# Classification of Dynamic Objects Using a Multilayer Perceptron

Ivan Peleshchak[1], Diana Koshtura[1], Mykhailo Luchkevych[1], Volodymyr Tymchuk[2]

[1]*Lviv Polytechnic National University, 12 Stepana Bandery Street, Lviv, 79000, Ukraine*
[2]*National Academy of Land Forces named after Hetman Petro Sahaidachny, 32 Heroiv Maidanu Street, Lviv, 79026, Ukraine*

**Abstract**

The article discusses the task of classifying projectiles (A - (a-ammunition), A/M - (a/m-ammunition), A/P - (armor-piercing), A/PC - (armor-piercing-cumulative), M - (m-ammunition), P - (armor-piercing ammunition)) using an MLP perceptron with the aim of determining the type of projectile (artillery or missile). This allows: the military to quickly respond and apply an effective strategy and tactics during military operations; civilian organizations and law enforcement agencies to determine whether a projectile has a potentially dangerous nature that can pose a threat to the safety of citizens; in regions with military conflict or after its completion, it is important to identify unexploded projectiles for their safe removal and disposal; scientific institutions to study the properties and characteristics of new types of projectiles.

The classification of projectiles was carried out using an MLP perceptron with two-, three-, five-, six-hidden layers, which have respectively the number of neurons (33 and 8), (33, 16 and 8), (33, 16, 16 and 8), (33, 16, 16, 16 and 8) with activation functions relu, logistic, tanh.

The dependence of the accuracy of projectile classification on the number of hidden layers for activation functions relu, logistic, tanh was experimentally investigated.

It is shown that the highest accuracy of projectile classification (95.9%) is achieved by a neural network with two hidden layers and the number of neurons 33 and 8 with Tanh activation functions respectively and an output layer with six neurons with the Softmax activation function.

The results of the study of the influence of the components of the input feature vector (x1 - position_x1, x2 - position_y1, x3 - position_h1, x4 – velocity, x5 - taget_class, x6 - explosion_x2, x7 - explosion_y2, x8 - explosion_h2, x9 – hour, x10 – minute, x11 – second, x12 - angle_big_tick, x13 - angle_small_tick, x14 - angle_degrees, x15 - angle_rotation_degrees, x16 - distance_2d, x17 - distance_3d, x18 - flight_time) on the accuracy of classification are visualized in the form of diagrams. The height, speed, and angle have the greatest influence on the values of the outputs of neurons in the output layer.

**Keywords**

Classification, Multi-Layer Perceptron, MLP, accuracy, precision, F1-score, artificial neural networks, activation functions, visualization.

## 1. Introduction

The task of neural network classification of projectiles (A - (a-ammunition), A/M - (a/m-ammunition), A/P - (armor-piercing), A/PC - (armor-piercing-cumulative), M - (m-ammunition), P - (armor-piercing ammunition) is relevant, as its solution automatically solves the problem of determining the type of projectile [1] (artillery or missile) for choosing an effective strategy and tactics during military operations, for safe removal and disposal of unexploded projectiles, and for studying the properties and characteristics of new types of projectiles.

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

ivan.r.peleshchak@lpnu.ua (I. Peleshchak); diana.a.koshtura@lpnu.ua (D. Koshtura); luchkevychmm@gmail.com (M. Luchkevych); v_tymchuk@yahoo.co.uk (V. Tymchuk)

0000-0002-7481-8628 (I. Peleshchak); 0000-0002-5665-5423 (D. Koshtura); 0000-0002-2196-252X ((M. Luchkevych); 0000-0002-3549-2813 (V. Tymchuk)

The aim of the work is to develop an optimized morphology of the MLP perceptron to ensure high accuracy metrics (accuracy, precision, recall, f1-score) when classifying projectiles. The optimization of the MLP perceptron morphology is achieved by determining the necessary number of hidden layers [2] and the number of neurons in them with corresponding activation functions.

## 2. Related works

In the context of modern technological products and the needs of the defense sector, the task of data classification, in particular the determination of types of projectiles, is an important task [3]. A literature review indicates the widespread use of machine learning methods, particularly classification, in solving similar tasks in the defense sector [4].

The training sample was formed by choosing a multifunctional complex with flexible hardware and algorithmic support that easily adapts to perform various combat tasks - 1L220U-KS [5]. The following technical characteristics of this complex were considered to generate data on which the neural network model can be trained.

Main technical characteristics of the 1L220U-KS/IL220U complex [5]:

Main Specifications:

| | |
|---|---|
| Range exploration VP: | |
| ▮artillery | 30 km |
| ▮mortars | 30 km |
| ▮MLRS | 30/40 km |
| ▮tactical missiles | 55 km |
| Range control Accuracy: | |
| ▮artillery | 30 km |
| ▮MLRS | 30/40 km |
| ▮tactical missiles | 80 km |
| Throughput goals | 50 per minutes |

**Figure 1:** Main technical characteristics of the 1L220U-KS/IL220U complex

After analyzing the characteristics of such a complex, the following tasks can be identified for which data can be generated based on materials [5]: classification of target types, classification of types of vehicles, reconnaissance of the range. Data will be generated for the classification of projectiles.

## 3. Materials and methods

The coordinates position_$x_1$ (x coordinates from where the projectile was launched), position_$y_1$ (y coordinates from where the projectile was launched), position_$h_1$ (height h from where the projectile was launched), explosion_$x_2$ (x coordinates where the explosion occurred), explosion_$y_2$ (y coordinates where the explosion occurred), explosion_$h_2$ (height h where the explosion occurred) are initially generated in the SK-42 coordinate system - a cartographic coordinate system [6].

