МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**ИНТЕЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

ОТЧЁТ

По лабораторной работе № \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил:

Студент группы ИИ-22

Копанчук Евгений Романович

Проверил\_\_:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Брест – 2023

**Ход работы**

*Код программы*

class Perceptron {  
  
 constructor({n\_inputs, treeshold}) {  
 this.weights = Array.from({length: n\_inputs}, () => 0);  
 this.treeshold = treeshold;  
 }  
  
 fit({X\_train, y\_train, epoches=100, aim\_loss=10e-3, learning\_rate=10e-3}) {  
 if (X\_train.length !== y\_train.length) {  
 return new Error('Error: X\_train size must be the same as y\_train  
size');  
 }  
 if (X\_train.length !== this.weights.length) {  
 return new Error('Error: X\_train size must be the same as n\_inputs');  
 }  
 const loss\_values = {};  
 for (let epoch = 0; epoch < epoches; epoch++) {  
 for (let j = 0; j < X\_train.length; j++) {  
 y\_pred = this.predict(X\_train[j]);  
 loss = y\_train[i] - y\_pred;  
 loss\_values[`epoch: ${epoch}`] = loss;  
 console.log(`epoch: ${epoch}\tloss: ${loss}`);  
 if (loss < aim\_loss) {  
 break;  
 }  
 for (let k = 0; k < this.weights.length; k++) {  
 this.weights[i] += learning\_rate \* loss \* X\_train[j][k];  
 }  
 }  
 }  
  
 return loss\_values;  
 }  
  
 test({X\_test, y\_test}) {  
 if (X\_test.length !== y\_test.length) {  
 return new Error('Error: X\_test size must be the same as y\_test  
size');  
 }  
 let accuracy = 0;  
 for (let i = 0; i < X\_test.length; i++) {  
 accuracy += (this.predict(X\_test[i]) === y\_test[i]) ? 1 /  
X\_test.length : 0;  
 }  
  
 return accuracy;  
 }  
  
 predict(X) {  
 let sum = 0;  
 for (let i = 0; i < this.weights.length; i++) {  
 sum += this.weights[i] \* X[i];  
 }  
  
 return (sum > this.treeshold) ? 1 : 0;  
 }  
  
 get\_weights() {  
 return this.weights;  
 }  
  
 load\_weightss(weights) {  
 this.weights = weights;  
 }  
}  
  
const X = [  
 [0, 0],  
 [0, 1],  
 [1, 0],  
 [1, 1]  
]  
  
const y = [0, 0, 0, 1];  
  
const model = new Perceptron({n\_inputs: 2, treeshold: 0.5});  
  
model.fit({X\_train: X, y\_train: y});  
const accuracy = model.test(X, y);  
console.log(accuracy);