UDC 004. 8:004.93'1 **doi:** https://doi.org/10.20998/2522-9052.2025.1.03

Viktor Chelak, Oleksii Hornostal, Yehor Chelak, Svitlana Gavrylenko

National Technical University “Kharkiv Polytechnic Institute”, Kharkiv, Ukraine

ADVANCED METHODS FOR CLASSIFICATION QUALITY ASSESSMENT LEVERAGING ROC ANALYSIS AND MULTIDIMENSIONAL CONFUSION MATRIX

Annotation. The object of the study is the process of classifying objects in scientific problems. **The subject of the study is** methods aimed at assessing the effectiveness of multiclass classification. **The goal of the study** is to study the classification process and develop a classifier evaluation module to increase the speed of such evaluation and reduce the time to build complex machine learning classifiers. **Methods used:** methods for evaluating machine learning classifiers, methods for constructing ROC curves, principles of parallel and distributed computing. **Results obtained:** an analytical review of the scope of application of the classification quality assessment module in the field of humanities, technical and economic sciences was conducted. Existing classification quality assessment metrics were considered and mathematical descriptions of metrics were formed for the multi-class case. Software was developed that implements the proposed mathematical descriptions using parallel calculations and optimization of identical operations. The developed module was tested for reliability. **Conclusions.** According to the results of the study, methods for effective classification quality assessment is proposed, which allows reducing the time for assessing the quality of multi-class classifiers by 40% compared to the classical methods. The development of this module opens up broad prospects for further research in the direction of improving the quality of classification, which will contribute to the development of various spheres of human activity and increase the efficiency of solving tasks related to data analysis.

Keywords: machine learning; decision rule; ensemble classifiers; confusion matrix; quality assessment; ROC-analysis; multi-class classification; parallel programming.

Introduction

Classification is one of the key steps in data analysis and machine learning. The growing amount of data and the variety of areas of their application require constant improvement of methods of classification and evaluation of their effectiveness [1]. In today's world, where decision-making is increasingly based on the analysis of large volumes of data, it is important to have the means to accurately assess the quality of classification. This affects strategic decision-making and successful problem solving.

This problem is especially relevant when working with a large number of models, for example, with ensembles that require the simultaneous calculation of the performance indicators of a large number of different classifiers (both homogeneous and heterogeneous) [2].

There are various metrics for evaluating the performance of classification models. One of the most common is the ROC curve, which takes into account the ability of the model to distinguish classes and shows the relationship between sensitivity and specificity of classification [3, 4].

However, the ROC curve does not always give a complete picture of the classification results, especially in the case of multi-class classification [5]. Here there is a need to use a multidimensional confusion matrix, which will avoid distortions when assessing the quality of classification.

The development of a module that combines ROC curve and multidimensional confusion matrix is of great importance for various industries. From medicine to finance, from advertising to technology, in all these areas, classification accuracy is important.

In addition, the availability of such a module can become an important tool for researchers and practitioners in the field of data analysis [6], as it will allow processing larger volumes of information and developing more efficient classification methods by, for example, improving ensembles or using epochal (multi-step) approaches.

The relevance of the development of such a mathematical library for classification assessment is confirmed by a large number of scientists who solve various problems, have a stage of quality assessment of the obtained models, however, due to the use of conventional calculations and standard libraries, the quality assessment time increased, which led to the impossibility of integrating computational methods at the stage of model training.

That is why, in order to increase the speed of the evaluation process both when building the model and when fully analyzing the results, a new method of calculating the classification quality metrics is proposed, which, using the principles of parallelism and resource allocation, significantly reduces the time required for processing the results and, accordingly, speeds up the entire process of building a classification model in in general.

Object, subject and methods of study

The main idea of the work is to study the classification process and develop a classifier evaluation module to increase the speed of such evaluation and reduce the time to build complex machine learning classifiers.

The object of the study is the process of classifying objects in scientific problems.

The subject of the study is methods aimed at assessing the effectiveness of multiclass classification.

The main objectives of the study are as follows:

1. To conduct an analytical review of the scope of application of the classification quality assessment module in various industries, to form a mathematical general description of the classification problem.

2. To consider existing classification quality assessment metrics, to form mathematical descriptions of metrics for the multi-class case, taking into account

the capabilities of parallel and distributed computing in computer systems.

3. To develop software that will implement the proposed quality assessment metrics for classification with several classes. To develop a random data generation module with the ability to adjust the number of measurements, the number of resulting classes.

4. To test the developed module for reliability, compliance with functional requirements and correct operation.

Statement of the study problem

For a multidimensional confusion matrix $CM_{k\times k}$, where k – the number of classes, the computational complexity of metrics can increase significantly, which limits the application of standard approaches in conditions of large amounts of data.

The problem can be presented as a problem of minimizing the time for assessing the quality of classification T_{eval}, which depends on the computational costs of constructing a multidimensional error matrix and ROC curve:

$$T_{eval} = T_{CM} + T_{ROC}, \quad (1)$$

where T_{CM} – time to build a multidimensional error matrix, T_{ROC} – time to calculate ROC curves.

The second condition of the problem is to preserve the accuracy of the evaluation metrics, such as the area under the curve (AUC).

To solve the problem, it is proposed to use parallel and distributed computing techniques using GPU. Optimization of calculations involves parallel processing of a set of elements of the CM_{ij} matrix, as well as independent calculation of metrics for each class or group of classes:

$$CM_{ij} = \sum_{n=1}^{N} I(t_n = i, y_n = j), \quad (2)$$

where N is number of samples in the sample, t_n is true class of the n-th sample, y_n is the label predicted by the model, $I(\cdot)$ is the indicator function.

Related works analysis

Object classification is one of the key tasks of modern machine learning technology. In cases where there are many objects and features that describe them, the relevance of creating machine learning models that will solve this problem increases. Such models are classifiers.

The formation of classifiers can be mathematically described as the process of finding a decision rule that will allow describing or approaching the unknown target dependence f: $X \rightarrow Y$. In this dependence X is a set of objects, each of which is described by a deterministic sample of features x_i. In this case, Y is a vector with numbers or names of classes to which objects X belong [7].

Analysis of scientific works [8–16], which were devoted to solving the problem of classification in various fields, allows us to distinguish five types of classifications:

1. Binary classification is the simplest case, when there are only two resulting classes. Such classification can also be divided into two cases. In the first case, it is necessary to detect an object of only one class, and the second class exists to inform about the absence of belonging of this object to the first class. In the second, there are two classes and the conditions of the problem guarantee that the absence of objects in the image, the presence of some undefined class is excluded and cannot exist in the context of the problem.

2. Multiclass classification is a problem with the number of resulting classes from three to excessively large values (for example, recognizing hieroglyphs of a language that has tens of thousands of different characters). In some cases, instead of such classification, several binary classifiers are used, but such models most often lead to an increase in the time for training and recognizing one sample.

3. Disjoint class classification is a task where an object is assigned one and only one class from a list of a large number of possible classes. Such tasks typically arise in areas where it is necessary to accurately determine whether an object belongs to a specific category, such as medical diagnostics, object recognition in images, or category selection for text or audio content.

4. Classification with intersecting classes, on the contrary, solves problems in which the result of the classification may not be a value, number or name of a class, but a vector of such values. A common example of such a classification is the classification of animals by species, families, classes, etc. In such cases, the animal can be immediately identified by species and family.

