble, that is,

$\hat{A}_k = \frac{1}{k}\sum_{j=1}^{k} A_j$. In the gradient boosting method, the optimal algorithm is sought as a linear combination of regression trees $\hat{A}_N = T_0 + \alpha_1 \times T_1 + \ldots + \alpha_N \times T_N$, where the original algorithm T_0 usually calculates a forecast identically equal to the average value of Y. At each step, a new term $\alpha_k \times T_k$ is added to the current linear combination $\hat{A}_{k-1} = A_0 + \alpha_1 \times T_1 + \ldots + \alpha_{k-1} \times T_{k-1}$. The choice of the latter is made based on minimizing the loss function for the linear combination $\hat{A}_k = T_0 + \alpha_1 \times T_1 + \ldots + \alpha_k \times T_k$. That is achieved by using the gradient descent procedure.

Let $L(\hat{A}_{k-1}, S)$ be some loss function depending on the predictions computed by $\hat{A}_{k-1}$ and the true values of Y of objects S. Let us associate the predictions computed for objects S with the variables $z_1, \ldots, z_m$. According to the gradient descent method, the expected minimum L will be achieved for the prediction at the kth step for object s_j, provided

$$\hat{A}_k(s_j) = \hat{A}_{k-1}(s_j) - \eta \frac{\partial L}{\partial z_j}\bigg|_{z_j = \hat{A}_{k-1}(s_j)}.$$

However, the forecast in the specified form cannot be calculated for objects for which the true value of y_j is unknown, since the loss function and its partial derivatives are defined only for known values of Y. Therefore, the authors of the method proposed to use instead of the values of g_j their forecasts calculated using a regression tree trained on a sample of $\{(g_1, x_1), \ldots (g_m, x_j)\}$. This tree becomes the tree T_k added to the linear combination. When training T_k, the bagging and the random subspace procedures are used.

Currently, there are several modifications of gradient boosting, including XGBoost [6], LightGBM [7], CatBoost [8]. In contrast to gradient boosting, conservative versions of the random forest method are widespread, i.e. the method remains virtually unchanged since its creation. At the same time, there is no convincing evidence that the ensemble generation scheme used in random forests is actually optimal when using simple averaging as an aggregation procedure.

2. CONSTRUCTION OF FORESTS WITH ENHANCED DIVERGENCE

In the [9–11] a new approach was proposed aimed at constructing an ensemble based on the condition of minimizing forecast losses, calculated as the average forecasts for the ensemble. In other words, the problem of selecting trees for the ensemble in such a way that the losses were as minimal as possible when using $\hat{A}_k$ as a forecast was set. In this case, the usual root-mean-square error was used as a loss functional:

$$L(S, A) = \frac{1}{m}\sum_{j=1}^{m}\left(y_j - \hat{A}_k(x_j)\right)^2.$$

It was shown that the mean squared error for the algorithm A_k is calculated by the formula

$$L(S, \hat{A}_k) = \frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{k}(y_j - T_i(x_j))^2 - \frac{1}{km}\sum_{j=1}^{m}\sum_{i=1}^{k}\left(\hat{A}_k(x_j) - T_i(x_j)\right)^2. \quad (1)$$

One can pose the problem of finding such a set of trees $T_1, \ldots, T_k$ that achieves a minimum of losses (1). This problem is extremely labor-intensive. However, a simple heuristic approach can be used, when at each step a tree is added to the ensemble that simultaneously approximates the dependence and is maximally distant from the current ensemble in terms of the calculated forecasts. To implement this approach, in [9–11] it was proposed to add a tree T_k at step k, in which the following functional would be as minimal as possible

$$Q(T_k, \hat{A}_k, \mu) = \frac{1}{m}\sum_{j=1}^{m}(y_j - T_k(x_j))^2 - \mu\frac{1}{m}\sum_{j=1}^{m}\left(\hat{A}_k(x_j) - T_k(x_j)\right)^2, \quad (2)$$

where μ is a parameter from the interval $(0,1)$. The functional $Q(T_k, \hat{A}_k, \mu)$ can be rewritten as

$$Q(T_k, \hat{A}_k, \mu) = \frac{1}{m}\sum_{j=1}^{m}(y_j - T_k(x_j))^2 - \theta\frac{1}{m}\sum_{j=1}^{m}\left(\hat{A}_k(x_j) - T_k(x_j)\right)^2,\ \theta = \frac{k^2\mu}{(k+1)^2}. \tag{3}$$

