Classification System Based on Ensemble Methods for Solving Machine Learning Tasks

Peter Bidyuk[1], Irina Kalinina[2], Oleksandr Zhebko[3], Aleksandr Gozhyj[2] and Tetiana Hannichenko[3]

1. *National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute, 37, Prospect Beresteiskyi (former Peremohy), Kyiv, Ukraine, 03056*
2. *Petro Mohyla Black Sea National University, St. 68 Desantnykiv 10, Mykolaiv, Ukraine, 54000*
3. *Mykolayiv National Agrarian University, St. Georgiy Gongadze 9, Mykolaiv, Ukraine, 54020*

Abstract

The paper investigates the solution of the classification problem using a two-level structure of model ensembles based on machine learning methods. The general structure of a two-level ensemble for solving classification problems is proposed. Based on the use of the two-level ensemble learning structure in the processing of two datasets, the quality of classification was improved. The procedures for processing the datasets included identifying and describing the key quality characteristics of the models, selecting a metric, selecting the base models, selecting parameters for the base models and ensemble methods. Preliminary data processing was performed. The basic datasets are divided into training and test samples, and input variables are generated. The results of applying simple classifiers and the ensemble of the two-level classification model are presented, and the efficiency of the developed classification models is evaluated. A two-level ensemble structure was used to find a compromise between the bias and variance inherent in machine learning models. At the first level of the ensemble, stacking was used to reduce the bias of the base models. This resulted in a preliminary improvement in classification quality. At the second level, bagging was used to reduce the variance of the base models. The basic classification models and ensemble models based on stacking and bagging, as well as metrics for assessing the quality of using basic classifiers and models of the first and second levels, were studied.

Keywords 1

Ensemble models, Classification task, Forecasting, Bagging, Stacking, Structure of the two-level ensemble, Quality metrics of classifiers.

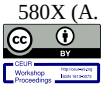
1. Introduction

The rapid development of machine learning technologies has initiated the development of new methods and algorithms that more effectively solve data mining and forecasting tasks. The main feature of machine learning methods is not a direct solution to a problem, but learning from a set of examples, which allows these methods to be adapted to solve specific problems of processing large amounts of data and discover new knowledge in them. A variety of methods are used to implement machine learning technologies, such as mathematical statistics, probabilistic methods, numerical methods, optimization methods, probability theory, graph theory, various data mining methods, etc [1-4].

MoMLeT+DS 2023: 5th International Workshop on Modern Machine Learning Technologies and Data Science, June 3, 2023, Lviv, Ukraine

EMAIL: pbidyuke_00@ukr.net (P. Bidyuk); irina.kalinina1612@gmail.com (I. Kalinina); al.zhebko@gmail.com (O. Zhebko); alex.gozhyj@gmail.com (A. Gozhyj); tetianagann@gmail.com (T. Hannichenko)

ORCID: 0000-0002-7421-3565 (P. Bidyuk); 0000-0001-8359-2045 (I. Kalinina); 0009-0009-1604-5952 (O. Zhebko); 0000-0002-3517-580X (A. Gozhyj); 0000-0001-7597-6946 (T. Hannichenko)



One new, effective approach to solving machine learning tasks (classification and regression) is the use of model ensembles [5-6]. Ensembles are a process in which different and independent models are combined to get the best result. This approach makes it possible not to rely on one single model, which may be overtrained or have other shortcomings. The ensemble technique is actually a technique of combining different models to create one "optimal" one. This allows you to mitigate the trade-off between bias and variance when selecting the optimal model.

No machine learning model is perfect. In order to understand where and how a model becomes wrong, the error of a machine learning model can be broken down into three parts: inherent error, bias error, and variance error. Inherent error is an error in the model that occurs due to the presence of noise in the data set, or incorrect formulation of the task, or bad training data with measurement errors and the presence of extraneous factors. The only way to reduce the error in machine learning models is to choose the model with the smallest error.

Bias is the inability of the model to learn enough information about the relationship between model features and labels, and variance reflects the inability of the model to generalize to new examples. A model with a high bias is considered inaccurate. A model with a high variance is overfitted to the training data and is considered over trained [7]. Any model aims to obtain low bias and variance, but in practice it is very difficult to achieve both.

The use of ensemble methods in solving classification problems is an effective way to improve classification accuracy. To build an ensemble, several methods are used, each of which provides different accuracy. The ensemble approach combines the results of individual classification methods and provides better accuracy compared to using a single classifier.