It divides the earth's surface into 60 numbered zones each 6 degrees of longitude wide. The values of the extreme meridians of six-degree zones will have values: the first zone 0 - 6 °, the second zone 6 - 12 °, the third zone 12 - 18 ° and so on.

To convert coordinates from the SK-42 system to the WGS84 system (longitude and latitude), you can use the converter: https://sk42.org/en/. This converter uses the ST_Transform() postgis method, which converts the output coordinates of SK-42, the SK-42 zone to latitude and longitude.

For example, coordinates: SK-42: 5445007, 07366730 with zone: SK-42: 7 will be converted to latitude: 49.122917949468516, longitude: 37.172574626199335. In this work, coordinates in the SK-42 system will be used for training and testing.

**Table 1**
**Vector of input features for projectile classification (A – 1, A/M – 2, A/P – 3, A/PC – 4, M – 5, P – 6)**

| № | Components of the input feature vector | | Units of measurement for input feature components |
|---|---|---|---|
| 1 | position_x1 | x coordinates from where the projectile was launched | SK-42 |
| 2 | position_y1 | y coordinates from where the projectile was launched | SK-42 |
| 3 | position_h1 | Height h from where the projectile was launched | Height above sea level, measurement - meters 4 |
| 4 | velocity | Projectile speed | Measurement – m/s |
| 5 | taget_class | Projectile class | [A,A/M,A/P, A/PC,M,P] |
| 6 | explosion_x2 | x coordinates where the explosion occurred | SK-42 |
| 7 | explosion_y2 | y coordinates where the explosion occurred | SK-42 |
| 8 | explosion_h2 | Height h where the explosion occurred | Height above sea level, measurement - meters |
| 9 | hour | Hour of projectile launch | Hours in 24-hour time format (16:45:56) |
| 10 | minute | Minute of projectile launch | Minutes in 24-hour time format (16:45:56) |
| 11 | second | Second of projectile launch | Seconds in 24-hour time format (16:45:56) |
| 12 | angle_big_tick | Large division of the protractor | One division of the protractor = 3.6’. One large division of the protractor = 360’ = 6° |
| 13 | angle_small_tick | Small division of the protractor | One division of the protractor = 3.6’. One large division of the protractor = 360’ = 6° |
| 14 | angle_degrees | Angle in degrees | Measurement - degrees, from 0 to 360 |
| 15 | angle_rotation_degrees | Angle of rotation between position and point of explosion | Measurement - degrees, from 0 to 360 |
| 16 | distance_2d | 2D distance between the initial position of the projectile and the point of its explosion | Measurement - meters |
| 17 | distance_3d | 3D distance between the initial position of the projectile and the point of its explosion | Measurement - meters |
| 18 | flight_time | Flight time of the projectile | Measurement - seconds |

The division of the protractor ("thousandth") is a central angle, the length of the arc of which is equal to $1/6000$ of the length of the circle. The formula for calculating the angle in degrees (angle_degrees) uses both the large and small part of the angle, which allows you to get complete information about the position and orientation of the data.

The length of the arc of angle ∪AB of one protractor division is equal to:

$$\cup AB = \frac{2\pi R}{6000} = \frac{2 \cdot 3{,}14 \cdot R}{6000} = \frac{1}{955} R = 0{,}001105R \approx 0{,}001R, \quad (1)$$

During practical calculations, it is convenient to assume that the length of the arc corresponding to one division of the protractor is equal to 1/1000 of the radius by which the circle is drawn. Therefore, the division of the protractor is also called a "thousandth". A circle contains 6000 divisions of the protractor, or 6000 "thousandths". However, such rounding introduces a systematic error of 4.5% (rounded - 5%) [7].

So, the central angle, which is based on an arc equal to 0.001 R, that is, a division of the protractor, is called a thousandth.

A circle contains 360°, or 21600'. One division of the protractor is equal to 21600/6000 = 3.6'. One large division of the protractor is equal to 3.6·100 =360’ = 6°. One degree is approximately equal to 6000/360 = 16.66 p.k. = 17 p.k. (P.k. - division of the protractor). To convert angle values expressed in protractor divisions to values expressed in degrees and minutes, and vice versa, the following ratios are used: 60-00 =360°, 30-00 = 180°, 15-00 = 90°, 1-00 = 6°, 0-01 = 3.6'

**Table 2**
**Vector of output features for projectile classification (A – 1, A/M – 2, A/P – 3, A/PC – 4, M – 5, P – 6)**

| № | Output feature vector components | | Output feature vector labels |
|---|---|---|---|
| 1 | class | Projectile class | [A - 1 (ammunition), A/M - 2 (a/m-ammunition), A/R - 3 (armor-piercing), A/RC - 4 (armor-piercing-incendiary), M - 5 (ammunition), R - 6 (armor-piercing ammunition)] |

## 4. Data preprocessing

For the data preprocessing, criteria will be defined using Table 1 and Table 2.

Before the data processing for the classification of projectiles collected from the 1L220U-KS complex, it is necessary to perform the generation of new values. This is done to increase the size of the training set and help the MLPClassifier_2HL_T neural network acquire generalizing properties.

The generation of new values uses the method of pseudo-random noise with a normal distribution. New values are generated closer to the values in the training sample, taking into account the standard deviation of each characteristic.