5. Fuzzy classification is a special case of classification with overlapping classes. Unlike overlapping class classification, classifiers of this type have as a result a vector of static size equal to the number of classes. Each element of this vector takes a value from 0 to 1, and its index reflects the class number. Such values are the degrees of belonging of the studied object to the class. The resulting classifiers of this model are widely used in decision support systems, in which the final decision is formed by an expert in a specific field.

The development of a module for assessing the quality of classification methods is a promising and quite relevant task, because almost all fields of knowledge have scientific directions, problems and tasks that can be solved as a classification task. Fields of knowledge in which the module for assessing the quality of classification methods can be used: information technology, engineering, production and construction, healthcare, security and defense [8], services, natural sciences, business, social sciences and education.

Limited to the field of information technologies, there are several applied groups of tasks and problems related to the classification of objects of any information systems [9]. Among the main and primary tasks, the following can be distinguished:

1) Object classification in information systems security. Classification tasks in this context include the detection and identification of potentially dangerous objects, such as malicious code or intrusions. The use of classification models can help in real-time recognition of

new types of threats and take quick and effective measures to prevent them [10].

2) Pattern recognition and natural language processing. In information technology, where data often has a complex structure, classification is used for pattern recognition and natural language processing. For example, pattern recognition systems can be used to classify graphical objects, and natural language processing models can classify text data into semantic categories [11].

3) Personalized recommendations and content filtering. In the field of Internet technologies, classification problems are used to create personalized recommendations and content filtering. Classification models can analyze user preferences and make predictions about what content may be interesting or meet individual user needs [12].

4) Email Classification and Spam Filtering. Email classification tasks address the problem of spam filtering. Classification algorithms help automatically recognize and filter out unwanted messages, providing users with a clean and secure inbox [13].

5) Classification of network traffic anomalies. In the field of networking technology, classification is used to detect anomalies in network traffic. This allows security systems to automatically respond to unusual patterns and identify possible threats or intrusions [14].

6) Incident Classification and Service Recovery. In the field of incident management and technical support, classification helps automate the registration and analysis of incidents. This contributes to faster service recovery and increases the efficiency of customer service [15].

7) Big Data analysis and classification for the diagnostic nature of computer systems. In the field of big data analysis, classification is used to predict the health of systems and equipment. This can include detecting anomalies in equipment operation and warning of possible failures using data analysis and classification models [16].

It was also found that the problem of forming, on the one hand, a fast and, on the other hand, an effective and visual method to assessing the quality of classification, becomes especially relevant in the process of time-consuming calculation of various metrics for ensemble classifiers, especially when trying to build heterogeneous models [17]. In such a case, for example, when using the epochal approach, there is a need to calculate similar metrics for a large number of models, as well as for the ensemble classifier that is built on them for each of their epochs. In such a case, there is a need to use a less time-consuming and more visual method to assessing the effectiveness of various classifiers at all stages of the construction process. In general, the use of classification models in the field of information technology is expanding, providing new opportunities for automation and optimization of processes [18].

Overview of approaches and methods

For the final quality assessment, all reviewed works [17–20] that use machine learning technology use two elements: the confusion matrix and the ROC curve.

Confusion matrix – a table containing the results of comparing the results of classifiers with the expected values, which are the target values when building machine learning models. Fig. 1 shows a concise view of the confusion matrix for the two-dimensional case with most of the quality metrics calculated based on the values of the confusion matrix. The total number of samples in the training/test sample is called *Total_Population*. According to the type of condition, the value of *Total_Population* is represented as Actual Condition (*AC*) and Predicted Condition (*PC*).

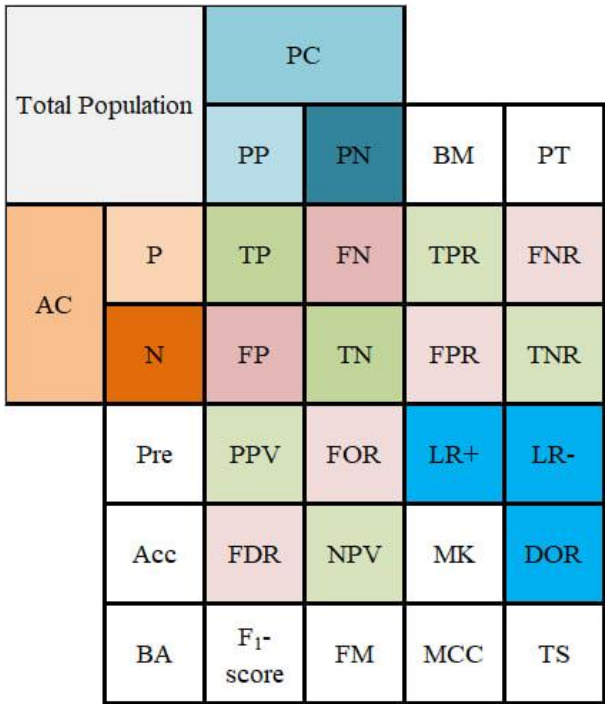


Fig. 1. Binary confusion matrix with classification quality metrics

An important check of the reliability of the matrix is the fulfillment of the condition $AC = PC$. If the condition is not met, we can conclude that the amount of data in the training/test sample does not match the number of resulting classifier values. In the case of binary classification, the actual value is divided into Positive (P) and Negative (N). For the classification results, the distribution is formed between Predicted Positive (PP) and Predicted Negative (PN). Four possible intersections between these sets form the value of the confusion matrix:

- The True Positive (TP) set consists of elements that are at the intersection of the sets P and PP ($TP = P \cap P$). Depending on the scope of use of classifiers, this set is also called: correct diagnosis, activation, triggering, or target class detection.
- The True Negative (TN) set is defined as $TN = N \cap PN$. This set includes all the training sample samples that did not actually belong to the target class, and the artificial intelligence model classified them as other classes.
- The False Positive (FP) set is defined as Type I errors or overestimation. Samples that the classifier has identified as elements of the target class, but which

actually belong to other classes, describe this set (FP = N ∩ PP).

- The False Negative (FN) set, also known as a type II error, a miss, a threat omission, or an underestimation. All samples that actually belong to the target class but were mistakenly recognized by the classifiers form this set. ($FN = P \cap PN$).

In the confusion matrix and quality metrics, the cardinalities of the above-considered sets (the number of elements belonging to them) are used. However, when forming the considered sets for multi-class classification, difficulties arise, because the opposite of the target class is a set of other classes. In other words, the size of the matrix increases, but *TP, TN, FP, FN* remain.

The class for which the sets *TP, TN, FP, FN* are formed is called the target class.

Fig. 2 shows the classification inconsistency matrices of six classes with different target classes. The set of these matrices is the multidimensional inconsistency matrix.