To build an ensemble, several techniques are used to aggregate the results of the underlying models, each of which provides different accuracy. *Bagging* [9,10], *boosting* [9,11], and *stacking* [9,12] are the most common approaches to building ensembles.

Bagging is a method of parallel training of base models that is suitable for different models that are considered to be independent of each other. The use of the bagging method helps to reduce the variance in the base models. In [13], various aspects of the bagging process are investigated. The method generates sample data for training from a dataset. This is achieved by randomly sampling with replacement of the original data set. Replacement sampling can repeat some observations in each new training dataset. Each item in the bagging is equally likely to appear in the new dataset. In [10], this type of ensemble is used to train several models in parallel. The average of all predictions from different ensemble models is calculated. The classification takes into account the majority of votes received by the voting mechanism. Different variants of building ensembles based on Bagging are presented in [14,15].

In recent years, the *Boosting* method of creating ensembles has been widely covered [11,13]. It is a sequential aggregation method that iteratively adjusts the weight of an observation according to the latest classification. If an observation is misclassified, it increases the weight of that observation. The term "boosting" refers to algorithms that turn a weak model into a stronger one. This reduces the bias error and creates reliable predictive models [16]. Data points that are incorrectly predicted during each iteration are identified and their weights are increased. In [17], the Boosting algorithm assigns weights to each resulting model during training. The model with the best results of predicting the training data is assigned a higher weight. If the provided input is inappropriate, its weight is increased. The goal behind this is to make a future hypothesis more likely to classify it properly by combining the entire set to finally turn weak models into better performing models [18].

The third method of creating ensembles is *Stacking*. This ensemble technique works by applying the aggregated predictions of several underlying models within a *metamodel* so that better forecasting results can be achieved [12]. Stacking is also known as generalization with pooling and is an extended form of the model averaging ensemble technique in which all sub models participate equally according to their performance weights and create a new model with better predictions. This new model is placed on top of the others, which is the reason why it is referred to as stacking [14].

In [6,8], stacking model architectures are designed in such a way that they consist of two or more base models and a metamodel that combines the predictions of the base models. These base models are called level 0 models, and the metamodel is called a level 1 model. Papers [13,17-19] present the results of studies of multilevel structures of model ensembles. In [13], the methods of a joint ensemble include input (training) data, primary level models, primary level forecast, secondary level model, and

final forecast. The authors propose a decision support system based on a two-level classifier that uses a weighted sum at both levels of aggregation. The weights are calculated based on the F-measure of each of the basic algorithms.

Problem statement. The purpose of this paper is to study ensemble methods for solving classification problems and to develop a classification system based on a two-level ensemble of models.

2. Classification system based on ensemble methods

The main approaches to ensemble classification, as well as the main methods for building classifiers, are discussed in detail in [5-18]. It is worth highlighting the features that prove the effectiveness of using ensemble methods in solving classification tasks in machine learning:

- *statistical* - the ability to average forecasts based on basic classifiers and combine their capabilities to achieve high accuracy;
- *computational* - the ability to rationally use computing resources in the process of applying classifiers for which it is difficult to select parameters with a large sample of data (neural networks, decision trees);
- *representative* - by using an ensemble structure and combining "weak models", a better solution can be obtained.

Ensemble structures are divided into two categories [5-18]: homogeneous ensemble structures and heterogeneous ensemble structures. A homogeneous ensemble structure uses base classifiers of the same type, while a heterogeneous ensemble structure uses base classifiers of different types. The main idea of ensemble classifiers is that they work better than their components when the base classifiers are not identical. A prerequisite for the usefulness of the ensemble approach is that the base classifiers must have a significant level of disagreement, which make errors independently of each other. The limitations of homogeneous ensemble frameworks can be overcome by using heterogeneous ensemble frameworks. Creating an ensemble is usually a two-step process: a set of different base models are generated by running different learning algorithms on the training data, then the generated models are combined into an ensemble. Numerous studies have shown that the strength of a heterogeneous ensemble is related to the performance of the underlying classifiers and the lack of correlation between them.

To solve the classification problem, a two-level ensemble structure is proposed, which makes it possible to effectively build an ensemble of models to further improve the prediction results on different datasets.