Generation of a new value:

$$V_i^* = V_i + \sigma_i \cdot N(0,1), \quad (2)$$

where $V_i^*$ – is the new value of the i-th characteristic, $V_i$ – is the initial value of the i-th characteristic, $\sigma_i$ – is the standard deviation of the i-th characteristic, $N(0,1)$ – is the function for generating pseudo-random values with a normal distribution (mean = 0, standard deviation = 1).

Calculation of the standard deviation:

$$\sigma_i = \sqrt{\frac{\sum(V_i - \mu_i)^2}{N}}, \quad (3)$$

where $V_i$ – is the i-th value of the characteristic, $\mu_i$ – is the mean value of the i-th characteristic, N – is the size of the training sample.

In this article, data were generated: position_$x_1$, position_$y_1$, position_$h_1$, velocity, taget_class, explosion_$x_2$, explosion_$y_2$, explosion_$h_2$, hour, minute, second, angle_big_tick, angle_small_tick, angle_degrees, angle_rotation_degrees, distance_2d, distance_3d, flight_time from the study [5] for the classification of projectiles based on data from the 1L220U device. The parameter values are given in Table 1.

Normalization:

$$V' = \frac{V_i - \mu_i}{\sigma_i}, \quad (4)$$

where $V'$ - is the normalized value of the i-th characteristic, $V_i$ – initial value of the i-th characteristic, $\mu_i$ – is the mean value of the i-th characteristic, $\sigma_i$ is the standard deviation of the i-th characteristic.

Each row of the table needs to be checked for compliance with certain conditions: validity of time, coordinates, velocity, angle, and others. This allows identifying and filtering acceptable data rows.

The training dataset will contain the following characteristics:

Initial projectile coordinates: position_$x_1$: X coordinate of the initial point, position_$y_1$: Y coordinate of the initial point, position_$h_1$: Z coordinate, height of the initial projectile position.

Velocity: velocity: projectile velocity.

Target class: target_class: class of the target aimed at by the projectile (categorical variable).

Coordinates of the final projectile position: explosion_$x_2$: X coordinate of the final point, explosion_$y_2$: Y coordinate of the final point, explosion_$h_2$: Z coordinate, height of the final projectile position.

Time: hour: hour of projectile launch, minute: minute of projectile launch, second: second of projectile launch.

Angles: angle_big_tick, angle_small_tick, angle_degrees, angle_rotation_degrees.

Distances: distance_2d: 2D distance between initial and final points, distance_3d: 3D distance between initial and final points.

Flight time: flight_time: projectile flight time.

For neural network training, the launch time was divided separately into hours: hour, minutes: minute, seconds: second, and these values were saved in separate columns.

The angle value was also divided into two parts: the large (angle_big_tick) and small (angle_small_tick) divisions of the angle, calculation of the total angle in degrees (angle_degrees), using the values angle_big_tick and angle_small_tick for an accurate calculation of the angle.

These transformations are aimed at increasing the data set and reflecting new aspects that may be important for further research and modeling, as shown in Table 1.

The rotation angle (angle_rotation_degrees) is a parameter that defines the rotation angle between the position and the point of explosion, which can be important when considering the trajectory of the projectile. The distances between the position and the point of explosion are calculated to determine the spatial position of the projectile. This data can be useful in analyzing the trajectory and impact of the projectile – 2D and 3D distances.

Calculation of 2D and 3D distances between the position and the point of explosion:

$$distance_{2d} = \sqrt{(explosion_x - position_x)^2 + \left(explosion_y - position_y\right)^2}, \quad (5)$$

$$distance_{3d} = \sqrt{distance_{2d}{}^2 + (explosion_h - position_h)^2}, \quad (6)$$

## 5. The architecture of the three-layer perceptron

The three-layer perceptron architecture used for the classification of dynamic objects (A - 1, A/M - 2, A/R - 3, A/RC - 4, M - 5, R - 6) in the work is optimized in terms of the number of hidden layers and the number of neurons in them, with the following morphology: two hidden layers and one output layer. The input sensory layer has 18 neurons, because the input feature vector has 18 components; the first hidden layer has 33 neurons with Tanh activation functions; the second hidden layer has 8 neurons with Tanh activation functions; the output layer has 6 neurons with Softmax activation functions, because 6 objects are classified.