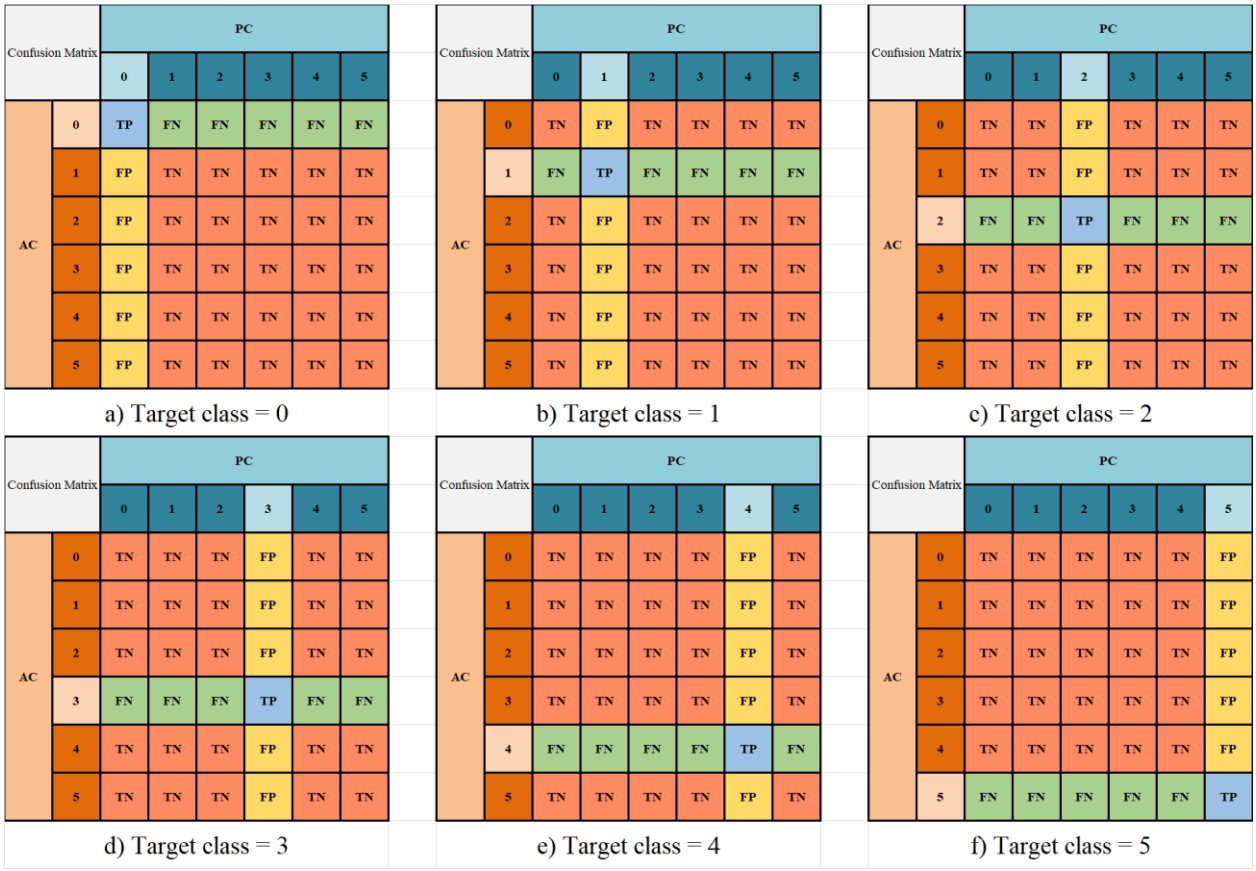


Fig. 2. Confusion matrices for classification problems with six target classes

By analyzing matrices for different numbers of resulting classes, it becomes possible to mathematically generalize calculations for sets *TP* (3), *TN* (4), *FP* (5) and *FN* (6).

$$TP_k = CM_{kk}, \tag{3}$$

$$TN_k = \sum_{i \neq k} \sum_{j \neq k} CM_{ij}, \tag{4}$$

$$FP_k = \sum_{i \neq k} \sum_{j = k} CM_{ij}, \tag{5}$$

$$FN_k = \sum_{i = k} \sum_{j \neq k} CM_{ij}, \tag{6}$$

where TP_k – number of elements of the set of true-positive cases of the target class k, TN_k – number of elements of the set of true-negative cases of the target class k, FP_k – number of elements of the set of errors of the first kind of the target class k, FN_k – number of elements of the set of errors of the second kind of the target class k, CM_{ij} – element of the confusion matrix located at the intersection of the row of the true class i and the column of the specified class j.

This generalization has the disadvantage of going through the matrix values four times. To significantly speed up the process of forming sets, we can use the fact that errors of the first and second kind are calculated along a certain vertical and horizontal axis of the matrix, and with a known *Total Population*, we can find the set *TN*. Improved calculation formulas can be written as expressions (7-10).

$$TP_k{}' = TP_k = CM_{kk}, \tag{7}$$

$$FP_k{}' = \sum_{i \neq k} CM_{ik}, \tag{8}$$

$$FN_k{}' = \sum_{i \neq k} CM_{ki}, \tag{9}$$

$$TN_k{}' = TotalPopulation - TP_k{}' - FP_k{}' - FN_k{}'. \tag{10}$$

The above sets for binary classification can also be computed in cases of multiclass classification with target classes. The actual state *AC* and the predicted state *PC*

do not change, but for each target class they are formed from different terms (11-14)

$$P_k = TP_k' + FN_k', \quad (11)$$

$$N_k = FP_k' + TN_k', \quad (12)$$

$$PP_k = TP_k' + FP_k', \quad (13)$$

$$PN_k = FN_k' + TN_k'. \quad (14)$$

Pre-calculating the values (11-14) for each target class will significantly simplify the calculations of other metrics, reducing them to a single operation of element-wise division of two vectors.

From the obtained values, it is possible to calculate the *Prevalence* (*Pre*) metric, which will allow to assess the balance of the training sample. If all Pre_k of all target classes are equal, the conclusion is made about the balance of the sample, in which case some metrics will coincide, which will allow to optimize the calculations. However, it should be noted that in the context of problems with many classes, the situation when the sample has the same number of samples of each class is quite rare, and occurs only in the case when the data was pre-processed on the unbalanced sample, using the *oversampling* and *undersampling* techniques.

The prevalence for the target class k can be calculated by the expression (15)

$$Pre_k = \frac{P_k}{P_k + N_k} = \frac{P_k}{TotalPopulation}. \quad (15)$$

The primary indicators of classification quality are the levels that determine the relation of each set of the confusion matrix table to the true state *AC*. Their efficient calculation is described in (16-19).

$$TPR_k = \frac{TP_k'}{P_k}, \quad (16)$$

$$FNR_k = 1 - TPR_k, \quad (17)$$

$$TNR_k = \frac{TN_k'}{N_k}, \quad (18)$$

$$FPR_k = 1 - TNR_k, \quad (19)$$

where k – target class, TPR_k – True Positive Rate (Recall), FNR_k – False Negative Rate (Miss rate), TNR_k – True Negative Rate (Specificity), FPR_k –False Positive Rate (Probability of false alarm).

Based on the relationships, a slight acceleration of calculations consists in replacing the division operation with the difference when calculating the levels of type I (*FPR)* and type II (*FNR)* errors by using the already calculated indicators of recall *TPR* and specificity *TNR*.

The predictive levels of classification are the ratios of matrix elements to the predicted state of *PC*. Analogously to the optimization used in calculating the *FPR* and *FNR* levels, we can find the predictive error levels. An efficient calculation of these ratios for multi-class classification cases can be formed by:

$$PPV_k = TP_k'/PP_k, \quad (20)$$

$$FDR_k = 1 - PPV_k, \quad (21)$$

$$NPV_k = TN_k'/PN_k, \quad (22)$$

$$FOR_k = 1 - NPV_k, \quad (23)$$

where PPV_k – Positive Predictive Value, FDR_k – False Discovery Rate, NPV_k – Negative Predictive Value, FOR_k – False Omission Rate.

The *PPV* and *NPV* coefficients allow us to estimate the space of the classifier's decision rule and analyze the volume of the plane in which the target class is located from the point of view of the constructed machine learning model.

