UDC 004.9 **doi: 10.20998/2522-9052.2018.3.03**

E. Tolstoluzka, B. Parshentsev, O. Moroz

V. N. Karazin Kharkiv National University, Kharkiv, Ukraine

PARALLEL IMPLEMENTATION OF THE METHOD OF GRADIENT BOOSTING

The issue of machine learning has been paying more attention in all areas of information technology in recent times. On the one hand, this is due to the rapid growth of requirements for future specialists, and on the other - with the very rapid development of information technology and Internet communications. One of the main tasks of e-learning is the task of classification. For this type of task, the method of machine learning called gradient boost is very well suited. Grading boosting is a family of powerful machine learning algorithms that have proven significant success in solving practical problems. These algorithms are very flexible and easily customized for the specific needs of the program, for example, they are studied in relation to different loss functions. The idea of boosting is the iterative process of sequential building of private models. Each new model learns based on information about errors made in the previous stage, and the resulting function is a linear combination of the whole ensemble of models, taking into account minimization of any penalty function. The mathematical apparatus of gradient boosting is well adapted for the solution of the classification problem. However, as the number of input data increases, the issue of reducing the construction time of the ensemble of decision trees becomes relevant. Using parallel computing systems and parallel programming technologies can produce positive results, but requires the development of new methods for constructing gradient boosting. The article reveals the main stages of the method of parallel construction of gradient boosting for solving the classification problem in e-learning. Unlike existing ones, the method allows to take into account the features of architecture and the organization of parallel processes in computing systems with shared and distributed memory. The method takes into account the possibility of evaluating the efficiency of building an ensemble of decision trees and parallel algorithms. Obtaining performance indicators for each iteration of the method helps to select the rational number of parallel processors in the computing system. This allows for a further reduction of the completion time of the gradient boosting. The simulation with the use of MPI parallel programming technology, the Python programming language for the architecture of the DM-MIMD system, confirms the reliability of the results. Here is an example of the organization of input data. Presented by Python is a program for constructing gradient boosting. The developed visualization of the obtained estimates of performance indicators allows the user to select the necessary configuration of the computing system.

Keywords: gradient boosting; parallel algorithm; decision tree; estimation of parallel algorithm efficiency; e-learning.

Statement of the problem

Recently there has been a rapid development of computing power and this development has significantly increased the ability to collect, process and analyze information. These opportunities are now used in a variety of subject areas, such as advertising, medicine and trade [1]. The section of science that deals with the research of algorithms for analyzing and forecasting data using information systems is called machine learning. Machine learning allows you to work with parameters that are not manually set, but appear in a semi-automatic mode thanks to the primary processing of data. One of the main tasks of data analysis in machine learning is the task of prediction and classification [2]. There are many algorithms for solving these problems, such as linear regression, support vector method, nearest neighbors' method, logistic regression, solving trees, neural networks, etc. Unfortunately, sometimes the data has complex or combined properties, in such cases, not one algorithm or model is used, but a set of models combined into a composition (ensemble of models). Then the result is determined in the form of aggregation of the results obtained, for example, in the form of a linear combination [3]. The first papers devoted to the study of this problem of weak and strong learning algorithms were carried out in the 80's. Weak learning algorithm means that in a polynomial time it is possible to build an algorithm of recognition, the accuracy of which will be at least slightly more than 50%. By strong learning ability it is meant that it is possible to construct an algorithm in polynomial time, which could give arbitrarily accurate results. Studies have shown that strong training is equivalent to a weak one, since a weak model can be strengthened by constructing the correct composition [2]. In 1996, based on these studies, the Ada Boost algorithm was developed. This algorithm quickly gained popularity due to its simplicity and efficiency. The continuation and generalization of the Ada Boost algorithm is a gradient boosting which is one of the most popular ensemble methods. Successful application of ensemble methods is conditioned by the variety of basic models, on the basis of which the final model is built. Boosting began to be actively used in the tasks of ranking the issuance of search engines. This task was discharged from the point of view of the function, which is penalized for errors in the order of issuance, so it became convenient to simply insert it into gradient boosting realization [2].

