Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра интеллектуальных информационных технологий

Лабораторная работа №2

По дисциплине «Модели решения задач в ИС»

Тема: «Адаптивный шаг обучения в однослойном линейном персептроне»

Выполнил:

Студент 3 курса

Факультета ЭИС

Группы ИИ-26

Рулько М.А.

Проверил:

Андренко К.В.

Брест 2026

Цель работы: изучить алгоритм оптимизации градиентного спуска с использованием адаптивного шага обучения. Реализовать модифицированный персептрон, в котором параметр скорости обучения t вычисляется на основе минимизации квадратичной формы ошибки для каждой итерации. Сравнить скорость сходимости с классическим алгоритмом из Лабораторной работы №1. Вариант сохраняется.

Задачи лабораторной работы:

1. Модифицировать алгоритм последовательного обучения (из Лаб №1) таким образом, чтобы на каждой итерации t значение α вычислялось автоматически на основе текущего входного вектора x по формул (см раздел 2.9).
2. Применить вычисленный α(t) для обновления весов ij и порогов Tj согласно дельта-правилу.
3. Используя данные своего варианта, провести два эксперимента:
   - Обучение с фиксированным шагом (например, α=0.1 или α=1p).
   - Обучение с адаптивным шагом по Теореме 2.1 (формула 2.36).

   Критерий остановки в обоих случаях – достижение заданной суммарной ошибки Es≤Ee.
4. Построить графики обучения Es(p), где p – номер эпохи, для обоих экспериментов на одних осях координат.
5. Выполнить графическую визуализацию разделяющей линии для адаптивного метода.
6. Реализовать режим функционирования сети:
   - пользователь задаёт произвольный входной вектор,
   - сеть вычисляет выходной класс,
   - соответствующая точка отображается на графике,

Написать вывод по выполненной работе. Оценить, насколько адаптивный шаг сокращает количество эпох обучения по сравнению с фиксированным. Обязательно сравнение результатов 1 и 2 лабораторных работ.

Вариант:

Ход работы

Код программы:

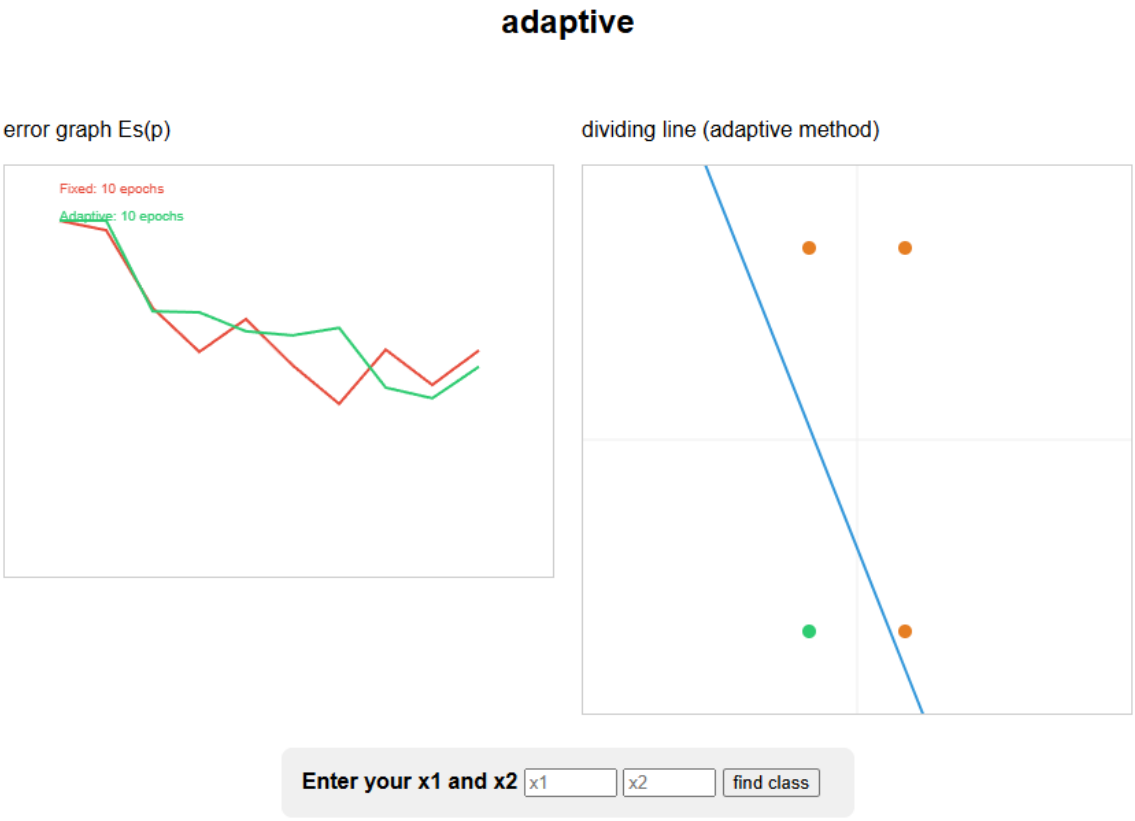
```
class Perceptron {
    constructor() {
        this.w1 = 0;
        this.w2 = 0;
        this.bias = 0;
        this.epsilon = 0.001;
    }

    getSum(x1, x2) {
        return x1 * this.w1 + x2 * this.w2 + this.bias;
    }

    predict(x1, x2) {
        return this.getSum(x1, x2) >= 0 ? 1 : -1;
    }

    train(dataset, mode = 'fixed', alphaFixed = 0.01) {
        let epochs = 0;
        const history = [];
        const maxEpochs = 10;

        while (epochs < maxEpochs) {
            let mseSum = 0;
            const shuffled = [...dataset].sort(() => Math.random() - 0.5);

            shuffled.forEach(point => {
                const out = this.getSum(point.x1, point.x2);
                const error = point.label - out;

                let alpha = alphaFixed;
                if (mode === 'adaptive') {
                    const norm = 1 + Math.pow(point.x1, 2) + Math.pow(point.x2, 2);
                    alpha = 0.5 * (1 / norm);
                }

                this.w1 += alpha * error * point.x1;
                this.w2 += alpha * error * point.x2;
                this.bias += alpha * error;

                mseSum += Math.pow(error, 2);
            });

            const avgMse = mseSum / dataset.length;
            history.push(avgMse);
            epochs++;

            if (avgMse <= this.epsilon) break;
        }
        return { history, epochs };
    }
}
```

Класс Персептрона

```
const dataset = [
    { x1: 1,  x2: 4,  label: -1 },
    { x1: -1, x2: 4,  label: -1 },
    { x1: 1,  x2: -4, label: -1 },
    { x1: -1, x2: -4, label: 1 }
];

const viz = new Visualizer('mseChart', 'decisionChart');

const netFixed = new Perceptron();
const resFixed = netFixed.train(dataset, 'fixed', 0.03);

const netAdapt = new Perceptron();
const resAdapt = netAdapt.train(dataset, 'adaptive');

viz.drawLearningCurves(resFixed.history, resAdapt.history);
viz.drawBoundary(netAdapt, dataset);
```

Инициализация скрипта

ГРАФИК



Сравнение результатов лабораторных работ:

Характеристика	Л.Р.1	Л.Р.2
Тип шага обучения	Фиксированный	Адаптивный
Необходимость подбора α	Требуется	Не требуется
Скорость сходимости	Ниже	Выше
Число эпох	Больше	Меньше
Устойчивость	Зависит от α	Более устойчива

Вывод: Во второй лабораторной работе использовался адаптивный шаг обучения, который вычисляется автоматически на каждой итерации. В первой работе шаг был фиксированным и его нужно было подбирать вручную.

По сравнению с классическим алгоритмом из ЛР №1, адаптивный метод сходится быстрее или как минимум не хуже, особенно если в первой работе выбран небольшой коэффициент обучения. Кроме того, адаптивный способ удобнее, так как не требует подбора параметра α.

Таким образом, при сохранении моего варианта данных адаптивный алгоритм оказался более практичным и эффективным по скорости сходимости.