All presented levels are defined on the interval [0;1]. For the TPR, TNR, PPV and NPV levels, the best value is obtained at the maximum (1). For the FPR, FNR, FDR and FOR levels, respectively, the best value is at the minimum (0).

However, these metrics can also be uncertain. The uncertainty of *TPR*, *TNR*, *FPR*, *FNR* indicates critical errors in the training data (the absence of samples of the target class or samples of other classes). The uncertainty of the predictive power of the *PPV*, *NPV*, *FDR* and *FOR* levels is interpreted as critical errors of the resulting predictive model (an example can be stub models that give the same, usually the most frequent, expected result for any data).

The simplest performance metric is the classical accuracy (*Acc*), which can be efficiently calculated by the expression:

$$Acc_k = (TP_k' + TN_k')/TotalPopulation. \quad (24)$$

For cases where the classes are unbalanced and for multi-class classification where the *TP* set is significantly smaller than the *TN* set, it is preferable to use Balanced Accuracy (*BA*). Balanced accuracy is defined as the arithmetic mean between the true-positive and true-negative rates:

$$BA_k = (TPR_k + TNR_k)/2. \quad (25)$$

One of the classical, quantitative indicators is *Informedness* (*BM*) and *Markedness* (*MK*), the mathematical description of which consists in doubling the measure of different probabilities. *Informedness* allows us to completely get rid of ROC analysis and find optimal values of the decision threshold, because geometrically *Informedness* determines the height to the optimal point of ROC analysis. *Markedness* performs a similar action, but with maximization of the levels of predictive significance of positive and negative results. This metric evaluates the machine learning model for excessive bias towards one of the classes, comparing it with a random distribution of samples. Unlike *Acc* and *BA*, these metrics take values in the range from -1 to 1:

$$BM_k = TPR_k + TNR_k - 1. \quad (26)$$

$$MK_k = PPV_k + NVP_k - 1. \quad (27)$$

Informedness can be calculated simultaneously with balanced accuracy: the sum of *TPR* and *TNR* is calculated, and two operations are performed on the result - division and decrement.

The Fowlkes–Mallows (*FM*) index is the geometric mean between the true-positive rate and the predictive value of a positive result (28). An alternative to the *FM* index that has gained greater popularity in medicine and engineering is the F_1-score. It is mathematically described as the harmonic mean between *TPR* and *PPV*.

Unlike the *FM* metric, it can become uncertain when *PPV=0* and *TPR=0.* This metric is calculated by the expression (29)

$$FM_k = \sqrt{TPR_k \times TNR_k}. \quad (28)$$

$$F_{1-score}(k) = \frac{2TPR_k \times PPV_k}{TPR_k + PPV_k}. \quad (29)$$

Modern quality assessment metrics include Prevalence Threshold (*PT*), Diagnostic Odds Ratio (*DOR*), Threat Score (*TS*), and Correlation Coefficient (*MCC*).

Prevalence Threshold (PT) unlike classical metrics, has a nonlinear plane and has a gap at *TPR=FPR*, which actually indicates the same percentage between detection and false positives. A model with such an uncertain metric cannot be used for diagnostics, threat detection or other classifications of critical importance. The *PT* metric is defined by (30). The diagnostic odds ratio or *DOR* is the ratio of the likelihood of a positive and negative result. The metric becomes uncertain in the absence of a type I or type II error.

$$PT_k = \frac{\sqrt{TPR_k \times FPR_k} - FPR_k}{TPR_k - FPR_k}. \quad (30)$$

The *DOR* is calculated by the likelihood ratio of a positive result LR_+ (31) and a negative result LR_- (32)

$$LR_+(k) = TPR_k / FPR_k, \quad (31)$$

$$LR_-(k) = FNR_k / TNR_k. \quad (32)$$

By calculating these ratios, we can find the *DOR* by the expression (33)

$$\log(DOR_k) = \log\left(LR_+(k)/LR_-(k)\right). \quad (33)$$

The Threat Score or *TS* (34) and the correlation coefficient or *MCC* (35) help to assess the quality of models by considering different aspects of the confusion matrix. *TS* focuses on determining the ratio between correctly classified positive examples and the sum of first and second type errors with the numerator. At the same time, *MCC* takes into account all elements of the confusion matrix, including *True Positives, True Negatives, False Positives,* and *False Negatives,* and provides a correlation between the predicted and observed values.

$$TS_k = \frac{TP_k{}'}{TP_k{}' + FP_k{}' + FN_k{}'}. \quad (34)$$

$$MCC_k = \frac{TP_k{}' \times TN_k{}' - FP_k{}' \times FN_k{}'}{\sqrt{PP_k \times P_k \times N_k \times PN_k}} \quad (35)$$

ROC curves and the area under them (*AUC-ROC*) are a powerful tool for visualizing the performance of models. They provide an understanding of how well a model discriminates between classes at different decision thresholds. The larger the area under the ROC curve, the better the model classifies the data. Comparing ROC curves for different models helps determine which model is more effective, as well as identify the optimal decision point that provides a balance between sensitivity and specificity.

By combining all the expressions and using the principles of parallel and distributed computing over matrices and vectors, it becomes possible to significantly accelerate the quality assessment process for multidimensional classification. Thanks to the schematic representation of the calculations, it is possible to distinguish 4 levels or tiers, along which calculations will be performed for all target classes simultaneously (provided that k does not exceed the number of cores of the distributed computing system):

- The first level metrics include: *Pre*, *P*, *N*, *PP*, *PN*, *TN*, *FN*, *FP*, *TP*. If we move on to calculating the next level metrics without completing the calculation of these, we will experience repeated operations on the same values.
- Second-level metrics include error rates, correct classification rates, and predictive classification rates: *TPR, TNR, FPR, FNR, PPV, NPV, FDR* and *FOR.*
- Third-level metrics include classic indicators that can be calculated in parallel using intermediate results: *Acc*, *BA*, *BM*, *MK*, F_1, *FM.*
- Fourth-level metrics: *PT,* LR_+, LR_-, *DOR, TS, MCC.*

The complete computational scheme in the Simulink mathematical package is presented in Fig. 3.