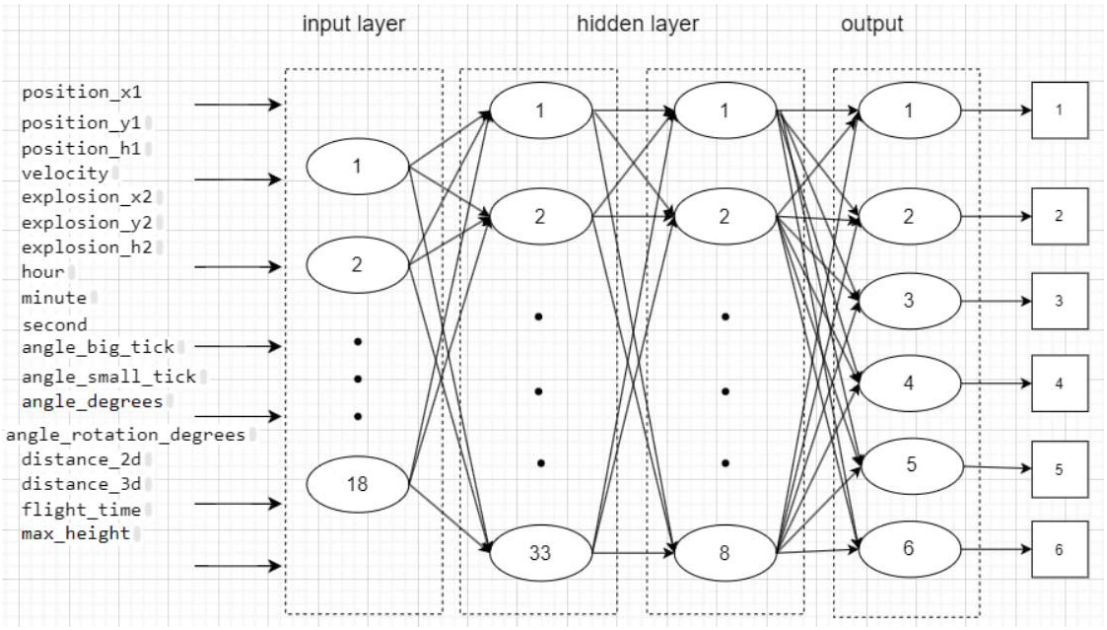


**Figure 2:** Three-Layer Neural Network Model

Model Formula:

$$y_i = f_{softmax}\left(\sum_{i=1}^{8} w_{li}\, f_{tanh}\left(\sum_{j=1}^{33} w_{ij}\, f_{tanh}\left(\sum_{k=1}^{18} w_{jk}\, x_k\right)\right)\right)\ i \in \{1;6\}, \qquad (7)$$

where $y_i$ – is an element of the output vector of probabilities related to each class of projectiles, $f_{softmax}$ – is the softmax activation function, $f_{tanh}$ – is the tanh activation function, $w_{li}$ – is an element of the weight matrix between the second hidden layer and the output layer, $w_{ij}$ – is an element of the weight matrix between the first and second hidden layers, $w_{jk}$ – is an element of the weight matrix between the input layer and the first hidden layer. The model receives the input feature vector $\bar{x} = (x_1, x_2, \dots, x_{18})$, where $x_1$ - position_$x_1$, $x_2$ - position_$y_1$, $x_3$ - position_$h_1$, $x_4$ – velocity, $x_5$ - taget_class, $x_6$ - explosion_$x_2$, $x_7$ - explosion_$y_2$, $x_8$ - explosion_$h_2$, $x_9$ – hour, $x_{10}$ – minute, $x_{11}$ – second, $x_{12}$ - angle_big_tick, $x_{13}$ - angle_small_tick, $x_{14}$ - angle_degrees, $x_{15}$ - angle_rotation_degrees, $x_{16}$ - distance_2d, $x_{17}$ - distance_3d, $x_{18}$ - flight_time.

## 6. Experimental results and their analysis

The experimental results investigated the influence of the number of hidden layers on the model's accuracy. The accuracy graph (Figure 3) showed that increasing the number of hidden layers improves the classification accuracy. The graph (Figure 3) displays three different activation functions: ReLU, Logistic, and TanH.

As seen from the graph (Figure 3), the accuracy of MLP classifiers increases with the increase in the number of hidden layers with ReLU and TanH activation functions. However, after 3 hidden layers with Logistic activation functions, the accuracy of MLP classifiers starts to decrease. This happens because complex networks can be overly adapted to the training data and may not perform well on new data [8].

In this case (Figure 3), MLP classifiers with three and two hidden layers and TanH activation functions have the highest accuracy. To optimize computational resources, it is advisable to use an MLP classifier with two hidden layers and TanH activation functions.

In the selected configuration, it showed high results on the test dataset. Tables 2 and 3 provide metrics for evaluating the accuracy of the three-layer neural network classifier using the Scikit-Learn library.

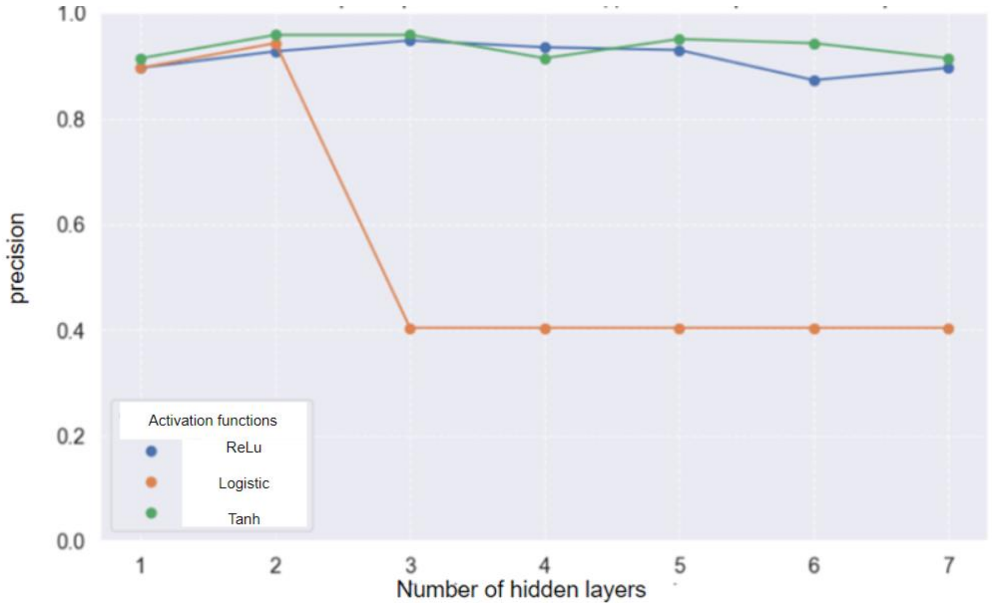


**Figure 3:** The dependence of the accuracy of projectile classification by the neural network on the number of hidden layers for 3 activation functions: ReLU, Logistic, and TanH for neurons in the first and second hidden layers.