2.1. Structure of a two-level ensemble for solving the classification task

To solve the forecasting tasks, a two-level ensemble learning structure was proposed, as shown in Figure 1. The ensemble structure consists of a base model training unit and a two-level ensemble training unit, each of which interacts with a model quality assessment unit. The base model training unit combines independent, parallel-trained classification models. After training and evaluation, the models are divided into two groups with respect to variance and bias estimates. The models with high bias are selected for the first level of ensemble learning. The models with high variance are selected for the second level of ensemble learning.

At the first level, stacking was chosen as an ensemble method. This is a meta-algorithm that combines several heterogeneous machine learning models and acts as a way to reduce bias. Also, depending on a well-chosen metamodel, it is possible to reduce the variance. To achieve a compromise between bias and variance, the second level of ensemble learning uses bagging, which averages the predictions of the high-dispersion baseline methods. One of the models at the second level is the result of stacking. This gradual refinement of the forecast at the end gives a better result.

2.2. Dataset description and pre-processing

Two datasets were used to create a classification system based on model ensembles: *E-Commerce Shipping Data* and *Airfoil Self-Noise Data Set*. The first dataset contains information about shipping data obtained from an international E-Commerce company in India [19]. The data is available in the Kaggle machine learning repository. The purpose of the classification is to identify factors associated with the risk of late delivery of pre-orders to customers and information about the recipients of goods.

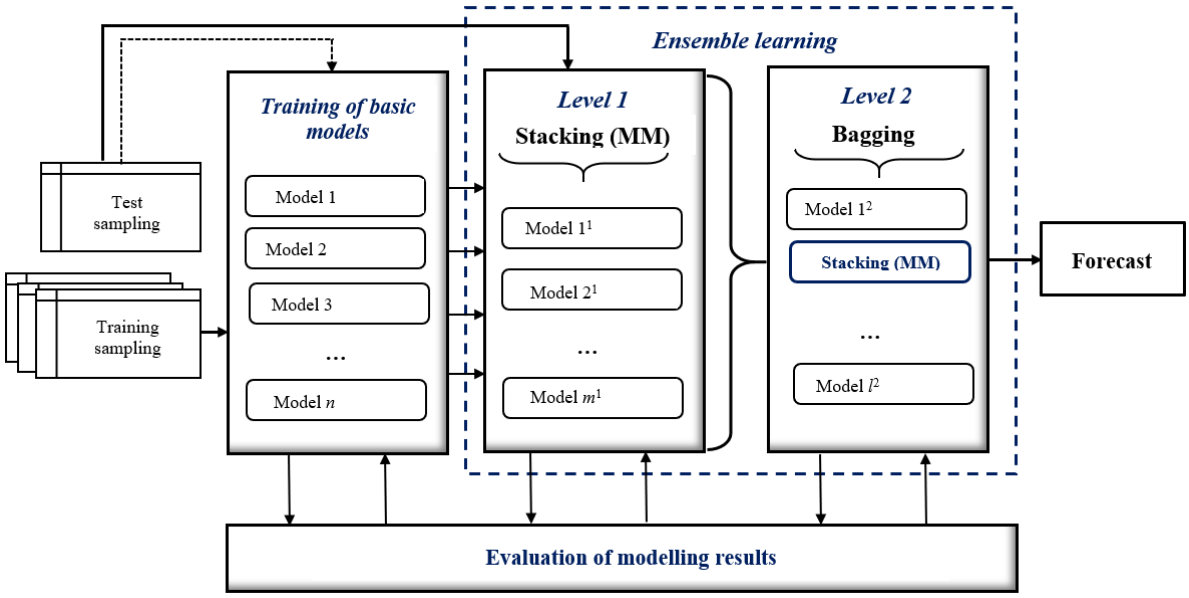


Figure 1: General structure of the two-level ensemble

The dataset includes 10999 examples of product delivery, as well as 12 variables. This is a set of numerical and nominal attributes that define the characteristics of the product and the customer. The data contains information such as: customer identification number; warehouse unit; method; customer support calls (number of calls made by customers to request the shipment of goods); customer rating (from 1 to 5); product value (product value in US dollars); previous purchases (number of previous purchases); product importance (low, medium, high); customer gender (male and female); discount; weight in grams; and delivery of goods.