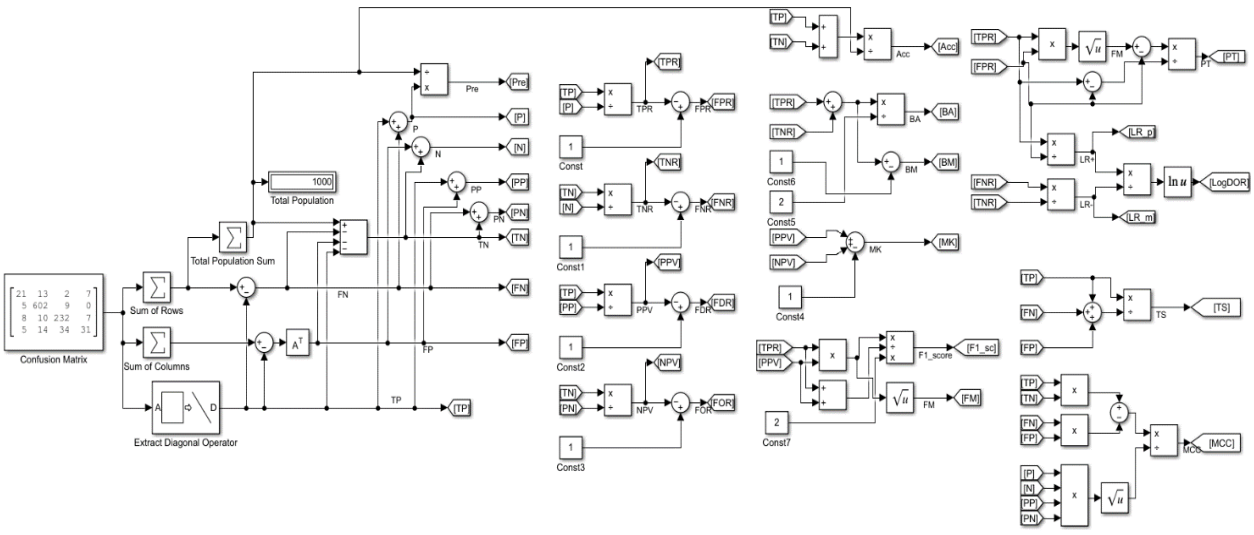


Fig. 3. Computational scheme of quality metrics for multi-class classification

In addition to increasing the speed of calculations, the problem of data interpretation is also considered. The developed module has an extremely high complexity of data interpretation, because with the addition of new classes to the classification tasks, the number of metrics increases. Thus, there is a need to form a general assessment that an expert could quickly assess and understand the main problems of the training sample and the built model. Analyzing the possible methods, four main approaches to forming a general assessment can be distinguished:

1. The worst-case metric value.
2. The arithmetic mean.
3. The weighted mean.
4. The result obtained by the meta-algorithm.

The worst-case metric value is determined by a system of equations (36)

$$F_1(QA\_M) = \begin{cases} \min\limits_k QA\_M_k, \text{Errors} \to \min \\ \max\limits_k QA\_M_k, \text{Errors} \to \max \end{cases}, \quad (36)$$

where F_1 – worst-case metric search function, QA_M – vector of metric values for different target classes k, Errors – number of errors at maximum value QA_M_i.

In other words, the worst indicator is determined by the minimum if, at a higher value of the metric, the number of errors tends to zero. Otherwise, the worst indicator is the maximum value at which the number of errors is maximum. An alternative solution may be to pre-process all metrics with errors as the difference between the maximum value that can be in a specific metric and its current value. This approach will allow us to abandon the system of equations and replace it with a regular min function. The disadvantage of this method is too poor results and destruction of the logic of the target classes, because after finding each worst indicator, the resulting score will consist of different metric values, different target classes. To take this disadvantage into account, it is also necessary to specify the address of the value in the form of a target class *k*.

The arithmetic mean can be specified by the function F_2 (37)

$$F_2(QA\_M) = \frac{\sum_{i=0}^{k-1} QA\_M_i}{k}. \quad (37)$$

The advantage of this approach is the processing of all values of each target class. However, there is a certain drawback associated with the imbalance of the training samples of multi-class classifications. In the case when the classes are unbalanced, the arithmetic mean value will not take into account the number of elements of the target class, which was the reason for the high results of individual metrics.

To avoid this drawback, it is proposed to use a prevalence metric that evaluates the sample, rather than a classifier (38)

$$F_3(QA\_M) = \sum_{i=0}^{k-1} Pre_i \times QA\_M_i, \sum_{i=0}^{k-1} Pre_i \equiv 1. \quad (38)$$

The sum of the elements of the prevalence vector Pre_k is always equal to one, so this method can be described by the usual sum of products.

The fourth method is more abstract and requires the selection and justification of a meta-model, which can be built using machine learning technologies. This approach is similar to the principle of building stacking ensembles, and can be described as (39)

$$F_4(QA\_M) = g(f(w, QA\_M), \hat{R}) \to R, \quad (39)$$

where R – expected value of the overall metric, $\hat{R}$ – predicted value of the overall metric; w – meta-algorithm configuration options; $f(x)$ – meta-model building function; $g(x, y)$ – decisive rule.

When setting up a meta-algorithm, such difficulties arise as: forming a training dataset (meta-dataset), choosing a training algorithm and its hyperparameters. The above problems can be solved as a separate study, which can be a promising development of this thesis. In addition, such a tool has disadvantages: high complexity, low speed compared to other approaches and the problem of the first method remains - the meta-model can also take into account only part of the target classes.

Thus, of the considered methods, the most attractive is the weighted average, which uses the *Prevalence* metric as weighting factors.

For multivariate ROC analysis, the following curve construction method is proposed:

Step 1. Search for the class that is geometrically closest to the origin (the class should not have a threshold value at the current moment).

Step 2. Finding the optimal threshold value to maximize *TPR* and minimize *FPR*.

Step 3. If all classes have been determined with a threshold value, go to step 4. Otherwise, go to step 1 and start searching for the threshold value from the previous threshold.

Step 4. The intervals describing the classes have been successfully constructed.

In this way, optimal threshold values are obtained and there is no need to adjust the intervals (the intervals do not intersect and there are no empty intervals between them, on which the decision rule will not give a result at all). The disadvantage of this method is the inability to combine the results into one curve, as can be done with metrics. However, a similar operation can be performed on the areas under such curves. Fig. 4 shows an example of ROC curves for target classes 0-3 for a four-class classification problem using the proposed algorithm. Fig. 5 shows the curves that were calculated separately.

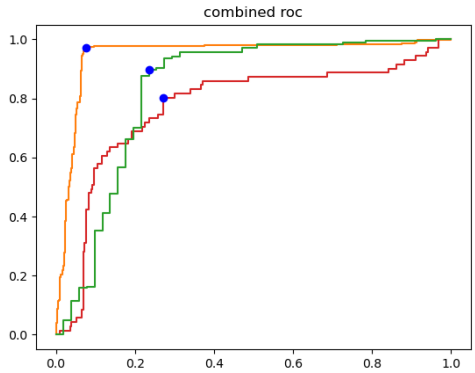


Fig. 4. ROC curves for different target classes using the proposed method

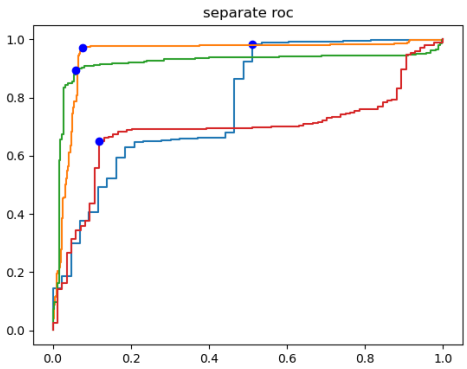


Fig. 5. Binary ROC curves for different target classes

Experimental part

For effective testing of the developed module, the resulting signals of classifiers and true class values are required. In the usual case, a training sample is found, a machine learning algorithm is selected, a classifier is built, and the values of the training sample are provided. However, in multi-class classification, the tasks may have different classes, and in many cases have a large number of zero cells of the multidimensional confusion matrix, which is a positive sign, but will not allow objectively assessing the effectiveness of the assessment with fully filled samples.

In order to obtain the most complex mismatch matrices to check the module for errors and unhandled exceptions, it is necessary to develop a generator program that will generate different values of the hypothetical model in the interval from zero to one. At the same time, it is necessary to generate these results with different values of the true class.

The algorithm for generating pseudo-random input data of the developed module is described in two stages:

1) Generation of quadratic equations describing the distribution of values separately for each class;

2) Generation of input data samples in a given number, taking into account the distribution functions obtained in the first stage.