Note that the minimum of the functional (2) is achieved if each point x_j is associated with a forecast t_j^o equal to the value of the auxiliary variable t_j which minimizes the function

$$(y_j - t_j)^2 - \theta\left(\hat{A}_{k-1}(x_j) - t_j\right)^2.$$

It is easy to show that the necessary conditions for an extremum are satisfied in the case when

$$t_j^o = \frac{y_j - \theta\hat{A}_{k-1}(x_j)}{1-\theta}. \tag{4}$$

It is obvious that $\theta \to 1$ for $k \to \infty$ and $\mu = 1$, which leads to instability of the estimates by formula (4). Thus, the additional factor $\mu \in (0,1)$ is introduced. To find the optimal ensemble, an approach was proposed in [10, 11], in which at step k a new tree T_k is added to the ensemble, predicting the values t_j^o calculated by formula (4) depending on the vector descriptions. The tree T_k is trained using the sample $\{(t_1^o, x_1), \dots (t_m^o, x_m)\}$. The use of the hyperparameter μ, which affects the efficiency of the constructed ensembles, is a heuristic technique. There is no theoretically justified method for calculating the optimal value of μ. It is assumed that the optimal value of μ can be selected as a result of experiments, which is one of the goals of this study.

3. APPLICATION FOR AUTOMATIC CLASSIFICATION

The approach described above is based on the use of quadratic loss functions and is intended for solving problems of predicting numerical variables. However, the method can also be used to solve automatic classification problems. When solving the latter, cross-entropy is usually used to estimate the accuracy of the approximation of a dependence, i.e., a value proportional to the minus logarithm of the likelihood function calculated based on the Bernoulli distribution. In the case of a binary classification problem, the target value Y takes values from the set $\{0,1\}$, where the equality $Y = 1$ indicates membership in the target class K. Suppose that some tree T calculates the probabilities of belonging to class K of objects from $S = \{s_1 = (y_1, x_1), \dots, s_m = (y_m, x_m)\}$. Let p_j be the probability of object s_j belonging to the target class, calculated by tree T from its description x_j. The cross-entropy loss for tree T on sample S is estimated by the formula

$$L(T,S) = -\ln\prod_{j=1}^{m} p_j^{y_j}(1-p_j)^{1-y_j} = -\sum_{j=1}^{m}[y_j\ln p_j + (1-y_j)\ln(1-p_j)]. \tag{5}$$

The losses according to the formula (5) are an analogue of the quadratic losses for the tree T at the sample S and correspond to the left term in the formula (2). For the estimation involving the deviation of the new tree T_k from the ensemble A_k, the following approach can be proposed. Let p_j^k be the probability of the object s_j belonging to the target class, calculated by the tree T_k from its description x_j; $\hat{p}_j^k$ be the probability of the object s_j belonging to the target class, calculated by the ensemble A_k. The deviation of the new tree from the ensemble is estimated by the formula

$$D(T_k, A_k, S) = -\ln\prod_{j}^{m}(p_j^k)^{\hat{p}_j^k}(1-p_j^k)^{1-\hat{p}_j^k} = \sum_{j=1}^{m}[-\hat{p}_j^k\ln p_j^k - (1-\hat{p}_j^k)\ln(1-p_j^k)].$$

To find the optimal values of probabilities p_j^k, the same approach can be used as in the case of quadratic losses. At the first stage, the values $p_1^o, \dots, p_m^o$ are sought for which the minimum of the

following functional is achieved

$$Q(T_k, A_k, S) = L(T, S) - \mu D(T_k, A_k, S).$$

Next, a new tree T_k is trained using the sample

$$\{(p_1^o, x_1), \ldots, (p_m^o, x_m)\}.$$

The ensemble A_k at step k is unknown. Therefore, instead of $Q(T_k, A_k, S)$, a similar functional can be used