Data preprocessing includes checking for missing, non-numeric, and anomalous values, converting categorical data to numeric data, selecting features, and normalising data. Non-numeric and missing values in the dataset are checked using the *NaValue* and *BlankValue* functions. The check reveals that there are no non-numeric or missing values in any attribute in the dataset selected for research. The interquartile range (IQR) method was used to check the data set for anomalous values. It was found that there are no outliers in the data set. Factor variables are converted to numeric variables: *Warehouse_block* (A→1, B→2, C→3, D→4, E→5), *Mode_of_Shipment* (Flight→1, Ship→2, Road→3), *Product_Importance* (high→1, medium→2, low→3), and *Gender* (F→1, M→2). Since most machine learning models take numeric values as input.

The next step is to normalise the data using the minimum-maximum method. In this case, the variables *Cost_of_the_Product*, *Discount_offered* and *Weight_in_gms* have quite large values and all the data from the set must be brought to a common scale without losing information about the difference in ranges.

The correlation matrix indicates that *Customer_care_calls* has a strong relationship with *Cost_of_the_Product* and *Discount_offered* has a strong relationship with *Weight_in_gms*. The vector *Reached.on.Time_Y.N* indicates whether the company was able to deliver a certain product to the customer on time or with a delay. Overall, approximately 60% of the goods in this dataset were delivered late.

The second dataset, *Airfoil Self-Noise Data Set*, is designed to study the aerodynamic properties of materials [20]. The NASA data set includes NACA 0012 airfoils of different sizes at different speeds and angles of attack in a wind tunnel. The profile span and observer position were the same in all experiments. The dataset consists of 1506 observations and 6 attributes: frequency in hertz; angle of attack in degrees; chord length in meters; incoming flow velocity in meters per second; and displacement thickness on the suction side in meters. The resultant value is the scaled sound pressure level in decibels.

To solve the binary classification task, an additional binary variable was created instead of the dependent variable. It takes a value of 1 if the dependent variable is greater than its own median, and 0 if it is less than its own median. The data preprocessing steps for the second dataset are the same as for the first dataset.

2.3. Quality metrics for classifiers

Assessing the quality of classifiers plays an important role in building and selecting a classification model. Many quality metrics are used in machine learning tasks. It all depends on the task, the models, and the presence of class imbalance in the initial dataset. Thus, choosing an appropriate evaluation metric is an important key to obtaining an optimal classifier. Typically, many classifiers use accuracy as a measure to select the optimal model during training. However, it is more appropriate to use several quality metrics.

In real-world classification tasks, a model usually cannot be 100% correct. Thus, when evaluating a model, it is useful to know not only how wrong the model was, but also in what respect the model was wrong.

The most common performance metrics take into account the model's ability to distinguish one class from the others. In this case, the class of interest is called positive, while the others are called negative. The relationship between positive and negative class predictions can be represented as a 2x2 discrepancy matrix, which shows whether the predictions belong to one of the four categories [21-23]:

- True Positive (TP) - correctly classified as a class of interest;
- True Negative (TN) - correctly classified as not belonging to the class of interest;
- False Positive (FP) - incorrectly classified as a class of interest;
- False Negative (FN) - incorrectly classified as not belonging to the class of interest.

Such a mismatch matrix is the basis for many of the most important model performance metrics [21]. A 2x2 mismatch matrix can be used to formalize the definition of prediction accuracy (sometimes called success rate) [22]:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \quad (1)$$

In this formula, TP, TN, FP, and FN denote the number of times the model's predictions fall into each of the respective categories. Thus, the accuracy is the ratio of the sum of true positive and true negative values to the total number of forecasts [21]. This metric is affected by the presence of an imbalance of classes in the original data. Therefore, other indicators are used to assess the quality of classifiers.

The error rate, or the proportion of incorrectly classified examples, is defined as follows [21]:

$$error\ rate = \frac{FP + FN}{TP + TN + FP + FN} = 1 - accuracy. \quad (2)$$

The error rate can be calculated as 1 minus the accuracy. For example, if a model is correct in 95% of cases, then it is wrong in 5% of cases [21].

Kappa statistics adjusts the accuracy value to account for the fact that a correct prediction can be made by chance. This is especially important for datasets with a severe imbalance of classes, as a classifier can get a high accuracy value only because it randomly guesses the most frequent class. A classifier can achieve a high kappa statistic only if it makes correct predictions more often than with this simplified strategy [22].