One of the first to implement the AltaVista ranking, and soon followed by Yahoo, Yandex, Bing and others. Gradient boosting is not a specific algorithm, but a general methodology, how to build model ensembles. Moreover, the methodology is quite flexible and extensible: it is possible to train a large number of models, considering the various loss functions, and at the same time to attach various weight functions to them [1]. The article presents the main stages of constructing a model using parallel gradient boosting and decision trees for solving the classification problem.

Objectives of the article. The aim of the article is to describe the main steps of the gradient-boosting method for e-learning tasks using the parallel construction of the tree in the interest of creating information technology for solving classification problems.

Research and Results

We introduce some definitions and concepts.

Gradient boosting is a method of machine learning for regression and classification problems that creates a forecasting model in the form of an ensemble of weak prediction models, usually decision trees. He builds the model step-by-step, like other enhancement methods, and generalizes them, allowing to optimize an arbitrary differentiable loss function. In Fig. 1 shows an example of the Python gradient program boosting.

```
for i in range(30):
   tree = DecisionTree(xi,yi)
    tree.find_better_split(0)
    r = np.where(xi == tree.split)[0][0]
   left_idx = np.where(xi <= tree.split)[0]
   right_idx = np.where(xi > tree.split)[0]
    predi = np.zeros(n)
    np.put(predi, left_idx, np.repeat(np.mean(yi[left_idx]), r))
    np.put(predi, right_idx,
np.repeat(np.mean(yi[right_idx]), n-r))
   predi = predi[:,None]
   predf = predf + predi
   ei = y - predf
   yi = ei
    # plotting after prediction
    xa = np.array(x.x) # column name of x is x
    order = np.argsort(xa)
    xs = np.array(xa)[order]
    ys = np.array(predf)[order]
   #epreds = np.array(epred[:,None])[order]
   f, (ax1, ax2) = plt.subplots(1, 2, sharey=True, figsize
= (13,2.5))
    ax1.plot(x,y, 'o')
    ax1.plot(xs, ys, 'r')
    ax1.set_title(f'Prediction (Iteration {i+1})')
    ax1.set_xlabel('x')
    ax1.set_ylabel('y / y_pred')
    ax2.plot(x, ei, 'go')
    ax2.set_title(f'Residuals vs. x (Iteration {i+1})')
    ax2.set_xlabel('x')
    ax2.set_ylabel('Residuals')
```

Fig. 1. Python gradient program boosting

Busting is a meta-algorithm of machine learning of the ensemble, which primarily reduces bias, as well as variance in controlled learning and a family of machine learning algorithms that turn weak students into strong ones [6].

Decision trees are a way of presenting rules in a hierarchical, consistent structure [1]. In Fig. 2 showed an example of the temporary parallel model of boosting decision trees.

Loss functions - In mathematical optimization, statistics, econometrics, decision theory, machine learning and computational neuroscience, a loss function or cost function is a function that maps an event or values of one or more variables onto a real number intuitively representing some "cost" associated with the event [6].

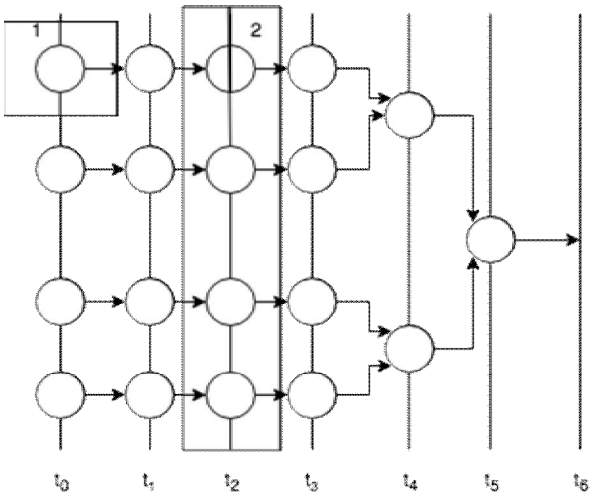

Fig. 2. Temporary parallel model of boosting decision trees

Ginny index - A statistical indicator by which one can describe the nature of the change of one magnitude relative to the change in the other [1].

