ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

UDC 338.24:339.138 **JEL Classification M12, M14, M16**

Chaikovska Maryna
Doctor of Economic Sciences., Professor
Professor of the Department of Marketing and Business Administration
Oleksandr Shkeda
Ph. D student
Odesa I.I. Mechnikov National University
(Odesa, Ukraine)

MACHINE LEARNING ALGORITHM FOR AN ARTIFICIAL NEURAL NETWORK FOR BUILDING A MODEL OF MANAGERIAL DECISION-MAKING WHEN DEVELOPING A MARKETING STRATEGY

The article describes the application of the machine learning method, namely the backpropagation algorithm, in order to optimize managerial decision making when developing a marketing strategy. A qualitative analysis of data processing has been carried out, which proves the relevance of using the backpropagation method in marketing interpretation. Using the example of a task with the choice of hashtags for social media, a five-step model has been built step by step, which, after passing through many iterations of machine learning algorithms, could automate the solution of problems similar to the original one. The step-by-step process of machine learning has been described in terms of logic, mathematical functioning and programming. KPIs to assess the accuracy of the task by the model have been highlighted. An example of comparing the base KPIs of two models to select a more accurate one has been given.
Keywords: marketing management, AI, ANN, IT, backpropagation algorithm, informational model, IT development.

DOI: 10.15276/mdt.7.2.2023.10

Statement of the problem in general form and it’s connection with important scientific or practical tasks. Modern marketing is inseparable from working with data. Data is a fundamental element in marketing decision-making. It enables brands to gain insight into customer needs and preferences, as well as analyze market trends and evaluate the effectiveness of marketing campaigns. Based on the data obtained, the most effective ways of promoting the product could be selected, as well as determining the most appropriate elements of the marketing strategy (choosing marketing and advertising channels, setting optimal prices for products, creating more exact and convincing messages, etc.). In general, the use of data in the development of a marketing strategy allows a brand to make more accurate and confident decisions, as well as to use resources more efficient and to improve marketing campaign results.

Due to the open access to the functionality of the Generative Pre-trained Transformer (GPT) 3.5 and the partially open access to its algorithms, a marketing buzz, that has formed around the neural network, comparable to the informational boom of a truly pop cultural phenomenon. The effect of neural network demand has scaled both to later versions of GPT (including those that are only at the design stage) and to other neural networks, especially text ones, which have appeared and appear one after another since the creation of computer programming and machine learning processes. With the advent of the GPT phenomenon, the abbreviation AI (Artificial intelligence), although not 100% appropriate, has gained logical justification for its existence. Demonstrated leaps in development, based on big data and machine learning methods, in practice confirmed their value and expediency. With empirical evidence that ANN (Artificial Neural Network) learning algorithms are effective, it makes sense to think about how these algorithms could be applied in other areas.

Analysis of the latest research and publications, which initiated the solution of this problem on which the author relies. The hypothesis of the relevance of using machine learning in marketing management and marketing strategies development is due to the essence of modern marketing. The processes of informatization and digitalization are still unstoppable, and marketing, as a field of knowledge, has been qualitatively integrated into the concept of Digital Humanities.

Digital humanities (DH) is an area of scholarly activity at the intersection of computing or digital technologies and the disciplines of the humanities. It includes the systematic use of digital resources in the humanities, as well as the analysis of their application [1,2].

There are parallels in neural network algorithms and marketing process algorithms. A common task for both algorithms is to determine the most effective ways to achieve goals. There are many informational models, the essence of which is to optimize marketing processes (e.g. spreading activation methods or the model for brand influence) [3,4].

An ANN algorithm usually consists of several stages: processing input data, analyzing its features, training and optimizing the model, and outputting the result. The most common ANN training method is backpropagation [5,6]. Similarly, a marketing communications algorithm may include the following steps: analysis of the target audience and its preferences, content creation, distribution of content on relevant platforms, analysis of the results and optimization of the strategy depending on them.

Both algorithms can also use machine learning and artificial intelligence techniques to optimize results. For example, a neural network could be used to determine the most effective content that will be most interesting to the target audience, and a marketing communications algorithm can use A-, B-testing methods to determine which content is most effective in achieving goals.

Thus, both algorithms have common principles and tools used to achieve their goals and can complement each other to achieve the best results.