Below is a formula for calculating the kappa statistic. In this formula, Pr(a) means the proportion of real correspondence, and Pr(e) means the expected correspondence between the predictive classifier and the true values, provided they are randomly selected [23]:

$$Kappa = \frac{Pr(a) - Pr(e)}{1 - Pr\ (e)}\ . \quad (3)$$

The sensitivity of a model is the proportion of correctly classified positive examples. Therefore, the sensitivity is calculated as the number of true positive predictions divided by the total number of positive outcomes classified both correctly (true positive) and incorrectly (false negative) [22].

$$sensitivity = \frac{TP}{TP + FN}\ . \quad (4)$$

The specificity of a model (or the frequency of true negative predictions) is the proportion of correctly classified negative examples. Similarly, to sensitivity, specificity is calculated as the number of true negative predictions divided by the total number of negative outcomes, both true negative and false positive [22].

$$specificity = \frac{TN}{TN + FP}\ . \quad (5)$$

Accuracy (or the predictive value of positive results) is defined as the proportion of actual positive examples that are predicted to be positive. In other words, how often does the model correctly predict a positive class? An accurate model will predict a positive class only in those cases that are truly likely to be positive. In the case of a non-bankrupt filter, high accuracy means that the model is able to accurately filter only non-bankrupts, while skipping bankrupts [23].

$$precision = \frac{TP}{TP + FP}\ . \quad (6)$$

Recall, on the contrary, is an indicator of how complete the results are. Completeness is defined as the proportion of true positive predictions in the number of positive predictions [23]. A model with a high level of completeness captures most of the positive examples, which means that it has a wide coverage.

$$recall = \frac{TP}{TP + FN}\ . \quad (7)$$

A measure of model performance that combines accuracy and completeness into a single number is called the F-measure (also called F1 or F-score). The F-measure combines accuracy and completeness by using a harmonic mean, a type of average that determines the rate at which a quantity is measured. The F-measure is calculated using the formula [21]:

$$F - measure = \frac{2 \times precision \times recall}{precision + recall} = \frac{2 \times TP}{2 \times TP + FP + FN}\ . \quad (8)$$

ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model for all possible values of classification thresholds. This curve is a graph of two parameters [21]:

- the proportion of true positive examples (True Positive Rate);
- the proportion of false positive examples (False Positive Rate).

The proportion of true positive examples (TPR) is synonymous with recall and is therefore defined as follows [23]:

$$TPR = \frac{TP}{TP + FN} \times 100\%. \quad (9)$$

The false positive rate (FPR) is defined as follows:

$$FPR = \frac{FP}{FP + FN} \times 100\%. \quad (10)$$

The ROC curve is a graph of TPR and FPR at different values of classification thresholds. Lowering the classification threshold classifies more examples as positive, while increasing the number of false positives and true positives.

AUC stands for Area under the ROC Curve. The AUC measures the entire two-dimensional area under the ROC curve, i.e. it calculates the integral from (0,0) to (1,1). AUC provides an aggregate measure of performance for all possible values of the classification threshold. One way to interpret the

AUC is to think of it as the probability that the model ranks a random positive example better than a random negative example.

In practice, a system of quality indicators is usually used. Most often, a data scientist evaluates all the indicators and selects the most effective ones.

2.4. Results of the classifiers' work

The main advantage of the two-level ensemble is the systematic use of ensemble methods and the selection of base classification models for each level of ensemble learning. In the block of training the base models (Fig. 1), the following models were selected as basic classification models: decision trees (decision tree C5.0); naive Bayesian classifier (2 distribution classes: on-time and off-time delivery); linear discriminant analysis and quadratic discriminant analysis; logistic regression (contains the evaluation parameters, standard deviation of the arithmetic mean, standardized z score and p-value of the probability of each model attribute, as well as the overall zero and final deviation); Support Vector Machine (*cross* parameter 10-fold cross-validation of training data, RBF kernel for training); Nearest Neighbors Method (k=2), Artificial Neural Networks (hidden parameter represents one hidden layer with 2 neurons respectively, activation function is logistic or sigmoid) and Random Forest Model [24-27].

Most of the models have satisfactory performance, but the result needs to be improved as the company seeks to improve its sales and delivery services. Therefore, a two-level ensemble model is built to improve the result. As input data, the test scores of the basic models are selected and added to the test set.

In the two-level ensemble learning block, stacking and bagging are used sequentially. At the *first level*, the stacking method is based on a logistic regression model. Model stacking is an effective ensemble method. The forecasts generated by the base models are used as input for training at the first level.