**Table 3**
**Evaluation of the accuracy of neural network classifiers with the Scikit-Learn library on the training dataset obtained based on the relationships of formulas 2-4**

| Morphology of the Multilayer Perceptron | Training set: accuracy | Training set: precision | Training set: recall | Training set: f1-score |
|---|---|---|---|---|
| 3 hidden layers, tanh activation function | 0.977 | 0.978 | 0.977 | 0.977 |
| 2 hidden layers, tanh activation function | 0. 952 | 0. 952 | 0. 952 | 0. 951 |
| 5 hidden layers, tanh activation function | 0.986 | 0.986 | 0.986 | 0.986 |
| 3 hidden layers, relu activation function | 0.986 | 0.986 | 0.986 | 0.986 |
| 6 hidden layers, tanh activation function | 0.976 | 0.976 | 0.976 | 0.976 |
| 2 hidden layers, logistic activation function | 0.952 | 0.952 | 0.952 | 0.951 |

**Table 4**
**Evaluation of the accuracy of neural network classifiers using the Scikit-Learn library on the test dataset obtained based on the relationships of formulas 2-4**

| Morphology of the Multilayer Perceptron | Test set: accuracy | Test set: precision | Test set: recall | Test set: f1-score |
|---|---|---|---|---|
| 3 hidden layers, tanh activation function | 0.959 | 0.962 | 0.959 | 0.959 |
| 2 hidden layers, tanh activation function | 0.959 | 0.962 | 0.959 | 0.959 |
| 5 hidden layers, tanh activation function | 0.951 | 0.954 | 0.951 | 0.952 |

| | | | | |
|---|---|---|---|---|
| 3 hidden layers, relu activation function | 0.948 | 0.950 | 0.948 | 0.949 |
| 6 hidden layers, tanh activation function | 0.943 | 0.946 | 0.943 | 0.944 |
| 2 hidden layers, logistic activation function | 0.943 | 0.944 | 0.943 | 0.943 |

The model of a feedforward neural network (MLPClassifier) [8] was chosen for its ability to solve complex nonlinear classification tasks and achieve high accuracy after training. A multilayer perceptron with 5 hidden layers and TanH activation functions showed high results after training on the training set: accuracy: 0.986, precision: 0.986, recall: 0.986, f1-score: 0.986. However, the accuracy evaluation results changed on the test dataset, and the models of the neural network with three and two hidden layers and TanH activation functions showed the highest accuracy: accuracy: 0.959, precision: 0.962, recall: 0.959, f1-score: 0.959. To optimize computational resources, it is advisable to choose a neural network with two hidden layers and TanH activation functions [9].

Using train_test_split, the data was split into training (80%) and test (20%) sets for efficient model validation. The random_state=42 was set for result reproducibility. The random_state parameter in the train_test_split function determines the method of selecting a random permutation for dividing the data into training and test sets. Setting random_state to a specific value (in this case, 42) ensures that the same random permutation will be used for data splitting at each program run. This makes the data split repeatable and suitable for result reproducibility [10[].

LabelEncoder is used for numerical encoding of categorical class labels. LabelEncoder is used to transform categorical labels into numerical values. We create LabelEncoder and MinMaxScaler [11] objects, which will be used for encoding class labels and normalizing features, respectively. We use the fit method with LabelEncoder to fit the encoder to the class labels of the target variable in the dataset, this method learns the unique values in the target_class column and builds the correspondence between them and integers. This step is necessary to map class labels to integer numerical values. Scaling numerical features using MinMaxScaler ensures uniformity and improves results. Using MinMaxScaler, we normalize numerical feature values [12] to ensure that they all fall within the same range (from 0 to 1).

In particular, the accuracy on the training set is 0.979, and the accuracy on the test set is 0.959.

Besides accuracy, other metrics such as precision, recall, and F1-score also have high values: on the training set: accuracy: 0.9786, precision: 0.9788, recall: 0.9786, F1-score: 0.9787; on the test set: accuracy: 0.9585, precision: 0.9617, recall: 0.9585, F1-score: 0.9592.

The MLP model used SHAP (SHapley Additive exPlanations) [13] to assess the weight coefficient of individual features in decision-making. This helped understand which features most influence the classification of projectiles.

SHAP is used to estimate the importance of features. For each component of the input feature vector and class i, the SHAP value $\phi_i^k$ determines the weight of $k$ and is described as follows:

$$\phi_i^k(x) = \omega_0 + \sum S \subseteq K\{k\} \frac{|S|!\,(K - |S| - 1)!}{K!}[f\,(S \cup \{k\}, x) - f(S, x)], \quad (8)$$

where $\omega_0$ – is the base value, $f\,(S, x)$ – is the model output for the subset of features $S$.