Highlighting the previously unresolved parts of the general problem to which the article is devoted. There are many informational models, the essence of which is to optimize marketing processes (e.g. spreading activation methods or the model for brand influence) [7,8]. However, marketing management and managerial decision-making regarding the elements of marketing strategies remain the least automated actions on the part of management. The development of neural networks and machine learning in recent years has shown a qualitative leap and has proved the possibility of automation and adequate performance of tasks that are based on data analysis. This is the reason to use ANN algorithms for marketing actions.

Formulation of the purpose of the article. The aim of the article is to transfer the algorithms of learning methods for neural networks to informational models used for marketing purposes.

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

Statement of the main material of the research with full justification of the scientific results obtained. An artificial neural network is a machine learning model that consists of several layers, each of which contains several artificial neurons. Each neuron receives input data, processes it, and passes the results to the next layer. The neural network is trained on a large number of examples to determine the optimal weights of connections between neurons that provide the best performance in the task that the network solves.

There are many methods of machine learning, they can be classified by the type of neural network architecture, by the type of learning, by the type of tasks being solved, etc. [9]. Each of the methods has its own advantages and disadvantages, and the choice of method depends on the specific task and available data. Point extrapolation of specific methods may be applicable to specific marketing processes. The most versatile ANN training method is the backpropagation method.

Backpropagation is a neural network training algorithm that is used to calculate the error gradient of a model over network parameters. It allows to optimize the parameters of the neural network which can predict the output values with the smallest possible error.

The description of the backpropagation algorithm was proposed in 1986, although in a rough form this method had been described and used by that time for more than a decade [10]. However, since then the algorithm has undergone many improvements and variations. In 2015 an optimization method that uses adaptive second-moment estimates of gradients to automatically adjust the learning rate during backpropagation was proposed. This method proved to be effective and quickly became one of the most popular optimization algorithms for neural networks [11].

In a 2020 scientific paper this method is described as suitable for solving problems of classification, regression and clustering in marketing. However, the algorithm of the method is more universal and its marketing potential is wider.

Let's consider an example of a neural network consisting of three layers: input, hidden and output. The neurons of each layer are connected to the neurons of the next layer. Each neuron of the input layer receives feature values as input, and each neuron of the output layer outputs a model prediction value. The goal of backpropagation is to change the weights and biases of the neurons in such a way as to minimize the prediction error. In the marketing interpretation, the input layer is the initial arguments of the mathematical function of the marketing strategy (quantitative parameters of the market, the production capacity of the company, the price of the goods, etc.). The output layer is the values of the function for given variables (projected demand, profit, audience coverage and other depending on the semantic content of the function, which is the marketing task). The hidden layer is the process of adjusting the given variables to achieve the desired result.

The backpropagation algorithm consists of the following steps:

- Forward propagation: the input data is fed to the input layer of the neural network, then the values are fed to the hidden layer, and finally to the output layer. Each neuron calculates the weighted sum of its input values, applies an activation function to it, and passes the result to the next neuron.
- Error calculation: after forward propagation, the difference between predicted and expected values is calculated. This difference is called loss.
- Backpropagation: starting at the output layer, the error is propagated back to the hidden layer and then to the input layer. Each neuron calculates its contribution to the error and passes this information back along the connections.

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

- Gradient calculation: after backpropagation, the error gradient is calculated for the weights and biases of the neurons. The gradient shows how changing each parameter affects the model error. The second moment of the gradient is an additional indicator that allows to take into account not only the direction of change in the weights, but also how strong the change should be.
- Weights and bias update: The neural network parameters are updated in the opposite direction of the error gradient. The larger the error gradient for a particular parameter, the more it will change.

At each stage, mathematical functions are involved (calculation of the weighted sum of input values, activation function, calculation of the difference between actual and predicted values, etc.). Just as for different neural networks, depending on their functionality (writing texts, creating illustrations, searching for objects, etc.), different functions are prescribed, so for different marketing tasks, different functions and variables will be used, on which the result depends. The universality of the back propagation method lies in the estimation methodology. KPIs for this method are:

1. Mean absolute error is the sum of absolute errors (modules of the difference between the predicted and observed effect) divided by the sample size (number of parameters):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y`_i|$$

2. Mean squared error is the sum of the squares of the difference between the predicted and observed effect) divided by the sample size (number of parameters):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y`_i)^2$$

In contrast to the mean absolute error, the mean square error can be applied in cases where the significance of the errors is not equal. Squaring allows to place accents - to visually increase the differences between higher and lower indicators.