$$Q(T_k, A_{k-1}, S) = L(T, S) - \mu D(T_k, A_{k-1}, S). \tag{6}$$

It is easy to show that the minimum of the functional (6) is achieved at

$$p_j^o = \frac{y_j - \mu \hat{A}_{k-1}(x_j)}{1 - \mu}. \tag{7}$$

Unfortunately, using (7) leads to a violation of the condition $p_j^o \in [0, 1]$. This condition can be preserved by moving to the problem of conditional optimization. However, the latter leads to a significant complication of the algorithm. A simpler solution is based on the use of formula (7) with subsequent training of a new tree from the sample

$$\{(p_1^o, x_1), \ldots, (p_m^o, x_j)\}. \tag{8}$$

To separate the classes, the estimate of the value p_j is compared with a threshold, for the search of which ROC analysis tools are used. We will further call this method a divergent decision forest (DDF).

4. APPLICATION TO RECOMMENDER SYSTEMS

The method proposed in the previous sections depends on a set of hyperparameters, which include both the multiplier μ and the hyperparameters used in the construction of individual trees. The successful application of most machine learning methods depends on the selected values of the hyperparameters. At the same time, accurate theoretical estimates for the choice of hyperparameters are usually absent. Therefore, their optimal values have to be sought through experiments with data. The goal of this work was an experimental search for optimal values of hyperparameters for a divergent decision forest. The studies were carried out in the context of solving problems of predicting user preferences that arise when creating recommender systems. The choice of this class of problems is associated with both the widespread use of recommender systems in various sectors of the economy, and with the intensive use of machine learning methods in this area [12]. Three problems of assessing the preferences of Internet users when choosing a computer game and two problems of assessing the preferences of Internet users when choosing a sticker in social networks were considered. The assessment was based on information about the interaction of users with the corresponding catalogs. The HR5 (hitrate at 5) indicator was used for assessment, that is the proportion of users for whom at least one relevant object was among the first 5 recommendations. The following notations are used: k is the number of users, p is the number of objects, n is the number of features and M is the number of rows in the training sample. The characteristics of the considered problems are presented in the Table 1.

The aim of the work was to study the dependence of the algorithm accuracy in terms of the HR5 indicator on the depth of the trees and the parameter μ. The results for the game0 and sticker1 tasks are presented in the Table 2 and 3, respectively. The value $\mu = 0$ corresponds to the usual random decision forest (RF) model.

Table 1. Characteristics of the problems considered

problem	k	p	n	M
game0	414	187	41	4132
game1	1706	190	41	16 327
game2	3888	398	55	29 065
sticker1	2685	118	40	27 032
sticker2	5942	197	41	44 613

Table 2. Relationship between HR(5) and tree depth and parameter μ for the game0 problem

	$\mu = 0.0$	$\mu = 0.25$	$\mu = 0.5$	$\mu = 0.75$	$\mu = 0.9$
$depth = 3$	0.584	0.594	0.599	0.613	0.640
$depth = 5$	0.596	0.604	**0.623**	**0.630**	**0.657**
$depth = 7$	0.591	0.606	0.621	0.621	0.623
$depth = 9$	0.592	**0.618**	0.606	0.606	0.570
$depth = 11$	**0.606**	0.601	0.606	0.611	0.493

Table 3. Relationship between HR(5) and tree depth and parameter μ for the sticker1 problem

	$\mu = 0.0$	$\mu = 0.25$	$\mu = 0.5$	$\mu = 0.75$	$\mu = 0.9$
$depth = 3$	0.447	0.449	0.448	0.458	0.514
$depth = 5$	0.482	0.488	0.498	**0.516**	**0.527**
$depth = 7$	0.475	0.491	**0.501**	0.508	0.513
$depth = 9$	0.479	0.492	0.500	0.507	0.477
$depth = 11$	**0.482**	**0.494**	0.499	0.494	0.467

Table 4. The best HR(5) value for each of the models

	RF	DDF ($\mu \neq 0$)	catboost
games0	0.606	0.657	**0.664**
games1	0.642	**0.654**	0.637
games2	0.513	0.550	**0.562**
stickers1	0.482	**0.527**	0.513
stickers2	0.337	0.372	**0.385**
mean	0.516	0.552	0.5522

The tables show that when the tree depth does not exceed 7 the HR(5) increases with the growth of μ. At the same time, when the tree depth is greater than 7, the HR(5) indicator decreases.