The basic models of the first level are:

- decision trees (DT);
- naive Bayesian classifier (NB);
- linear discriminant analysis (LDA);
- logistic regression (LR);
- nearest neighbor method (KNN);
- artificial neural networks (ANN).

The first layer: Stacking. To create an ensemble model of the first layer of basic models with the variable *Reached.on.Time_Y.N* as a response and all other variables as predictors, the caret package is used. Basic information of the ensemble model of the first layer: logistic regression as a metamodel, 10 predictors, two classes of the response variable ("no" and "yes"). Figure 2 shows a code snippet for implementing the stacking-based ensemble.

```
# First level: Stacking
control <- trainControl(method="repeatedcv", number=10, repeats=3,
           savePredictions=TRUE, classProbs=TRUE)

predictors<-c('testPrediction_tr','testPrediction_NB','testPrediction_LR',
           'testPrediction_LDA','testPrediction_KNN','testPrediction_ANN')

models<-caretList(delivery_test[,predictors],
          delivery_test$Reached.on.Time_Y.,trControl=control,
          methodList=c("glm"))

stackControl <- trainControl(method="repeatedcv", number=10, repeats=3,
              savePredictions=TRUE, classProbs=TRUE)

stack.glm <- caretStack(models, method="glm", trControl=stackControl)

summaey(models)
stack.glm
```

Figure 2: Code for implementing an ensemble based on stacking

The *second level* is a bagging method based on the Bagged CART algorithm. The algorithm creates N regression trees using M initial training sets and averages the resulting predictions. These trees are grown deeply and are not pruned. Each individual tree has a high variance but a low error. Averaging N trees reduces the variance. The predicted values for the observations are the mode (classification) or mean (regression) of the trees. One of the disadvantages of Bagged Trees is that a small number of additional training observations can dramatically change the prediction performance of the trained tree.

The base models of the second layer are:

- the first level model (Stacking(LR));
- Random Forest model (RF);
- quadratic discriminant analysis (QDA).

The second layer: Bagging. The *caret* package is used to create an ensemble model of the second layer of basic models with the variable *Reached.on.Time_Y.N* as a response and all other variables as predictors. The *treebag* method works best with algorithms that have high variance, such as decision trees. Figure 3 shows a code snippet for implementing a bagging-based ensemble.

```
# Second level: Bagging
predictorsBag<-c('testPrediction_stack', 'testPrediction_svm',
                 'testPrediction_rF','testPrediction_QDA')

controlBag <- trainControl(method="repeatedcv", number=10, repeats=3)
bagCART_model<-train(delivery_test[,predictorsBag],
                delivery_test$Reached.on.Time_Y.,
                method='treebag',trControl=controlBag)
bagCART_model
```

Figure 3: Code for implementing an ensemble based on begging

Table 1 shows the values of the performance indicators for the first dataset. These data take into account the model's ability to distinguish one class from another (prediction accuracy, kappa statistic, sensitivity and specificity, precision and completeness, F-measure and area under the ROC curve).

Table 1
Model performance indicators for dataset #1

Model type	accuracy	Kappa	sensitivity	specificity	precision	recall	F-measure	AUC ROC
Decision Tree	0,69	0,44	0,51	0,96	0,96	0,51	0,67	0,74
NB	0,67	0,36	0,51	0,8	0,81	0,51	0,67	0,69
LR	0,60	0,27	0,56	0,7	0,58	0,6	0,57	0,64
LDA	0,64	0,27	0,68	0,59	0,71	0,68	0,69	0,63
KNN	0,60	0,17	0,67	0,5	0,66	0,05	0,67	0,68
ANN	0,33	-0,45	0,05	0,44	0,04	0,63	0,04	0,58
Stacking (LR)	**0,70**	**0,45**	**0,59**	**0,98**	**0,97**	**0,59**	**0,68**	**0,75**
RF	0,67	0,34	0,63	0,72	0,96	0,51	0,69	0,68
SVM	0,67	0,35	0,6	0,77	0,79	0,67	0,68	0,68
QDA	0,67	0,38	0,45	0,98	0,97	0,45	0,61	0,72
Bagging	**0,88**	**0,77**	**0,81**	**0,98**	**0,98**	**0,81**	**0,89**	**0,90**

The data in Table 1 show that decision trees and the quadratic discriminant analysis method have the best specificity and accuracy scores, while decision trees have the highest precision. The F-measure has the highest value for the random forest and linear discriminant analysis results, and the area under the ROC curve is the largest for the classification results using decision trees. In general, the baseline classifiers performed average on this dataset and need to be improved. The artificial neural network gave the worst results.