3. Standard deviation is a measure of the spread of data relative to their average value, is calculated as the square root of the mean squared error:

$$RMSE = \sqrt{MSE}$$

It measures how far the values in a dataset diverge from the mean.

The value of MAE should tend to zero, which would indicate the accuracy of predicting the desired values of the function. The RMSE value is used to compare the quality of the models because it represents the deviation in the same units as the original data, but retains a higher accuracy than the mean absolute error due to squaring the MSE.

Let’s consider an example with the results of two models (table 1).

Table 1. The results of Model 1 (M1) and Model 2 (M2)

Actual values	5	8	10	12	15
Predicted values of model M1	4	10	9	14	13
Predicted values of model M2	6	9	9	12	9

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

From the example given in the table, the backpropagation KPIs for models M1 and M2 could be compared:

MAE (M1) = (|4-5| + |10-8| + |9-10| + |14-12| + |13-15|) / 5 = 1.6

MSE (M1) = $((4-5)^2 + (10-8)^2 + (9-10)^2 + (14-12)^2 + (13-15)^2)$ / 5 = 2.8

RMSE (M1) = sqrt 2.8 = 1.7

MAE (M2) = (|6-5| + |9-8| + |9-10| + |12-12| + |9-15|) / 5 = 1.8

MSE (M2) = $((6-5)^2 + (9-8)^2 + (9-10)^2 + (12-12)^2 + (9-15)^2)$ / 5 = 7.8

RMSE (M2) = sqrt 7.8 = 2.8

All three indicators of the M1 model turned out to be lower than the corresponding indicators of the M2 model. Thus, it becomes obvious that the M1 model is more accurate in the predicted values.

The backpropagation method, like other machine learning methods, can be applied in different areas, including for marketing purposes. Using the described algorithm and methodology for estimating the accuracy of forecasting, it remains only to identify logical quantitative relationships that can be converted into a mathematical function with the sought-for (desired) result and given arguments. Thus, it is a working model for planning a marketing strategy.

Let's consider the backpropagation algorithm in detail.

Step 1 is direct propagation. The neural network is given a set of input data, they pass through layers of neurons. Each neuron adds up the weighted input values and applies an activation function to this sum to get the output value.

The data is fed to the input layer of the neural network as a vector, where each element of the vector represents one of the input features or variables that the model should use to make decisions. Depending on the type of task that the neural network performs, there may be several ways to feed data to the input layer (if the neural network performs tasks with an image, then it is converted into a pixel vector; if the neural network performs tasks with text, then text vectorization methods are used to transform it into a numeric vector, etc.). As data passes through the input layer, each neuron in that layer receives the value of each element of the input data vector. Each neuron then performs calculations that take into account the input values as well as weights and biases that determine how the input data should be processed by the neuron. The result of the neuron's calculation is passed to the next layer, and so on, until the data reaches the output layer.

A vector is an ordered collection of numbers that can be represented as a one-dimensional array. Each number in a vector is called an element of the vector, and is usually denoted by an index that corresponds to the position of the element in the vector. The element of the vector can be a single feature or characteristic that the model uses to make decisions.

Considering the simplest neural network, where there are only three layers: input, hidden and output, the neurons calculate the weighted sum of their input values, apply the activation function on the hidden layer to it and pass the result to the next neuron at the output of the hidden layer. Each neuron in the hidden layer takes a vector of values as input data. They are the output values of the neurons in the input layer, multiplies each value by the corresponding weight, and then sums all these products. The resulting scalar product is passed through an activation function to get the neuron's output value. Thus, each neuron in the hidden layer computes a linear combination of input values and weights, and then applies a

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

non-linear activation function to produce an output value. The values on the hidden layer are summed up and passed to the next output layer, where the same process of calculating the output value for each neuron on this layer takes place. Thus, each neuron in the output layer computes a linear combination of input values and weights, and then applies an activation function to obtain the output value of the model.

Each connection between neurons has its own weight, which determines the importance of this connection in calculating the output value. At the beginning of training, the connection weights between neurons are randomly initialized, and then during the training process they are gradually adjusted in such a way as to minimize the model error. In error backpropagation, the connection weights between neurons are adjusted on the basis of the error gradient. Gradient descent, or variants such as stochastic gradient descent, are commonly used [12]. Thus, the connection weights are adjusted during the training process so that the model can properly process the input data and produce the correct answer.