Using the shap.sample function, a subset of data is determined for analysis. 100 random objects are used for this sample to reduce the number of computations required for SHAP analysis because the dataset is very large. Using shap.KernelExplainer [14], an explainer is created, which is passed the model.predict function for prediction and the X_train_sample sample. The explainer is ready to use to obtain SHAP values. SHAP uses the Shapley method to analyze the impact of each feature on the model's prediction. This allows to reveal how each feature contributes to or decreases the likelihood of a specific prediction [15].

Using the explainer.shap_values method, SHAP values are obtained, indicating how each individual feature affects the model's predictions. After executing this command, shap_values will contain a matrix of SHAP values, where each row corresponds to a sample from X_train_sample, and each column corresponds to a specific feature [16].

The values in the matrix show the influence of each feature on the model's prediction for the corresponding sample. By the sign of the value, it can be understood whether this feature contributes (positive value) or decreases (negative value) the prediction. Using shap.summary_plot, a feature importance plot is generated. This plot shows the importance of each feature, helping to understand which features have the greatest impact on the model's predictions.

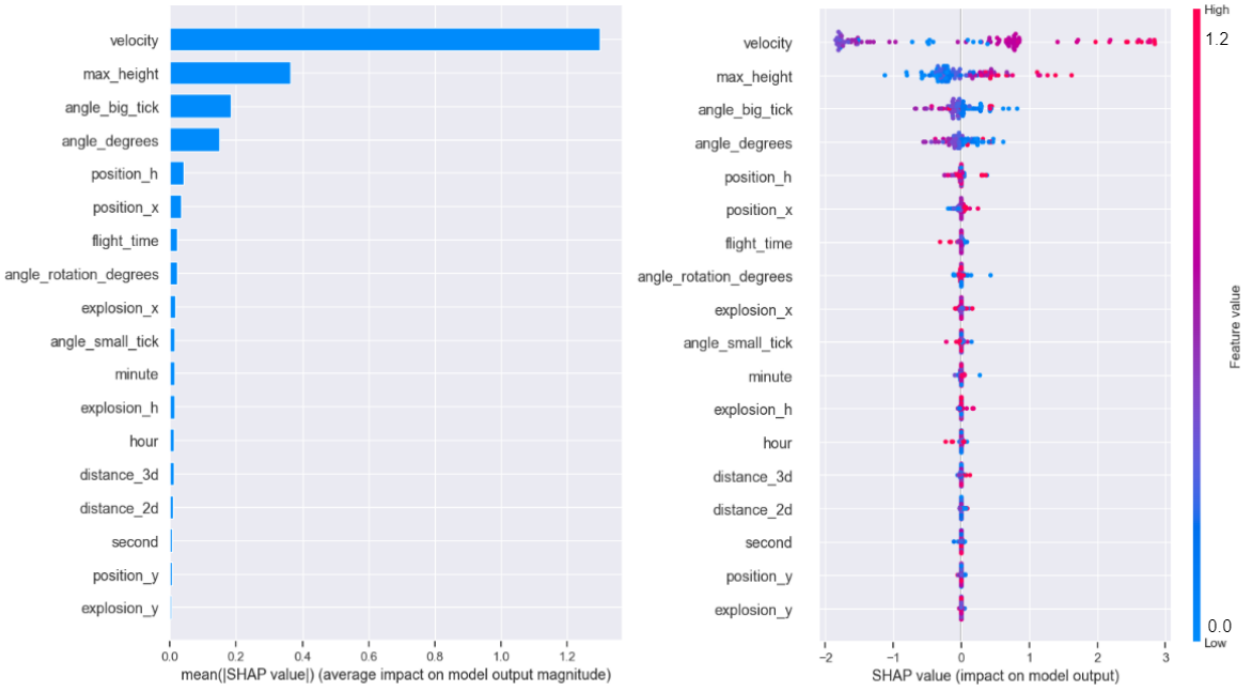


**Figure 4:** Influence of input feature components on projectile classification accuracy

Parameter values are presented on the x-axis, and SHAP values are presented on the y-axis. SHAP values represent how strongly each parameter affects the model output.

As shown in Figure 4, the parameters of altitude, speed, and angle have the greatest impact on the model output. An increase in speed by 1 unit leads to an increase in the SHAP value by 3 units. An increase in the angle by 1 unit leads to an increase in the SHAP value by 2 units [17]. These variables have the greatest impact because they directly affect the classification of the projectile [18]. Speed determines how fast the projectile flies, maximum altitude determines how high it climbs, and altitude position determines where it stops. Flight time determines how long the projectile flies before falling [19]. Other parameters also have some impact on the model output, but it is smaller. For example, an increase in maximum altitude by 1 unit leads to an increase in the SHAP value by 1 unit [20].

The plot_confusion_matrix and plot_result_area functions are used to visualize classification results, specifically to display the confusion matrix and detailed classification report.