The distribution of values is described as a quadratic equation, which is combined with some line of minimum probability of obtaining elements outside the parabola. This probability is denoted as p and is given as a parameter. The parameters t, u and s are given randomly (40)

$$\begin{gathered} P(x) = \max(ax^2 + bx + c, p), \\ a = -\frac{1}{s^2}, \\ b = -2ua, \\ c = au^2 + t, \\ u \in [0;1], s \in (0;1/3], t \in (0;1], p \in [0;1] \end{gathered} \quad (40)$$

The parameters *t* and *s* affect the prevalence of the class in the sample. The parameter u affects the center of the cluster. First and second type errors will occur in 2 situations: when the element is to the left or right of the parabola, and when the element is at the intersection of two distribution functions.

The second stage is described according to the principle of simulation. A pair of numbers from zero to one is generated. The first number is the value of the hypothetical classification model, the second number defines the real class. After that, the first number is checked for compliance with the requirements of the distribution function of the real class. If this requirement is not met, the generated element is not added to the resulting list. Having successfully generated *N* samples, we obtain *k* groups, which are described by pseudo-random distributions.

Figures 6–8 present different results of data generation for different numbers of classes and objects. The results confirm the feasibility of using this software tool to check the developed module for errors. Based on the generation, the results of which are presented in Fig. 7, the developed generator program can simulate even those tasks where the number of resulting classes is in the thousands.

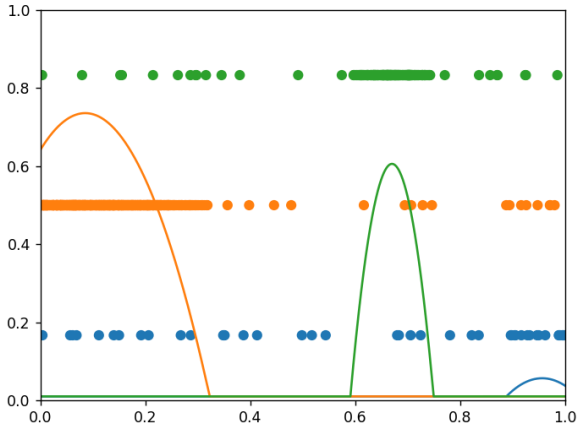


Fig. 6. The result of generation when *N=1000, k=3*

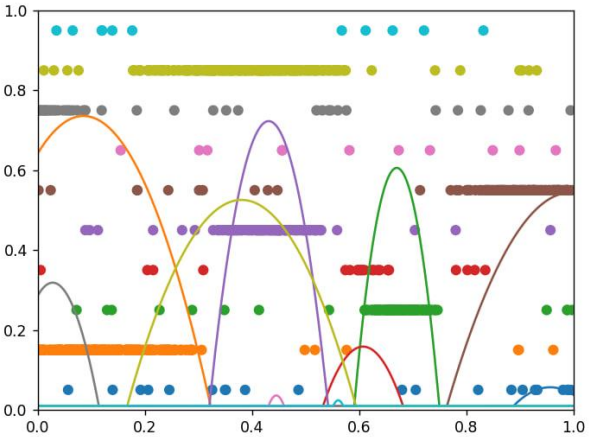


Fig. 7. The result of generation when *N=1000, k=10*

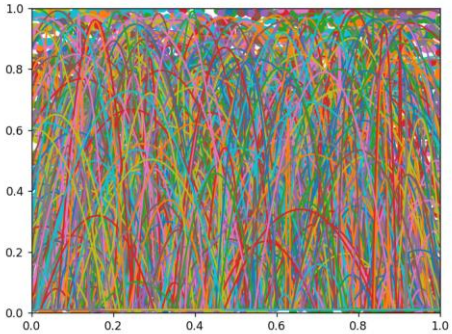


Fig. 8. The result of generation when *N=100000, k=10000*

To test the software implementation of the classification quality assessment module, input data with various configuration parameters were generated and deterministic tests were performed, which were aimed at intentionally triggering exceptional situations when a separate metric became uncertain.

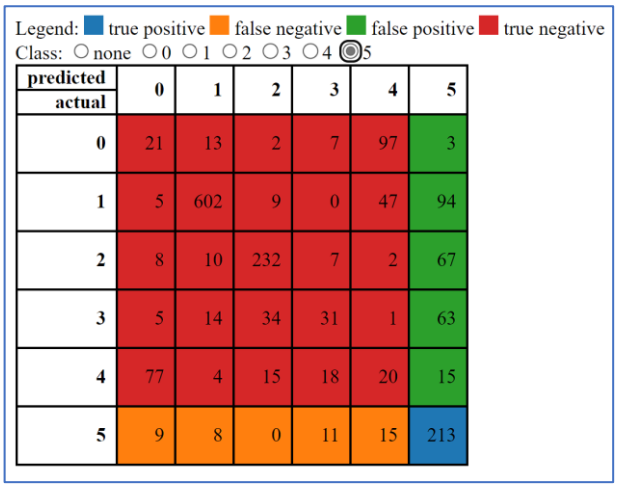
Fig. 9 shows the result of forming the confusion matrix for the case of classification of six classes.

As a result, together with the confusion matrix, we obtain a table of metrics for each target class.

An additional column calculates the weighted average value of each metric.

Since most metrics have a certain range of valid values, we can add conditional formatting.

Fig. 10 shows the resulting table with quality metrics.



predicted / actual	0	1	2	3	4	5
0	21	13	2	7	97	3
1	5	602	9	0	47	94
2	8	10	232	7	2	67
3	5	14	34	31	1	63
4	77	4	15	18	20	15
5	9	8	0	11	15	213

Fig. 9. One of the deterministic tests

metric	0	1	2	3	4	5	weighted
Pre	0.08038	0.4255	0.1832	0.08319	0.08375	0.1439	nan
P	143	757	326	148	149	256	455
N	1636	1022	1453	1631	1630	1523	1324
PP	125	651	292	74	182	455	427.4
PN	1654	1128	1487	1705	1597	1324	1352
TN	1532	973	1393	1588	1468	1281	1232
FN	122	155	94	117	129	43	119.7
FP	104	49	60	43	162	242	92.17
TP	21	602	232	31	20	213	335.3
metric	0	1	2	3	4	5	weighted
TPR	0.1469	0.7952	0.7117	0.2095	0.1342	0.832	0.629
FPR	0.8531	0.2048	0.2883	0.7905	0.8658	0.168	0.371
TNR	0.9364	0.9521	0.9587	0.9736	0.9006	0.8411	0.9335
FNR	0.06357	0.04795	0.04129	0.02636	0.09939	0.1589	0.06646
PPV	0.168	0.9247	0.7945	0.4189	0.1099	0.4681	0.664
FDR	0.832	0.07527	0.2055	0.5811	0.8901	0.5319	0.336
NPV	0.9262	0.8626	0.9368	0.9314	0.9192	0.9675	0.9069
FOR	0.07376	0.1374	0.06321	0.06862	0.08078	0.03248	0.09313

metric	0	1	2	3	4	5	weighted
Acc	0.873	0.8853	0.9134	0.9101	0.8364	0.8398	0.8809
BA	0.5416	0.8736	0.8352	0.5915	0.5174	0.8366	0.7813
BM	0.08328	0.7473	0.6704	0.1831	0.03484	0.6731	0.5625
MK	0.09424	0.7873	0.7313	0.3503	0.02911	0.4357	0.5709
F1sc	0.1567	0.8551	0.7508	0.2793	0.1208	0.5992	0.6336
FM	0.1571	0.8575	0.7519	0.2962	0.1215	0.6241	0.6399
metric	0	1	2	3	4	5	weighted
PT	0.7068	0.3366	0.389	0.6602	0.7175	0.31	0.431
LR+	0.1721	3.884	2.468	0.265	0.155	4.953	2.867
LR-	0.06789	0.05036	0.04307	0.02708	0.1104	0.1889	0.07346
lnDOR	0.9304	4.345	4.048	2.281	0.34	3.267	3.354
TS	0.08502	0.7469	0.601	0.1623	0.06431	0.4277	0.5152
MCC	0.08859	0.767	0.7002	0.2533	0.03185	0.5415	0.5635