The choice of activation function depends on the task that the neural network is solving. Each activation function has its own unique properties that can be useful in certain situations.

Extrapolating the algorithm to marketing tasks, let's develop a model for choosing a more relevant hashtag for social media, given the hashtag virality data in the used social network. To train the model, there are two hashtags #tag1 and #tag2. The first option would be the correct choice.

The input data is the hashtags #tag1 and #tag2. Let's encode them as vectors:
x = [0.5, 0.8]

Let's create a simple model with one hidden layer that has two neurons, and an output layer with one neuron that is responsible for choosing the hashtag:

```
model = Sequential()
model.add(Dense(2, input_dim=2, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))
```

The activation function is the sigmoid function:

$$sigmoid(x) = 1 / (1 + exp(-x))$$

The sigmoid function is not a universal activation function, but it is suitable for tasks related to mathematical modeling based on input data.

The activation function in neural networks is needed to introduce nonlinearity into the data processing procedure. Without an activation function, a neural network would simply be a linear model, which would limit its ability to approximate complex non-linear relationships in data. The activation function is applied to a linear combination of input values and neuron weights, which is computed in the direct propagation step. It determines the output value of the neuron and passes it on to the next layer.

Next, let's compile the model:

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

For the compilation process (preparing the model for training), let's the use the function of the Keras library. The compilation includes the definition of the loss function, the optimizer, and evaluation metrics, that are used during training.

The loss function is a function that measures the discrepancy between the actual values and the results of the model. *loss='binary_crossentropy'* is a parameter that defines the loss function that the model minimizes during training. *binary_crossentropy* is used when solving a binary classification problem where the answer can be one of two classes.

Mathematically, the loss function is expressed as

 ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

$$L(y, f(x))$$

where y – the actual values,
f(x) – the predicted values the model produces for the input x.

To solve the binary classification task, the "binary crossentropy" loss function is used, which can be defined as:

$$L(y, f(x)) = -(y * log(f(x)) + (1 - y) * log(1 - f(x)))$$

where y – takes the value 0 or 1,
f(x) – the result of the model (it is also in the range from 0 to 1),
L – the value of the loss function.

The goal of the model is to minimize the value of the loss function L on the training data, which means that the predicted values of f(x) should be as close as possible to the actual values of y.

Mathematically, the Adam optimization method uses the following formulas to update the weights (parameters) of the model at each training iteration:

- calculation of estimates of the first and second moments of the gradient:

$$m_t = beta1 * m_{\{t-1\}} + (1 - beta1) * g_t v_t = beta2 * v\_\{t - 1\} + (1 - beta2) * g\_t\text{^}2$$

where: g_t – the gradient of the loss function over the model parameters at the current iteration;
beta1 and beta2 – method hyperparameters that determine the decay rate of the first and second moments, respectively (usually set at 0.9 and 0.999);

- calculation of corrected estimates of the first and second moments:

$$m\_t\_hat = m\_t \,/\, (1 - beta1\text{^}t)\, v\_t\_hat = v\_t \,/\, (1 - beta2\text{^}t)$$

where t – the number of the current iteration;

- updating model weights:

$$w\_t = w\_\{t - 1\} - lr * m\_t\_hat \,/\, (sqrt(v\_t\_hat) + epsilon)$$

where lr – the learning rate;
epsilon – a small number added for stability (usually set at e^{-8}).

The goal of the Adam optimization method is to minimize the model's loss function by updating its parameters (weights) in the opposite direction of the loss function gradient using an adaptive learning rate at each iteration. This allows the method to quickly converge to the optimal solution and avoid getting stuck in local minima of the loss function.

The accuracy metric is a measure of how accurately the model performs the classification task, that is, how often the model correctly classifies examples. It is the ratio of the number of correctly classified examples to the total number of examples in the dataset. Mathematically, the accuracy metric can be defined as:

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

$$accuracy = (TP + TN) / (TP + TN + FP + FN)$$

where: TP (true positive) – the number of truly positive examples (the model correctly predicted that the example belongs to the positive class);
TN (true negative) – the number of true negative examples (the model correctly predicted that the example belongs to the negative class);
FP (false positive) – the number of false positive examples (the model incorrectly predicted that the example belongs to the positive class);
FN (false negative) – the number of false negative examples (the model incorrectly predicted that the example belongs to the negative class).