The main stages of the construction of the method are shown in Fig. 3.

Input data:

– Decision trees built using a parallel method.;

– Architecture Classes (VLIW - GPU, AMD/ATI Radeonv (HD5770), RISC - ARM ThunderX, CISC - IBM System z10);

– decision time T_s task;

– architecture characteristics: the number of NM processors.

– Parallelization (CUDA, Open MP API-interface, MPI)[2]

Output data:

A subset of the original data distributed in accordance with the classes. Estimates of performance indicators: acceleration, accuracy, time building a model, the complexity of the program. Consider the appointment and formalized description of the main stages of modeling [1].

Stage 1 (symbol 2, Fig. 3) choice of the architecture of the computer system, depending on the requirements of the customer.

Step 2 (symbols 3,4,5,6, Fig. 3) - provides a kind of decomposition depending on the chosen architecture.

Step 3 (symbol 7, Fig. 3) - provides a choice of technology for parallel construction of the algorithm of gradient boosting.

Step 4 (symbol 9, Fig. 3) - building a decision tree on each node (Fig. 3.4)

Stage 4 (symbol 8, Fig. 3) - the construction of an ensemble of trees in one process.

Step 5 (symbol 10, Fig. 3) forwarding all an-tree-trees to a single process. The parallelism of the algorithm for constructing gradient boosting becomes possible only with the method of constructing the calculation process, based on the use of the associativity of the addition operation to calculate the loss factor.