In the work, classifiers were trained on training data and evaluated on test data [21]. The main steps include copying training and test data, initializing classifiers, and calling the perform_training_cycle function for training and evaluating each classifier. The perform_training_cycle function trains each classifier and computes quality metrics for training and test data. The results are stored in the reporting DataFrame training_report_df_fulldata and presented in Figure 5.

```
Test - Acc: 0.9845, Precision: 0.9853, Recall: 0.9845, F1: 0.9846 | DecisionTreeClassifier
Test - Acc: 0.7254, Precision: 0.6977, Recall: 0.7254, F1: 0.7025 | SVC
Test - Acc: 0.8964, Precision: 0.9017, Recall: 0.8964, F1: 0.8973 | MLPClassifier_1HL_R
Test - Acc: 0.9275, Precision: 0.9373, Recall: 0.9275, F1: 0.9299 | MLPClassifier_2HL_R
Test - Acc: 0.9482, Precision: 0.9497, Recall: 0.9482, F1: 0.9487 | MLPClassifier_3HL_R
Test - Acc: 0.9352, Precision: 0.9405, Recall: 0.9352, F1: 0.9368 | MLPClassifier_4HL_R
Test - Acc: 0.9301, Precision: 0.9359, Recall: 0.9301, F1: 0.9316 | MLPClassifier_5HL_R
Test - Acc: 0.8731, Precision: 0.8864, Recall: 0.8731, F1: 0.8759 | MLPClassifier_6HL_R
Test - Acc: 0.8964, Precision: 0.9042, Recall: 0.8964, F1: 0.8980 | MLPClassifier_7HL_R
Test - Acc: 0.8964, Precision: 0.8998, Recall: 0.8964, F1: 0.8934 | MLPClassifier_1HL_L
Test - Acc: 0.9430, Precision: 0.9445, Recall: 0.9430, F1: 0.9430 | MLPClassifier_2HL_L
Test - Acc: 0.4041, Precision: 0.1633, Recall: 0.4041, F1: 0.2326 | MLPClassifier_3HL_L
Test - Acc: 0.4041, Precision: 0.1633, Recall: 0.4041, F1: 0.2326 | MLPClassifier_4HL_L
Test - Acc: 0.4041, Precision: 0.1633, Recall: 0.4041, F1: 0.2326 | MLPClassifier_5HL_L
Test - Acc: 0.4041, Precision: 0.1633, Recall: 0.4041, F1: 0.2326 | MLPClassifier_6HL_L
Test - Acc: 0.4041, Precision: 0.1633, Recall: 0.4041, F1: 0.2326 | MLPClassifier_7HL_L
Test - Acc: 0.9145, Precision: 0.9232, Recall: 0.9145, F1: 0.9173 | MLPClassifier_1HL_T
Test - Acc: 0.9585, Precision: 0.9617, Recall: 0.9585, F1: 0.9592 | MLPClassifier_2HL_T
Test - Acc: 0.9585, Precision: 0.9618, Recall: 0.9585, F1: 0.9593 | MLPClassifier_3HL_T
Test - Acc: 0.9145, Precision: 0.9244, Recall: 0.9145, F1: 0.9169 | MLPClassifier_4HL_T
Test - Acc: 0.9508, Precision: 0.9543, Recall: 0.9508, F1: 0.9517 | MLPClassifier_5HL_T
Test - Acc: 0.9430, Precision: 0.9457, Recall: 0.9430, F1: 0.9439 | MLPClassifier_6HL_T
Test - Acc: 0.9145, Precision: 0.9291, Recall: 0.9145, F1: 0.9183 | MLPClassifier_7HL_T
```

**Figure 5:** The results of training the three-layer neural network classifier (Figure 2 - architecture) and calculating quality metrics for the test dataset

Figure 5 shows the results of the experiment for the test datasets. The table presents the model names: MLPClassifier_2HL_T, SVC, MLPClassifier_2HL_R, MLPClassifier_2HL_R, where 2HL is the number of hidden layers (two in this case), R is the Relu activation function, L is the Logistic activation function, T is the Tanh activation function, and others, including accuracy, precision, recall, and F1-score for each model [22].

## 7. Discussions

As seen from the results and experiment, the best results were obtained by the MLPClassifier_2HL_T model, which achieved an accuracy of 95.85%, precision of 96.17%, recall of 95.85%, and F1-score of 95.92% on testing. Compared to other models, the MLPClassifier_2HL_T model demonstrated high training and testing accuracy. This is because the model has two hidden layers, allowing it to better represent complex dependencies between the data [23].

The performance of the models suggests that a neural network with two hidden layers and the tanh activation function is well-suited for the classification task. This finding is consistent with the notion that deeper networks can capture more complex patterns in the data.

The SHAP (SHapley Additive exPlanations) values were used to determine the importance of input features in the classification process [24]. The analysis revealed that the speed and launch angle had the most significant impact on the classification decision.

The study's results were compared with previous research on projectile classification using different methodologies. The comparison showed that the MLPClassifier_2HL_T model outperformed other models used in previous studies [25], demonstrating the effectiveness of the chosen approach.

The high performance of the MLPClassifier_2HL_T model suggests its potential application in real-world scenarios, such as military operations or security systems, where the classification of projectiles is essential for decision-making.

Overall, the study demonstrates the effectiveness of using neural networks for projectile classification and highlights avenues for further research to enhance the model's performance and applicability.

## 8. Conclusions

Developed an optimized three-layer network with two hidden layers, consisting of 33 neurons in the first hidden layer and 8 neurons in the second hidden layer, using Tanh activation functions, and an output layer with 6 neurons using Softmax activation for classifying projectiles (A - (anti-personnel), A/M - (anti-personnel high-explosive), A/P - (anti-personnel high-explosive anti-tank), A/PC - (high-explosive), M - (high-explosive anti-tank)) with an accuracy of 95.85%.

Experimentally determined that the MLP classifier (95.85% accuracy) with two hidden layers, 33 neurons in the first hidden layer, and 8 neurons in the second hidden layer using Tanh activation functions, and an output layer with 6 neurons using Softmax activation, achieved the highest accuracy.