Thus, the accuracy metric measures the proportion of correctly classified examples relative to the total number of examples in the dataset.

For direct distribution, use the command:
y_pred = model.predict(x)

This command does not have a strict mathematical interpretation. In fact, it returns the result of the model for the input data with the current model parameter.

The actions that the described model performs are given below.

After the hashtags (input data) have been converted into vectors, the neuron activations are calculated on the hidden layer. To do this, the input data is multiplied by the weights that connect the input and hidden layers. There are two neurons on the hidden layer, and the connection weights look like this:
w1 = [0.4, 0.2]
w2 = [0.3, 0.9]

To get the neuron activations on the hidden layer, the input data is multiplied by each of the weight vectors:
*a1 = x[0]*w1[0] + x[1]*w1[1] = 0.5*0.4 + 0.8*0.2 = 0.36*
*a2 = x[0]*w2[0] + x[1]*w2[1] = 0.5*0.3 + 0.8*0.9 = 0.87*

Next, the activation on the output layer is calculated. To do this, the activations of neurons in the hidden layer are multiplied by the weights that connect the hidden and output layers. There is one neuron on the output layer, and the weights for the connection look like this:
w3 = [0.1, 0.3]

To get the activation on the output layer, it is needed to multiply the neuron activations on the hidden layer by the weight vector of the neuron on the output layer:
*y = a1*w3[0] + a2*w3[1] = 0.36*0.1 + 0.87*0.3 = 0.297*

This is the result on the output layer.

Step 2 is error calculation. After the data has passed through the network, the predicted values are compared with the expected values to find the difference between them. This difference is called the error or loss function (described above).

For this example with hashtags it is:
y_true = [1]
where y_true is the correct answer (hashtag #tag1).
loss = model.evaluate(x, y_true)

This command also does not perform mathematical calculations. It returns the values of the loss function and the metric (described above).

The calculations themselves take place as follows:

$$delta\_y = (y - y\_true) * f'(z\_y)$$

where delta_y – the output layer error;
y – the output value of the neuron on the output layer;

f'(z_y) – the derivative of the activation function at the output layer with respect to the input;
z_y – the weighted sum on the output layer.

Desired value is:

*delta_y = (0.297 - 1) * 0.297 * (1 - 0.297) = -0.147*

The negative value is due to the use of weights and bias. Bias in neural networks is a constant that is added to the sum of the weighted inputs in each layer neuron before being passed through the activation function. The model used an offset on each layer. For each layer, the offset is calculated using a formula similar to the activation calculation, but without multiplying by the input values.

Step 3 is backpropagation. Now the error propagates back through the neural network, starting at the output layer and moving back to the input layer. Each neuron calculates its contribution to the total error and passes this information back along the connections.

After calculating the error, the weights are adjusted:

model.fit(x, y_true, epochs=1000, verbose=0)

where epochs – the number of epochs (iterations) for model training;
verbose – a parameter that controls the output of information during model training (if verbose=0, then no information will be output during training).

Step 4 is calculating the gradient. After backpropagation, the error gradient is calculated the weights and biases of the neurons. The gradient shows how changing each parameter (weight and bias) affects the model error. For this, the derivative of the error with respect to each parameter of the weights is used.

Step 5 is update weights and biases. After the gradient is computed, the weights and biases of the neurons are updated in the opposite direction of the gradient. This allows the model to improve accuracy and find the local minimum of the loss function.

After adjusting the weights, the iteration is repeated using direct propagation. The result of the model is improved until the accuracy of the solution to the task matches the desired value (#tag1). Obtaining the result of the model after adjusting the weights is carried out using the command:

y_pred = model.predict(x)

These five steps are repeated many times until the model reaches the required level of accuracy. Then the model could be used to predict values on new data.

Conclusions from this research and prospects for further developments in this area. Created model serves as a demonstration of the applicability of such an ANN machine learning method as an error backpropagation method for marketing purposes to build decision-making models when developing a marketing strategy. The KPIs described in the article can serve as fundamental signals for choosing a more appropriate information model for solving the tasks. Since the accuracy of data processing in the functioning of the informational model is of higher importance than, for example, usability or digital compatibility, the described KPIs are considered basic. However, the fact that they are described mathematically in the article makes it possible to adapt any model to output data with a higher accuracy factor. The hashtag problem served as a good example of using machine learning algorithms to perform marketing tasks, but it should be understood that this approach is not limited to the described functions, variables or data arrays.