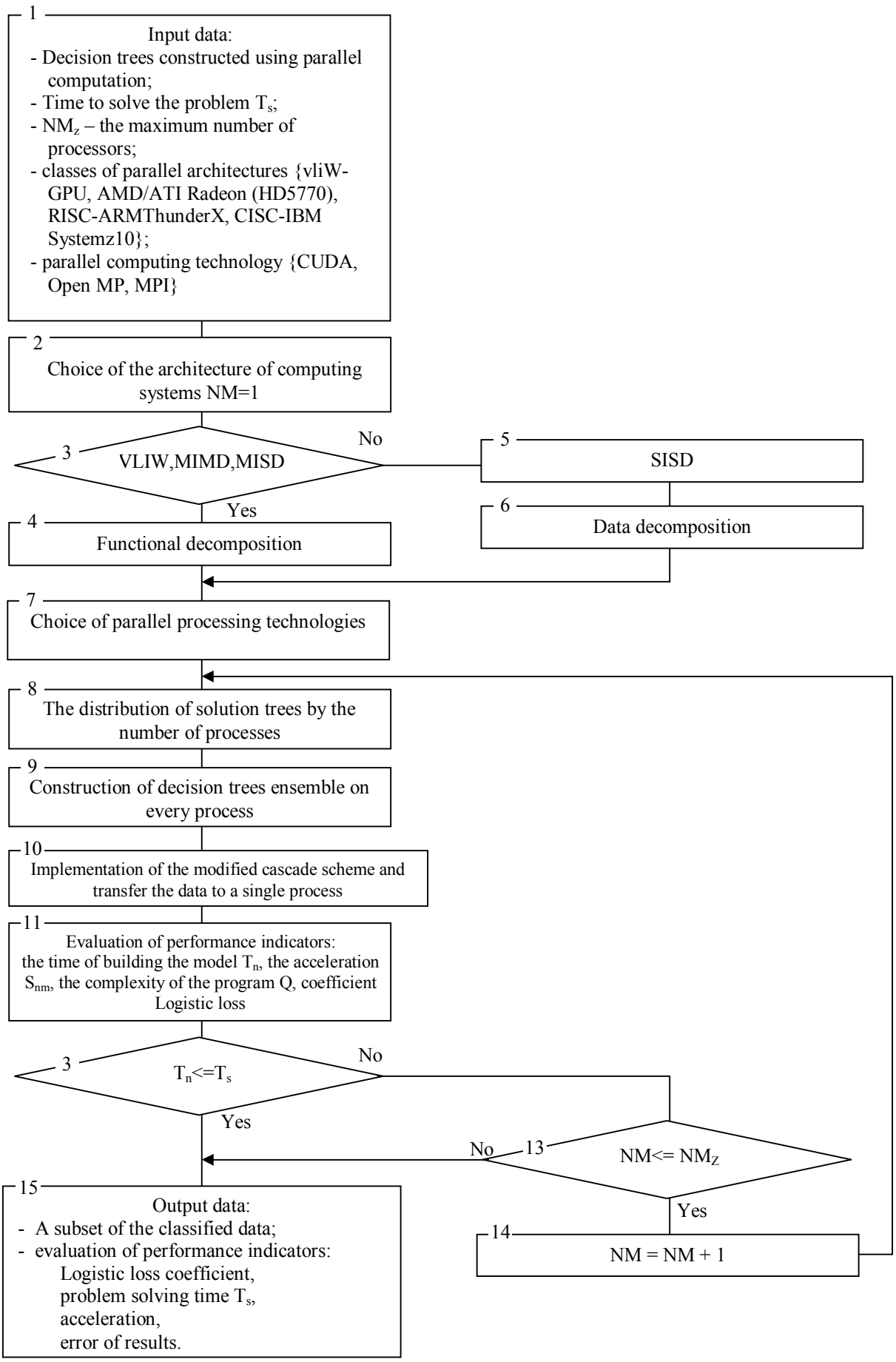

Fig. 3. The main stages of the method of parallel gradient boosting

This stage is divided into several iterations. sum able values are divided into $(n/\ell og_2 n)$ groups, each of which contains $\ell og_2 n$ elements;

Next, for each group, the sum of the values is calculated using a sequential summation algorithm; calculations in each group can be carried out independently of each other (ie, parallel – for this, there must be at least $(n/\ell og_2 n)$ processors); further for $(n/\ell og_2 n)$ the sum of individual groups is applied cascade scheme. In this scheme, there are two stages in the first stage of the cascade scheme, all the initial data is divided into pairs and for each pair the sum of the values is calculated, then all the resulting sums of pairs are also divided into pairs and again the summation of the values of the pairs, [5]. This approach is the implementation of a modified cascade scheme (Fig. 4).

The results of using parallelism in the construction of gradient boosting can be commented as follows.

The best acceleration gain is achieved by using 8 processes. Using the parallel gradient boosting algorithm allows you to speed up the retrieval of results by 14 times, but the process of writing a program becomes more complicated, and adding parallelism increases the probability of the need to change the classifier (improving the detail) otherwise the function of the classifier logistic loss.

