

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки

Лабораторна робота №3.3  
“Дослідження нейронних мереж. модель perceptron”

Виконав:  
студент групи ІП-84  
ІП-8408  
Засько Євгеній

Київ 2021

## Теоретичні відомості

Важливою задачею якої система реального часу має вирішувати є отримання необхідних для обчислень параметрів, її обробка та виведення результату у встановлений дедлайн. З цього постає проблема отримання водночас точних та швидких результатів. Модель Перцептрон дозволяє покроково наблизити початкові значення.

Розглянемо приклад: дано дві точки A(1,5), B(2,4), поріг спрацювання  $P = 4$ , швидкість навчання  $\delta = 0.1$ . Початкові значення ваги візьмемо нульовими  $W_1 = 0$ ,  $W_2 = 0$ . Розрахунок вихідного сигналу  $y$  виконується за наступною формулою:

$$x_1 * W_1 + x_2 * W_2 = y$$

Для кожного кроку потрібно застосувати дельта-правило, формула для розрахунку похибки:

$$\Delta = P - y$$

де  $y$  – значення на виході.

Для розрахунку ваги, використовується наступна формула:

$$W_{Nn} = W_{Nn} + \delta * x_N * g$$

## Умови завдання для варіанту бригади

Поріг спрацювання:  $P = 4$  Дано точки: A(0,6), B(1,5), C(3,3), D(2,4). Швидкості навчання:  $\delta = \{0,001; 0,01; 0,05; 0,1; 0,2; 0,3\}$  Дедлайн: часовий =  $\{0.5с; 1с; 2с; 5с\}$ , кількість ітерацій =  $\{100; 200; 500; 1000\}$  Обрати швидкість навчання та дедлайн. Налаштувати Перцептрон для даних точок. Розробити відповідний мобільний додаток і вивести отримані значення. Провести аналіз витрати часу та точності результату за різних параметрах навчання.

## Лістинг програми із заданими умовами завдання

```
package ua.kpi.comsys.lab3_2
```

```
import kotlinx.coroutines.*
```

```
import kotlin.system.measureTimeMillis
```

```
data class NeuroResult(
```

```
    val success: Boolean,
```

```

    val w0: Float = 0f,
    val w1: Float = 0f,
    val iterations: Int = 0,
    val time: Long = 0,
    val errorMsg: String? = null,
)

```

```

class NeuroAlgorithm(
    private val iterations: Float,
    private val deadline: Float,
    private val learningSpeed: Float,
    private val points: List<Pair<Int, Int>>,
    private val p: Int,
) {
    private var job: Job? = null
    private var cancelJob: Job? = null

```

```

suspend fun start(onDone: suspend (NeuroResult) -> Unit) {
    job = GlobalScope.launch {
        var w0 = 0f
        var w1 = 0f
        var curIteration = 0
        val nr: NeuroResult

        val time = measureTimeMillis {
            for (iteration in 1..iterations.toInt()) {
                yield()
                curIteration = iteration
                val pointIndex = (iteration - 1) % points.size
                val point = points[pointIndex]
                val y = w0 * point.first + w1 * point.second
            }
        }
    }
}

```

```

val res = points.mapIndexed { index, pair ->
    val tempY = w0 * pair.first + w1 * pair.second
    if (index >= points.size / 2) {
        tempY < p
    } else {
        tempY > p
    }
}

if (res.all { it }) break

val d = p - y
w0 += d * point.first * learningSpeed
w1 += d * point.second * learningSpeed

}

}

nr = if (iterations.toInt() == curIteration) {
    NeuroResult(
        success = false,
        errorMsg = "Iteration limit exceeded"
    )
} else {
    NeuroResult(
        success = true,
        w0 = w0,
        w1 = w1,
        iterations = curIteration,
        time = time
    )
}

```

```
}
```

```
cancelJob?.cancel()
```

```
onDone(nr)
```

```
}
```

```
cancelJob = GlobalScope.launch {
```

```
    delay((deadline * 1000).toLong())
```

```
    yield()
```

```
    job?.cancel()
```

```
    val nr = NeuroResult(
```

```
        success = false,
```

```
        errorMsg = "Deadline time exceeded",
```

```
    )
```

```
    onDone(nr)
```

```
}
```

```
}
```

```
}
```

## Результати виконання кожної програми



## Висновки щодо виконання лабораторної роботи

В ході виконання лабораторної роботи ознайомився з принципами роботи машинного навчання за допомогою математичної моделі Перцептрон. Дослідив вплив параметрів на час виконання та точність результату і отримав такі результати: так як точки в нас незмінні то результат роботи алгоритму залежить тільки від параметру швидкості навчання. Якщо цей параметр зробити надто великим то алгоритм буде обмежуватись кількістю ітерацій. З сучасними технологіями навіть мінімального обмеження в 500 мс не виникає