ISSN 2522-9087 (Print)
ISSN 2523-434X (Online)

1. Drucker, Johanna (2021). The Digital Humanities. An Introduction to Digital Methods for Research and Scholarship. UCL: UCLA Center for Digital Humanities. 252 p.
2. Terras, Melissa (2011). Quantifying Digital Humanities. [Digital Humanities] Retrieved from https://www.ucl.ac.uk/infostudies/melissa-terras/DigitalHumanitiesInfographic.pdf (accessed 25 May 2023).
3. Shkeda, Oleksandr (2022). Convergence of spreading activation methods in the informational environment and marketing communications. *Formation of accounting and analytical support for organizational management support of subjects of the national economy*. Proceeding of the International scientific and practical conference, Odesa I.I. Mechnikov national university, pp. 126-128.
4. Chaikovska Maryna; Shkeda Oleksandr. (2022). Analysis of digital indicators of the convergence of behavioral and informational trends of society management in the framework of the societal concept of influencer marketing. *Marketing and digital technologies.* Vol. 6(2), pp. 73-85.
5. Chaikovska M., Shkeda A. (2022). Convergence of user experience and marketing communications management under reflexive digital transformations. *Science, education, culture*: Papers of international sci.-pract. conf. Vol. 1. Komrat: KSU, 2022. pp. 183-187.
6. Chaikovska M., Shkeda O. (2023). Application of artificial neural network training for management decision-making in marketing activities. *Management and marketing as a factor of business development in conditions of economic recovery*: materials of international science and practice conf. (April 18-19, 2023, Kyiv, KMA). Kyiv: Kyiv-Mohyla Academy Publishing House, pp.302-305.
7. Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron. (2016). Deep Learning. MIT Press. 781 p. Retrieved from: https://www.deeplearningbook.org/
8. Graves, Alex; Wayne, Greg; Danihelka, Ivo (2014). Neural Turing Machines. *Cornell University archive*. Retrieved from: https://arxiv.org/abs/1410.5401 (accessed 21 March 2023).
9. Bishop, Christopher. (2006). Pattern Recognition and Machine Learning. NY: Springer.730 p.
10. Rumelhart, David; Hinton, Geoffrey; Williams, Ronald. (1986). Learning representations by back-propagating errors. *Nature*, no. 323, pp. 533–536.
11. Kingma, Diederik; Ba, Jimmy. (2015). [Cornell University archive]. Retrieved from: https://arxiv.org/abs/1410.5401 (date of application: 04/15/2023) (accessed 15 April 2023).
12. Robbins, Herbert; Monro, Sutton. (2015). A Stochastic Approximation Method. The Annals of Mathematical Statistics, 22(3), pp. 400 – 407.

Чайковська М.П., *д.е.н, професор, професор кафедри маркетингу та бізнес-адміністрування, Одеський національний університет імені І.І. Мечникова (Одеса, Україна).*
Шкеда О.О., *аспірант, Одеський національний університет імені І.І. Мечникова (Одеса, Україна).*

Алгоритм машинного навчання штучної нейронної мережі для побудови моделі прийняття управлінських рішень при розробці маркетингової стратегії.

У статті описано застосування методу машинного навчання, а саме алгоритму зворотного поширення помилки, з метою оптимізації прийняття управлінських рішень при розробці маркетингової стратегії. Проведено якісний аналіз процесів обробки даних, що підтверджує доцільність використання методу зворотного поширення в маркетинговій інтерпретації. На прикладі завдання з вибором хештегів для соціальних медіа покроково побудовано п'ятиступінчасту модель, яка, пройшовши багато ітерацій алгоритмів машинного навчання, може автоматизувати вирішення задач, подібних до вказаної. Покроковий процес машинного навчання було описано з точки зору логіки, математичного функціонування та програмування. Виділено KPI для оцінки точності виконання завдання моделлю. Наведено приклад порівняння базових KPI двох моделей для вибору більш точної.

Ключові слова: управління маркетингом, ШІ, ШНМ, ІТ, метод зворотного поширення помилки, інформаційна модель, ІТ девелопмент.

Надійшло до редакції: *12 квітня 2023*