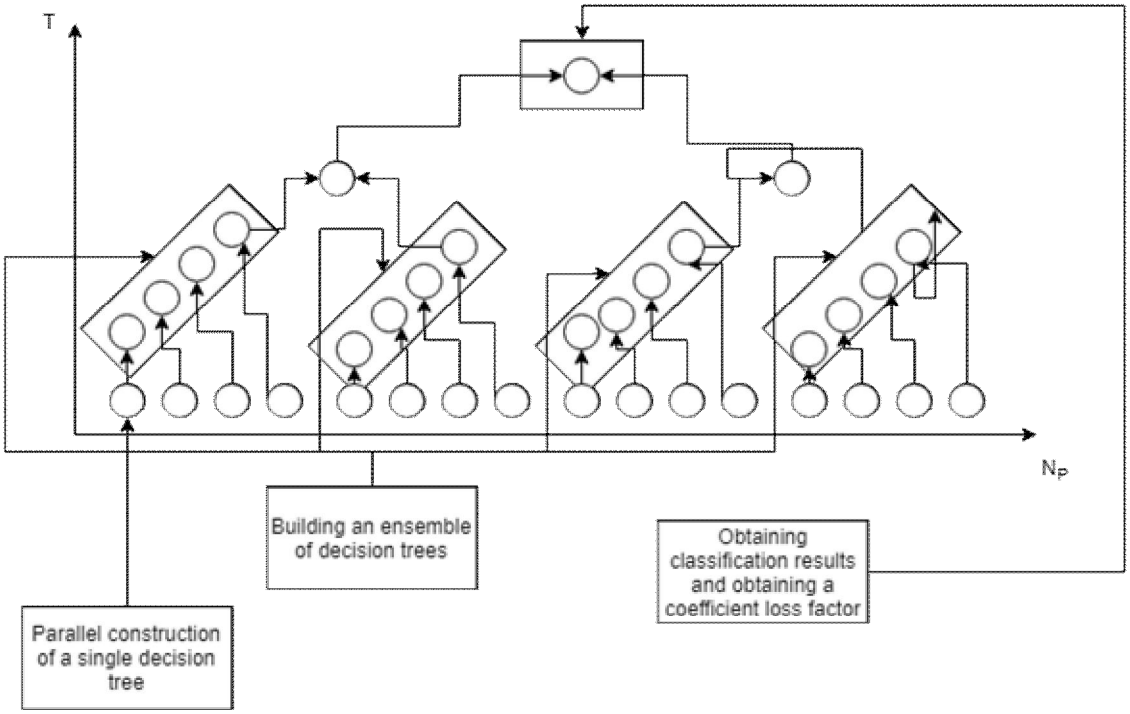

Fig. 4. The scheme of parallel gradient boosting

Level of confidence in parallel execution 72.24% when using one stream confidence level 68.1%.

Conclusions

1. The main stages of constructing gradient boosting using a parallel construction of the decision tree are considered for e-learning tasks in the interests of creating information technology for solving classification problems.

2. Parallelism of the gradient algorithm

It becomes possible only when construction of the process of calculations, based on the use of associatively of the operation of addition.

REFERENCES

1. Tolstoluzka, O. and Parshencev, B. (2018), "The solution of the classification problem in e-learning based on the method parallel construction of decision trees", *Advanced Information Systems*, Vol. 2, No. 3, pp. 5–9.
2. Hastie, T., Tibshirani, R. and Friedman, J. (2001), "Linear Methods for Classification", *The Elements of Statistical Learning*, ser. Springer series in statistics, Springer, New York, pp. 101–137.
3. Vapnik, V. (1998), *The Nature of Statistical Learning Theory*, John Wiley and Sons, N.Y., 300 p.
4. Gergel, V.P. (2007), *Theory and Practice of Parallel Computing*, Bean, Moscow, 71 p.
5. Voevodin, V.V. and Voevodin, Vl.V. (2002), *Parallel Computations*, BHV-Petersburg, St. Petersburg, 608 p.
6. Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.T. (1984), Classification and Regression Trees, Wadsworth, Belmont, California, 332 p.
7. Gehrke, J., Ganti, V., Ramakrishnan R. and Wei-Yin Loh (1999), "BOAT — optimistic decision tree construction", *ACM SIGMOD International Conference on Management of Data*, pp. 169–180.

8. Polyakov, G.A., Shmatkov, S.I., Tolstoluzhskaya, E.G. and Tolstoluzhsky, D.A. (2012), *Synthesis and analysis of parallel processes in adaptive time-parametric computing systems*, V. N. Karazin Kharkiv National University, Kharkiv, pp. 434-575.

Received (Надійшла) 29.05.2018
Accepted for publication (Прийнята до друку) 25.07.2018