Found that increasing the number of hidden layers, starting from 3, decreases the accuracy of projectile classification.

Demonstrated that the most significant input features influencing the accuracy of projectile classification are the projectile's speed, initial height, and launch angle.

## References


[1] Mosavi, Mohammad Reza, et al. "Multi-layer perceptron neural network utilizing adaptive best-mass gravitational search algorithm to classify sonar dataset." Archives of Acoustics 44.1 (2019): 137-151.

[2] Wang, Shouxiang, and Haiwen Chen. "A novel deep learning method for the classification of power quality disturbances using deep convolutional neural network." Applied energy 235 (2019): 1126-1140

[3] Lucie-Smith, Luisa, Hiranya V. Peiris, and Andrew Pontzen. "An interpretable machine-learning framework for dark matter halo formation." Monthly Notices of the Royal Astronomical Society 490.1 (2019): 331-342

[4] Hoffmann, Luiz Felipe Simões, Francisco Carlos Parquet Bizarria, and José Walter Parquet Bizarria. "Detection of liner surface defects in solid rocket motors using multilayer perceptron neural networks." Polymer Testing 88 (2020): 106559.

[5] State Enterprise Specialized Foreign Trade Firm Progress Catalogue. Radar, Radio Communication, and Air Defence Systems, 2022. URL: http://progress.gov.ua/wp-content/uploads/2020/08/radar-radio-communication-and-air-defence-systems.pdf

[6] Yakimchik, A. I. "On the transformation of the coordinates of points from the SK-42 system to WGS-84." Geofizicheskiy Zhurnal 41.5 (2019): 165-189

[7] Cheli, Federico, et al. "Design and testing of an innovative measurement device for tyre–road contact forces." Mechanical Systems and Signal Processing 25.6 (2011): 1956-1972

[8] Yardi, Deny Ari, et al. "Model Teoritis Penyesuaian Kurikulum Bagi Anak Berkebutuhan Khusus." CENDEKIA: Jurnal Ilmu Sosial, Bahasa dan Pendidikan 4.1 (2024): 170-181

[9] Musayev, Ilgar, and Magsad Gojamanov. "Geodetic Errors Arising from the Differences Between Sk-42 and Wgs-84 Coordinate Systems when Implemented in Modern Weapons Systems." Konya Journal of Engineering Sciences 9.2 (2021): 306-313

[10] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Convolutional networks." Deep learning 2016 (2016): 330-372

[11] Savalia, Shalin, and Vahid Emamian. "Cardiac arrhythmia classification by multi-layer perceptron and convolution neural networks." Bioengineering 5.2 (2018): 35.

[12] Peleshchak, Roman, et al. "Neural Network Architecture with Oscillatory Synaptic Blocks for Signal Processing of Spatially Spaced Multiband Radar Complex." IEEE lnternational Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering. Cham: Springer Nature Switzerland, 2022

[13] Gaikwad, Nikhil B., et al. "Efficient FPGA implementation of multilayer perceptron for real-time human activity classification." Ieee Access 7 (2019): 26696-26706.

[14] Jeon, Jueun, Jong Hyuk Park, and Young-Sik Jeong. "Dynamic analysis for IoT malware detection with convolution neural network model." IEEE Access 8 (2020): 96899-96911
[15] Li, Guang, et al. "Research on the natural language recognition method based on cluster analysis using neural network." Mathematical Problems in Engineering 2021 (2021): 1-13
[16] Rashid, Muhammad, et al. "A sustainable deep learning framework for object recognition using multi-layers deep features fusion and selection." Sustainability 12.12 (2020): 5037.
[17] Sharifzadeh, Foroogh, Gholamreza Akbarizadeh, and Yousef Seifi Kavian. "Ship classification in SAR images using a new hybrid CNN–MLP classifier." Journal of the Indian Society of Remote Sensing 47 (2019): 551-562.
[18] Desai, Meha, and Manan Shah. "An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (MLP) and Convolutional neural network (CNN)." Clinical eHealth 4 (2021): 1-11.
[19] Fekri-Ershad, Shervan. "Bark texture classification using improved local ternary patterns and multilayer neural network." Expert Systems with Applications 158 (2020): 113509.
[20] Wen, Junhao, et al. "Convolutional neural networks for classification of Alzheimer's disease: Overview and reproducible evaluation." Medical image analysis 63 (2020): 101694
[21] Adedeji, Olugboja, and Zenghui Wang. "Intelligent waste classification system using deep learning convolutional neural network." Procedia Manufacturing 35 (2019): 607-612
[22] Ting, Fung Fung, Yen Jun Tan, and Kok Swee Sim. "Convolutional neural network improvement for breast cancer classification." Expert Systems with Applications 120 (2019): 103-115
[23] Zhang, Ce, et al. "Joint Deep Learning for land cover and land use classification." Remote sensing of environment 221 (2019): 173-187.
[24] Radhakrishnan, Sita, Suresh G. Nair, and Johney Isaac. "Multilayer perceptron neural network model development for mechanical ventilator parameters prediction by real time system learning." Biomedical signal processing and control 71 (2022): 103170
[25] Talatian Azad, Saeed, Gholamreza Ahmadi, and Amin Rezaeipanah. "An intelligent ensemble classification method based on multi-layer perceptron neural network and evolutionary algorithms for breast cancer diagnosis." Journal of Experimental & Theoretical Artificial Intelligence 34.6 (2022): 949-969