Fig. 10. Result of calculating quality metrics

The effectiveness of the developed module is assessed according to the following rules:

1) The number of samples for all performance tests is the same (N=10000);

2) The number of classes for each test is a multiple of four ($K \bmod 4 = 0$);

3) The methods used are:

3.1) Classic linear calculation of quality metrics without optimization (C);

3.2) Improved calculation of quality metrics, with optimization, but linear (1 process, B);

3.3) Calculation with parallel execution (2 processors, A_2);

3.4) Calculation with parallel execution (4 processors, A_4);

3.5) Calculation with parallel execution (8 processors, A_8).

The dependence of the quality assessment time on the number of classes for the above configurations is shown in Fig. 11. Based on the results of performance testing, configuration B allows us to reduce the calculation time by ~15%. Using two processors in configuration A_2 reduces the calculation time by ~32% compared to configuration C. For the standard case, which exists on almost all modern processors (A_4), acceleration allows us to reduce the classification quality assessment time by ~40%. When using processors with 8 command pipelines (A_8), it allows us to reduce the calculation time by ~59%. Thus, the effectiveness of the developed module has been confirmed experimentally. The next stage of work may be the formation of a special API for training neural networks with a large number of layers and neurons.

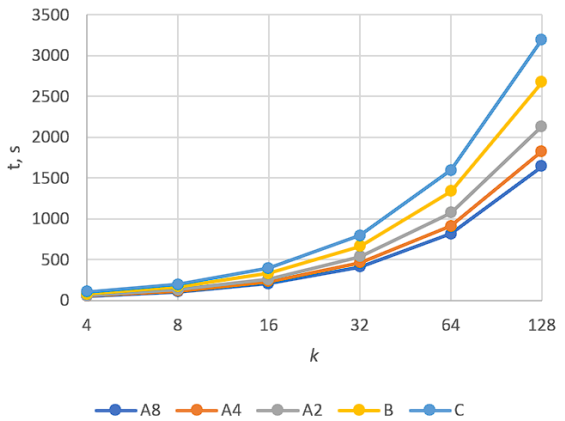


Fig. 11. Dependence of quality assessment time on the number of classes

Conclusions

The work is devoted to solving the current scientific and applied problem of assessing the quality of classification models built using machine learning technology and improving the evaluation process to increase the speed of processing classification results on training and test samples.

Existing metrics for assessing the quality of classification were considered and mathematical descriptions of metrics were formed for the multi-class case.

Existing mathematical descriptions were improved for their use in further parallel calculations. The proposed mathematical descriptions take into account the fact of simultaneous calculation of all metrics, which allows reducing the number of identical operations.

Software was developed that implements the proposed mathematical descriptions using parallel calculations and optimization of identical operations. The computational scheme of the module was developed on Simulink.

A multi-class ROC analysis module was developed, which allows assessing the quality of classification of different target classes.

The module includes the three formulated methods: parallel computing of confusion matrices for multi-class classifiers, general selection and assessment of confusion matrix metrics, multivariate ROC curve construction. They can be used when working with a large number of classification models, for example, when constructing decision trees, neural networks or ensembles, including heterogeneous ones. A side effect of study is the algorithm for generating pseudo-random input data of the developed module, which can be used in syntactic data with noise creation.

The developed module was tested for reliability, compliance with functional requirements, and a comparative analysis of the developed module was conducted on different settings (parallel and linear). The developed software allows reducing the time for assessing the quality of multi-class classifiers by 40%.

Scientific novelty of the results obtained: for the first time, a module for assessing the quality of classification has been developed, which increases the speed of assessing the accuracy of multi-class classifiers and reduces the time for building artificial intelligence models by using parallel computing techniques when calculating the indicators of the multidimensional confusion matrix.

The development of this module opens up broad prospects for further research in the direction of improving the quality of classification, which will contribute to the development of various spheres of human activity and increase the efficiency of solving tasks related to data analysis.

References

1. An, Q., Huang, S., Han, Y. and Zhu, Y. (2024), “Ensemble learning method for classification: Integrating data envelopment analysis with machine learning”, *Computers and Operations Research*, 169, doi: https://doi.org/10.1016/j.cor.2024.106739
2. Gavrylenko, S., Chelak, V. and Hornostal, O. (2021), “Ensemble Approach Based on Bagging and Boosting for Identification the Computer System State”, *2021 XXXI International Scientific Symposium Metrology and Metrology Assurance (MMA)*, Sozopol, Bulgaria, 2021, pp. 1–7, doi: https://doi.org/10.1109/MMA52675.2021.9610949
3. Li, J. (2024), “Area under the ROC Curve has the most consistent evaluation for binary classification”, *PLoS ONE*, vol. 19, is. 12, e0316019, doi: https://doi.org/10.1371/journal.pone.0316019
4. Phuong, L.B. and Zung, N.T. (2023), “Accuracy Measures and the Convexity of ROC Curves for Binary Classification Problems”, *Studies in Computational Intelligence*, 1045, pp. 155–163, doi: https://doi.org/10.1007/978-3-031-08580-2_15
5. Gavrylenko, S., Vladislav, Z. and Khatsko, N. (2023), “Methods For Improving The Quality Of Classification On Imbalanced Data”, *2023 IEEE 4th KhPI Week on Advanced Technology, KhPI Week 2023 – Conf. Proc.*, doi: https://doi.org/10.1109/KhPIWeek61412.2023.10312879
6. Petrovska, I., Kuchuk, H. and Mozhaiev, M. (2022), “Features of the distribution of computing resources in cloud systems”, *2022 IEEE 4th KhPI Week on Advanced Technology*, KhPI Week 2022 - Conference Proceedings, 03-07 October 2022, Code 183771, doi: https://doi.org/10.1109/KhPIWeek57572.2022.9916459
7. Gavrylenko, S., Hornostal, O. and Chelak, V. (2022), “Research of Methods of Identifying the Computer Systems State based on Bagging Classifiers”, *2022 IEEE 3rd KhPI Week on Advanced Technology* (KhPIWeek), Kharkiv, Ukraine, 2022, pp. 1–6, doi: https://doi.org/10.1109/KhPIWeek57572.2022.9916439
8. Brenych, Y. (2011), “Classification methods using Winners-Take-All neural networks”, *Perspective Technologies and Methods in MEMS Design*, Polyana, Ukraine, pp. 234–236, available at: https://ieeexplore.ieee.org/document/5960381
9. Gavrylenko, S., Chelak, V., Hornostal, O. and Gornostal, S. (2019), “Identification of the Computer System State Based on Multidimensional Discriminant Analysis”, *2019 XXIX International Scientific Symposium "Metrology and Metrology Assurance" (MMA)*, Sozopol, Bulgaria, 2019, pp. 1–4, doi: https://doi.org/10.1109/MMA.2019.8936011
10. Markatopoulou, F., Mezaris, V., Pittaras, N. and Patras, I. (2015), “Local Features and a Two-Layer Stacking Architecture for Semantic Concept Detection in Video”, IEEE Transactions on Emerging Topics in Computing, vol. 3, no. 2, pp. 193–204, June 2015, doi: https://doi.org/10.1109/TETC.2015.2418714
11. Hayes, J. H., Li, W. and Rahimi, M. (2014), “Weka meets TraceLab: Toward convenient classification: Machine learning for requirements engineering problems: A position paper”, *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering* (AIRE), Karlskrona, Sweden, pp. 9–12, doi: https://doi.org/10.1109/AIRE.2014.6894850
12. Olewnik, A. and Memarian, B. (2022), “Characterizing Student Engineering Problem Engagement Through Process Diagramming”, *2022 IEEE Frontiers in Education Conference* (FIE), Uppsala, Sweden, pp. 1–5, doi: https://doi.org/10.1109/FIE56618.2022.9962445
13. Mottier, M., Chardon, G. and Pascal, F. (2022), “RADAR Emitter Classification with Optimal Transport Distances”, *2022 30th European Signal Processing Conference* (EUSIPCO), Belgrade, Serbia, 2022, pp. 1871–1875, doi: https://doi.org/10.23919/EUSIPCO55093.2022.9909967