Паралельна реалізація методу градієнтного бустінгу

О. Г. Толстолузька, Б. В. Паршенцев, О. Ю. Мороз

Останнім часом питанню машинного навчання приділяється все більше уваги у всіх галузях інформаційних технологій. З одного боку це пов'язано зі стрімким ростом вимог до майбутніх фахівців, а з іншого - з дуже швидким розвитком інформаційних технологій та Інтернет комунікацій. Однією з головних задач e-learning є задача класифікації. Для даного типу задач дуже добре підходить метод машинного навчання під назвою градієнтий бустінг. Градієнтний бустинг це сімейство потужних алгоритмів машинного навчання, які продемонстрували значний успіх у вирішенні практичних завдань. Дані алгоритми є дуже гнучкими і легко налаштованим для конкретних потреб програми, наприклад, вивчаються по відношенню до різних функцій втрат. Ідея бустінга полягає в ітеративному процесі послідовного побудови приватних моделей. Кожна нова модель навчається ґрунтуючись на інформації про помилки, зроблених на попередньому етапі, а результуюча функція являє собою лінійну комбінацію всього ансамблю моделей з урахуванням мінімізації будь штрафний функції. Математичний апарат градієнтного бустінгу гарно пристосовується для рішення задачі класифікації. Однак, з ростом кількості вхідних даних стає актуальним питання зменшення часу побудови ансамблю дерев рішень. Використання паралельних обчислювальних систем та паралельних технологій програмування дозволяє отримати позитивні результати, але вимагає розробки нових методів побудови градієнтного бустінгу. **У статті** розкриваються основні етапи методу паралельної побудови градієнтного бустінгу для вирішення задачі класифікації в e-learning. На відміну від існуючих, метод дозволяє враховувати особливості архітектури і організації паралельних процесів в обчислювальних системах із загальною і розподіленою пам'яттю. В методі врахована можливість оцінки показників ефективності побудови ансамблю дерев рішень та паралельних алгоритмів. Отримання показників ефективності на кожній ітерації методу допомагає обрати раціональну кількість паралельних процесорів в обчислювальній системі. Це дозволяє домогтися подальшого скорочення часу завершення градієнтного бустінга. Проведене моделювання з використанням технології паралельного програмування MPI, мови програмування Python для архітектури обчислювальної системи DM-MIMD підтверджує достовірність отриманих результатів. Наводиться приклад організації вхідних даних. Представлено Python програму для побудови градієнтного бустінга. Розроблена візуалізація отриманих оцінок показників ефективності дозволяє користувачу обрати необхідну конфігурацію обчислювальної системи.

Ключові слова: градієнтне підсилення; паралельний алгоритм; дерево рішень; оцінка ефективності паралельного алгоритму; електронне навчання.

Параллельная реализация метода градиентного бустинга.

Е. Г. Толстолужская, Б. В. Паршенцев, О. Ю. Мороз

В последнее время вопросу машинного обучения уделяется все больше внимания во всех областях информационных технологий. С одной стороны это связано со стремительным ростом требований к будущим специалистов, а с другой - с очень быстрым развитием информационных технологий и Интернет коммуникаций. Одной из главных задач e-learning является задача классификации. Для данного типа задач очень хорошо подходит метод машинного обучения под названием градиент бустинг. Градиент бустинг это семейство мощных алгоритмов машинного обучения, которые продемонстрировали значительный успех в решении практических задач. Данные алгоритмы являются очень гибкими и легко настраиваемым для конкретных нужд программы, например, изучаются по отношению к различным функциям потерь. Идея бустинга заключается в итеративном процессе последовательного построения частных моделей. Каждая новая модель учится основываясь на информации об ошибках, сделанных на предыдущем этапе, а результирующая функция представляет собой линейную комбинацию всего ансамбля моделей с учетом минимизации любой штрафной функции. Математический аппарат градиентного бустинга хорошо приспосабливаемый для решения задачи классификации. Однако, с ростом количества входных данных становится актуальным вопрос уменьшения времени построения ансамбля деревьев решений. Использование параллельных вычислительных систем и параллельных технологий программирования позволяет получить положительные результаты, но требует разработки новых методов построения градиентного бустинга. **В статье** раскрываются основные этапы метода параллельной построения градиентного бустинга для решения задачи классификации в e-learning. В отличие от существующих, метод позволяет учитывать особенности архитектуры и организации параллельных процессов в вычислительных системах с общей и распределенной памятью. В методе учтена возможность оценки показателей эффективности построения ансамбля деревьев решений и параллельных алгоритмов. Получение показателей эффективности на каждой итерации метода помогает выбрать рациональную количество параллельных процессоров в вычислительной системе. Это позволяет добиться дальнейшего сокращения времени завершения градиентного бустинга. Проведено моделирование с использованием технологии параллельного программирования MPI, языки программирования Python для архитектуры вычислительной системы DM-MIMD подтверждает достоверность полученных результатов. Приводится пример организации входных данных. Представлены Python программу для построения градиентного бустинга. Разработанная визуализация полученных оценок показателей эффективности позволяет пользователю выбрать необходимую конфигурацию вычислительной системы.

Ключевые слова: повышение градиента; параллельный алгоритм; дерево решений; оценка эффективности параллельного алгоритма; электронное обучение.