By using stacking to combine the six results of the baseline classifiers, the overall result is improved. Bagging to combine three results of the basic classifiers and the result of the first level model - stacking - significantly improved the overall result for all evaluation metrics.

Thus, at the first stage, combining some basic models increased the accuracy to 70%, and at the second stage, combining some basic models and the stacking model into an ensemble model using bagging increased the model's accuracy to 88%.

Table 2 shows the values of classification error rates for the models in dataset 2, which are necessary for the correct selection of base models for the first and second levels of ensemble learning. Table 3 shows the values of performance indicators for the second dataset.

Table 2

Frequency of classification errors for the models of dataset #2

Model type	The frequency of errors	
	On the training sample of data, %	On a control sample of data, %
Decision Tree	22,2	26,4
NB	25,5	29,8
QDA	14,8	23,2
LR	24,4	28,9
SVM	23,7	25,5
KNN	20,5	21,1
ANN	8,3	22,2
RF	20,8	22,1

Table 3

Model performance indicators for dataset #2

Model type	accuracy	Kappa	sensitivity	specificity	precision	recall	F-measure
Decision Tree	0,736	0,475	0,600	0,878	0,678	0,878	0,765
NB	0,711	0,419	0,800	0,617	0,747	0,647	0,676
QDA	0,745	0,488	0,808	0,678	0,772	0,678	0,722
LR	0,770	0,540	0,775	0,765	0,765	0,765	0,765
KNN	0,774	0,549	0,791	0,756	0,777	0,757	0,767
ANN	0,240	-0,517	0,275	0,209	0,216	0,209	0,212
Stacking (LR)	**0770**	**0,542**	**0,708**	**0,835**	**0,733**	**0,835**	**0,780**
RF	0,813	0,625	0,817	0,809	0,809	0,809	0,809
SVM	0,787	0,575	0,742	0,835	0,756	0,835	0,793
Bagging	**0,817**	**0,634**	**0,825**	**0,809**	**0,816**	**0,809**	**0,812**

For the second dataset, at the first stage, combining the basic models increased the accuracy to 77%, and at the second stage, combining some basic models and the stacking model into an ensemble model using bagging increased the model accuracy to 82%.

Thus, the use of a two-level ensemble increases the efficiency of classification models.

3. Conclusions

The general structure of a two-level ensemble was developed. Based on the use of the two-level ensemble learning structure in the processing of two datasets, the classification quality was improved. The procedures for processing the datasets included identifying and describing the key quality characteristics of the models, selecting a metric, selecting the base models, selecting parameters for the base models and ensemble methods. Preliminary data processing was performed. The basic datasets are divided into training and test samples, and input variables are generated. The results of applying simple classifiers and the ensemble of the two-level classification model are presented, and the efficiency of the developed classification models is evaluated.

Based on the analysis of metrics for assessing the quality of the basic classifiers, it is determined that they need to be improved. A two-level ensemble scheme is used for improvement. At the first level of the ensemble, stacking was used to reduce the bias of the base models. This resulted in a preliminary improvement in classification quality. At the second level, bagging was used to reduce the variance of the base models. Thus, the use of an ensemble-based classifier solved the problem of finding a compromise between bias and variance, which improved the classification results using machine learning models.

References

[1] S. Marsland, Machine Learning: An Algorithmic Perspective. Palmerston North: Massey University, 2015, 452 p.

[2] L. Wang, L. Cheng, G. Zhao, Machine Learning for Human Motion Analysis. Anhui: IGI Global, 2009, 318 p.

[3] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. California: Springer–Verlag, 2009, 746 p.

[4] Artificial Intelligence: A Modern Approach. URL: https://towardsdatascience.com/understanding–the–bias–variance–tradeoff (application date: 20.12.2022).

[5] D. Opitz, R. Maclin, Popular ensemble methods: An empirical study: journal of Artificial Intelligence Research, No. 11. El Segundo, 1999, pp. 169–198.

[6] Ensemble Methods to Optimize Machine Learning Models. URL: https://hub.packtpub.com/ensemble–methods–optimize–machine–learning–models (application date: 20.12.2022).