Table 4 presents a comparison of the proposed technology with standard random decision forests, as well as with the CatBoost method. Standard decision forests correspond to $\mu = 0$. The table shows that the accuracy in terms of the HR5 metric for the proposed technology significantly exceeds the accuracy of the random decision forest and is close to the accuracy of the CatBoost method.

5. CONCLUSION

A divergent decision forest method has been developed for solving binary classification problems. The method is based on the development of approaches previously proposed for solving regression problems [9–11]. Using the example of problems related to recommender systems, a study of the

divergent decision forest (DDF) method was conducted. The method is based on achieving higher divergence in the prediction space in comparison with the standard random decision forest.

A significant dependence of the efficiency on the tree depth and on the coefficient μ, which regulates the relative contributions of the components responsible for the approximation of the target variable and the divergence of ensembles, is shown. At the same time, with a small tree depth, the accuracy of the algorithm increases with an increase in μ. Such an effect is not observed with a large tree depth. However, in general, DDF outperforms the standard random decision forest corresponding to $\mu = 0$. Experiments have shown that the efficiency of DDF is close to the well-known boosting model CatBoost. The DDF method is based on sequential generation of tree sets. Therefore, its Performance is close to the performance of conventional random forests and gradient boosting variants. One of the problems of the method is the choice of the optimal value of the hyperparameter μ. This problem can be solved using known tools for choosing hyperparameters. However, experiments indicate that higher efficiency is achieved with high values of the hyperparameter, that is, with μ equal to 0.8 or 0.9. Unlike the methods of decision and regression trees, which have high transparency and interpretability, methods using large ensembles of trees, unfortunately, lose these properties. The latter also applies to the DDF method. This drawback can be compensated by various known methods for achieving interpretability. Data mining methods and statistical analysis can also be used to increase transparency.

REFERENCES

1. Hastie, T., Tibshirani, R., and Friedman, J., *The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer Series in Statistics*, New York: Springer, 2009.
2. Breiman, L., Random Forests, *Machine Learning*, 2001, vol. 45, no. 1, pp. 5–32.
3. Friedman, J., Stochastic gradient boosting, *Computational Statistics & Data Analysis*, 2002, vol. 38, no. 4, pp. 367–378.
4. Breiman, L., Bagging predictors, *Machine Learning*, 1996, no. 24, pp. 123–140.
5. Tin Kam Ho, The random subspace method for constructing decision forests, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998, vol. 20, no. 8, pp. 832–844.
6. Chen, T. and Guestrin, C., XGBoost: A Scalable Tree Boosting System, *Proceedings of the 22nd ACM SIGKDD International Confer*, 2016, pp. 785–794.
7. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y., LightGBM: A Highly Efficient Gradient Boosting Decision Tree, *Advances in Neural Information Processing Systems*, 2017.
8. Hancock, J.T. and Khoshgoftaar, T.M., CatBoost for big data: an interdisciplinary review, *J. Big Data*, 2020, vol. 7, no. 94.
9. Zhuravlev, Yu.I., Senko, O.V., Dokukin, A.A., Kiselyova, N.N., and Saenko, I.A., Two-Level Regression Method Using Ensembles of Trees with Optimal Divergence, *Doklady Mathematics*, 2021, vol. 103, pp. 1–4.
10. Dokukin, A.A. and Sen'ko, O.V., A New Two-Level Machine Learning Method for Evaluating the Real Characteristics of Objects, *Journal of Computer and Systems Sciences International*, 2023, vol. 62, no. 4, pp. 607–614.
11. Senko, O.V., Dokukin, A.A., Kiselyova, N.N., Dudarev, V.A., and Kuznetsova, Yu.O., New Two-Level Ensemble Method and Its Application to Chemical Compounds Properties Prediction, *Lobachevskii Journal of Mathematics*, 2023, vol. 44, no. 1, pp. 188–197.
12. Roy, D. and Dutta, M., A systematic review and research perspective on recommender systems, *J. Big Data*, 2022, vol. 9, no. 59.

This paper was recommended for publication by V.A. Kalyagin, a member of the Editorial Board