14. Zhang, R. (2022), “Classification and Management of Accounting Management Data Based on Big Data Technology”, *2022 IEEE 2nd International Conference on Mobile Networks and Wireless Communications* (ICMNWC), Tumkur, Karnataka, India, 2022, pp. 1–5, doi: https://doi.org/10.1109/ICMNWC56175.2022.10031282
15. Ohrimenco, S., Borta, G. and Tetiana, B. (2019), “Shadow of Digital Economics”, *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology* (PIC S&T), Kyiv, Ukraine, pp. 776–780, doi: https://doi.org/10.1109/PICST47496.2019.9061545
16. Sztojanov, V. V. and Popescu-Mina, C. (2007), “Image Processing for Classification in Biology Systems”, *2007 2nd International Workshop on Soft Computing Applications*, pp. 33–38, doi: https://doi.org/10.1109/SOFA.2007.4318301
17. Hornostal, O. and Gavrylenko S. (2023), “Application of heterogeneous ensembles in problems of computer system state identification”, *Advanced Information Systems*, vol. 7, no. 4, pp. 5–12, doi: https://doi.org/10.20998/2522-9052.2023.4.01
18. Mathew, R. M. and Gunasundari, R. (2021), “A Review on Handling Multiclass Imbalanced Data Classification In Education Domain”, *2021 International Conference on Advance Computing and Innovative Technologies in Engineering* (ICACITE), Greater Noida, India, 2021, pp. 752–755, doi: https://doi.org/10.1109/ICACITE51222.2021.9404626
19. Gavrylenko, S., Chelak, V. and Hornostal, O. (2020), “Research of Intelligent Data Analysis Methods for Identification of Computer System State”, 30th International Scientific Symposium Metrology and Metrology Assurance, MMA 2020, 9254252, doi: https://doi.org/10.1109/MMA49863.2020.9254252
20. Antoni, L., Cornejo, M. E., Medina, J. and Ramírez-Poussa, E. (2021), “Attribute Classification and Reduct Computation in Multi-Adjoint Concept Lattices”, *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 5, pp. 1121–1132, May 2021, doi: https://doi.org/10.1109/TFUZZ.2020.2969114

Received (Надійшла) 27.10.2024
Accepted for publication (Прийнята до друку) 15.01.2025

ABOUT THE AUTHORS / ВІДОМОСТІ ПРО АВТОРІВ

Челак Віктор Володимирович – PhD (комп’ютерна інженерія), доцент кафедри “Комп'ютерна інженерія та програмування”, Національний технічний університет “Харківський політехнічний інститут”, Харків, Україна;
Viktor Chelak – PhD in Computer Engineering, Associate Professor of Department of "Computer Engineering and Programming", National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: victor.chelak@gmail.com; ORCID Author ID: https://orcid.org/0000-0001-8810-3394;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57189040595.

Горносталь Олексій Андрійович – PhD (комп’ютерна інженерія), асистент кафедри "Комп'ютерна інженерія та програмування", Національний технічний університет “Харківський політехнічний інститут”, Харків, Україна;
Oleksii Hornostal – PhD in Computer Engineering, Lecturer of Department of "Computer Engineering and Programming", National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: gornostalaa@gmail.com; ORCID Author ID: https://orcid.org/0000-0001-5820-9999;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57216331944.

Челак Єгор Володимирович – магістр кафедри "Комп'ютерна інженерія та програмування", Національний технічний університет “Харківський політехнічний інститут”, Харків, Україна;
Yehor Chelak – Master of Science in Computer Engineering, Department of "Computer Engineering and Programming", National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: egor.chelak@gmail.com; ORCID Author ID: https://orcid.org/https://orcid.org/0000-0002-3898-4370.

Гавриленко Світлана Юріївна – доктор технічних наук, професорка, професорка кафедри "Комп'ютерна інженерія та програмування", Національний технічний університет “Харківський політехнічний інститут”, Харків, Україна;
Svitlana Gavrylenko – Doctor of Technical Science, Professor, Professor of Department of "Computer Engineering and Programming", National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine;
e-mail: gavrilenko08@gmail.com; ORCID Author ID: https://orcid.org/0000-0002-6919-0055;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57189042150.

Удосконалені методи оцінки якості класифікації з використанням ROC аналізу та багатовимірної матриці невідповідності

В. В. Челак, О. А. Горносталь, Є. В. Челак, С. Ю. Гавриленко

Анотація. Об'єктом дослідження є процес класифікації об’єктів в наукових задачах. **Предметом дослідження** є методи, спрямовані на оцінку ефективності багатокласової класифікації. **Метою є дослідження** процесу класифікації та розробка модуля оцінки класифікаторів для підвищення швидкості такої оцінки та дозволить зменшити час побудови складних класифікаторів машинного навчання. **Методи, що використовуються:** методи оцінки класифікаторів машинного навчання, методи побудови ROC-кривих, принципи паралельних та розподілених обчислень. **Отримані результати:** проведено аналітичний огляд області застосування модуля оцінки якості класифікації в галузях гуманітарних, технічних та економічних наук. Розглянуто існуючи метрики оцінки якості класифікації та сформовано математичні описи метрик для багатокласового випадку. Розроблено програмне забезпечення, яке реалізує запропоновані математичні описи з використанням паралельних обчислень та оптимізації однакових операцій. Виконано тестування розробленого модуля на надійність. **Висновки.** За результатами дослідження запропоновано методи ефективного оцінювання якості класифікації, що дозволяє зменшити час оцінки якості багатокласових класифікаторів на 40% порівнюючи зі класичними методами. Розвиток даного модуля відкриває широкі перспективи для подальших досліджень у напрямку покращення якості класифікації, що сприятиме розвитку різних сфер людської діяльності та підвищенню ефективності вирішення завдань, пов'язаних з аналізом даних.

Ключові слова: машинне навчання; вирішальне правило; ансамблеві класифікатори; матриця невідповідності; оцінка якості; ROC-аналіз; багатокласова класифікація; паралельне програмування.