[7] Understanding the Bias–Variance Tradeoff URL: http://scott.fortmannroe.com/docs/BiasVariance.html (application date: 20.12.2022).

[8] Dietterich T., Ensemble Methods in Machine Learning. URL: http://web.engr.oregonstate.edu/~tgd/publications/mcs–ensembles.pdf (application date: 25.12.2022).

[9] Ensemble methods: bagging, boosting and stacking. URL: https://towardsdatascience.com/ensemble–methods–bagging–boosting–andstacking-c9214a10a205 (application date: 27.12.2022).

[10] F. Moretti, S. Pizzuti, S. Panzieri, M. Annunziato, Urban traffic flow forecasting through statistical and neural network bagging ensemble hybrid modeling, Neurocomputing, 2015.

[11] M. J. Kim, D. K. Kang, H. B. Kim, Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction, Expert Syst. Appl. 42 (3), 2015, pp. 1074–1082.

[12] S. Kang, S. Cho, P. Kang, Multi-class classification via heterogeneous ensemble of one-class classifiers, Eng. Appl. Artif. Intell. 43, 2015, pp. 35–43.

[13] S. Bashir, U. Qamar, F. H. Khan, IntelliHealth: A medical decision support application using a novel weighted multi-layer classifier ensemble framework. Journal of Biomedical Informatics, 2016, Vol. 59, pp.185- 200

[14] Zhi–Hua Zhou, Ensemble Learning. URL: https://cs.nju.edu.cn/zhouzh/ zhouzh.files/ publication/springerEBR09.pdf (application date: 20.12.2022).

[15] B. R. Shah, L. L. Lipscombe, Clinical diabetes research using data mining: a Canadian perspective, Can. J. Diabetes 39 (3), 2015, pp. 235–238.

[16] Improvements on Cross–Validation: The 632+ Bootstrap Method. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1997.10474007#.U2o7MVdMzTo (application date: 22.12.2022).

[17] A. Ahmad, G. Brown, Random ordinality ensembles: ensemble methods for multi-valued categorical data, Inf. Sci. 296, 2015, pp. 75–94.

[18] B. Sluban, N. Lavrac, Relating ensemble diversity and performance: a study in class noise detection, Neurocomputing 160, 2015, pp. 120–131.

[19] E–Commerce Shipping DataSet. URL: https://www.kaggle.com/datasets/+prachi13/ customer-analytics (application date: 15.11.2022)
[20] Machine Learning Repository. URL: http://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise. M.
[21] M. Hossin, A Review on Evaluation Metrics for Data Classification Evaluations. Article in International Journal of Data Mining & Knowledge Management Process (IJDKP), 2015, Vol.5, No.2, pp. 1-11. DOI: 10.5121/ijdkp.2015.5201.
[22] Y. Liu, Y. Zhou, Sh. Wen, Ch. Tang, A Strategy on Selecting Performance Metrics for Classifier Evaluation. 20 International Journal of Mobile Computing and Multimedia Communications, 2014, 6(4), pp. 20-35. DOI: 10.4018/IJMCMC.2014100102.
[23] I. H. Sarker, A.S.M. Kayes, P. Watters, Efectiveness analysis of machine learning classification models for predicting personalized context aware smartphone usage. Journal of Big Data, 6:57, Open Access, Springer Open, 2019, pp. 1-28.
[24] M. Maniruzzaman, M.J. Rahman, B. Ahammed, Classification and prediction of diabetes disease using machine learning paradigm. Health information science and systems, 2020, Vol. 8, Texas, pp. 1–14.
[25] Zhan Zh. Introduction to machine learning: k-nearest neighbors. Vienna: Ann Transl Med, 2016, 218 p.
[26] P. Bidyuk, A. Gozhyj, I. Kalinina, V. Vysotska, Methods for forecasting nonlinear non-stationary processes in machine learning. In: Data Stream Mining and Processing. DSMP 2020. Communications in Computer and Information Science. 2020, Vol. 1158, pp. 470–485. Springer, Cham, (2020). https://doi.org/10.1007/978-3-030-61656-4 32.
[27] P. Bidyuk, I. Kalinina, A. Gozhyj, An Approach to Identifying and Filling Data Gaps in Machine Learning Procedures. International Scientific Conference “Intellectual Systems of Decision Making and Problem of Computational Intelligence” ISDMCI 2021: Lecture Notes in Computational Intelligence and Decision Making, 2021, pp. 164–176.