Міністерство освіти і науки України
Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЗВІТ

з лабораторної роботи №3.2 з дисципліни
«Інтелектуальні вбудовані системи»

Виконав:
ІП-83
Сергійчук Н. С.

Київ 2020

1. Завдання

Поріг спрацювання: P = 4

Дано точки: A(0,6), B(1,5), C(3,3), D(2,4).

Швидкості навчання: δ = {0,001; 0,01; 0,05; 0.1; 0.2; 0,3}

Дедлайн: часовий = {0.5c; 1c; 2c; 5c}, 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.отримані значення. Провести аналіз витрат часу на розрахунки

2. Лістинг програми

```
class Point {
final double x;
final double y;

const Point(this.x, this.y);

String toString() => "($x, $y)";
}

class Perceptron {
double w1;
double w2;

Perceptron(this.w1, this.w2);

double y(Point p) => p.x * w1 + p.y * w2;
void advance(Point a, double delta, double sigma) {
w1 = w1 + a.x * delta * sigma;
w2 = w2 + a.y * delta * sigma;
}

String toString() => "Perceptron{ w1: $w1, w2: $w2 }";
}

class Network {
double p;
double sigma;
List<Point> points;
int iters;
```

```
Network(this.iters, this.p, this.sigma, this.points);
```

```
Perceptron run() {
int i = 0;
var percep = Perceptron(0, 0);
while (iters > 0) {
var point = points[i];
var delta = p - percep.y(point);
percep.advance(point, delta, sigma);
i = (i + 1) % points.length;
iters--;
print("[$iters] : $percep");
}
return percep;
}
}
```

```
void main() {
var points = [Point(0, 6), Point(1, 5), Point(3, 3), Point(2, 4)];
var iters = 100;
var sigma = 0.1;
var p = 4.0;
var network = Network(iters, p, sigma, points);
var result = network.run();
print("ok: $result");
}
```

```
import 'package:flutter/material.dart';
import 'package:lab3_2/perceptron.dart';

void main() {
runApp(MyApp());
}
```

```
class MyApp extends StatelessWidget {
// This widget is the root of your application.
@override
Widget build(BuildContext context) {
return MaterialApp(
title: 'Flutter Demo',
theme: ThemeData(
// This is the theme of your application.
//
// Try running your application with "flutter run". You'll see the
```

```
// application has a blue toolbar. Then, without quitting the app, try
// changing the primarySwatch below to Colors.green and then invoke
// "hot reload" (press "r" in the console where you ran "flutter run",
// or simply save your changes to "hot reload" in a Flutter IDE).
// Notice that the counter didn't reset back to zero; the application
// is not restarted.
primarySwatch: Colors.blue,
),
home: MyHomePage(title: 'Flutter Demo Home Page'),
);
}
}

class MyHomePage extends StatefulWidget {
MyHomePage({Key key, this.title}) : super(key: key);

// This widget is the home page of your application. It is stateful,
meaning
// that it has a State object (defined below) that contains fields that
affect
// how it looks.

// This class is the configuration for the state. It holds the values (in
this
// case the title) provided by the parent (in this case the App widget)
and
// used by the build method of the State. Fields in a Widget subclass are
// always marked "final".

final String title;

@override
_MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
static const List<String> iterList = ["10", "20", "50", "100"];
static const List<String> sigmaList = [
"0.001",
"0.01",
"0.05",
"0.1",
"0.2",
"0.3"
];
static const List<Point> points = [
Point(0, 6),
```

```
Point(1, 5),
Point(3, 3),
Point(2, 4)
];
static const p = 4.0;
```

```
String iters = iterList[0];
String sigma = sigmaList[0];
Perceptron result = Perceptron(0, 0);
String elapsed = "";
```

```
@override
Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(
title: Text("Lab3.2"),
),
body: Center(
child: Padding(
padding:
const EdgeInsets.symmetric(horizontal: 40.0, vertical: 120.0),
child: Column(
mainAxisAlignment: MainAxisAlignment.spaceEvenly,
crossAxisAlignment: CrossAxisAlignment.start,
children: <Widget>[
Text("Points: $points"),
Text("P: $p"),
Row(
children: [
Padding(
padding: const EdgeInsets.only(right: 10.0),
child: Text("Iterations:"),
),
buildDropDown(iterList, iters, (String value) {
setState(() {
iters = value;
run();
});
}),
],
),
Row(
children: [
Padding(
padding: const EdgeInsets.only(right: 10.0),
child: Text("Sigma:"),
```

```
),
buildDropDown(sigmaList, sigma, (String value) {
setState(() {
sigma = value;
run();
});
}),
],
),
Column(
crossAxisAlignment: CrossAxisAlignment.start,
children: [
Text("Result:"),
Text("w1 = ${result.w1}"),
Text("w2 = ${result.w2}"),
],
),
Text("Elapsed: ${elapsed}ms")
],
),
),
),
);
}
```

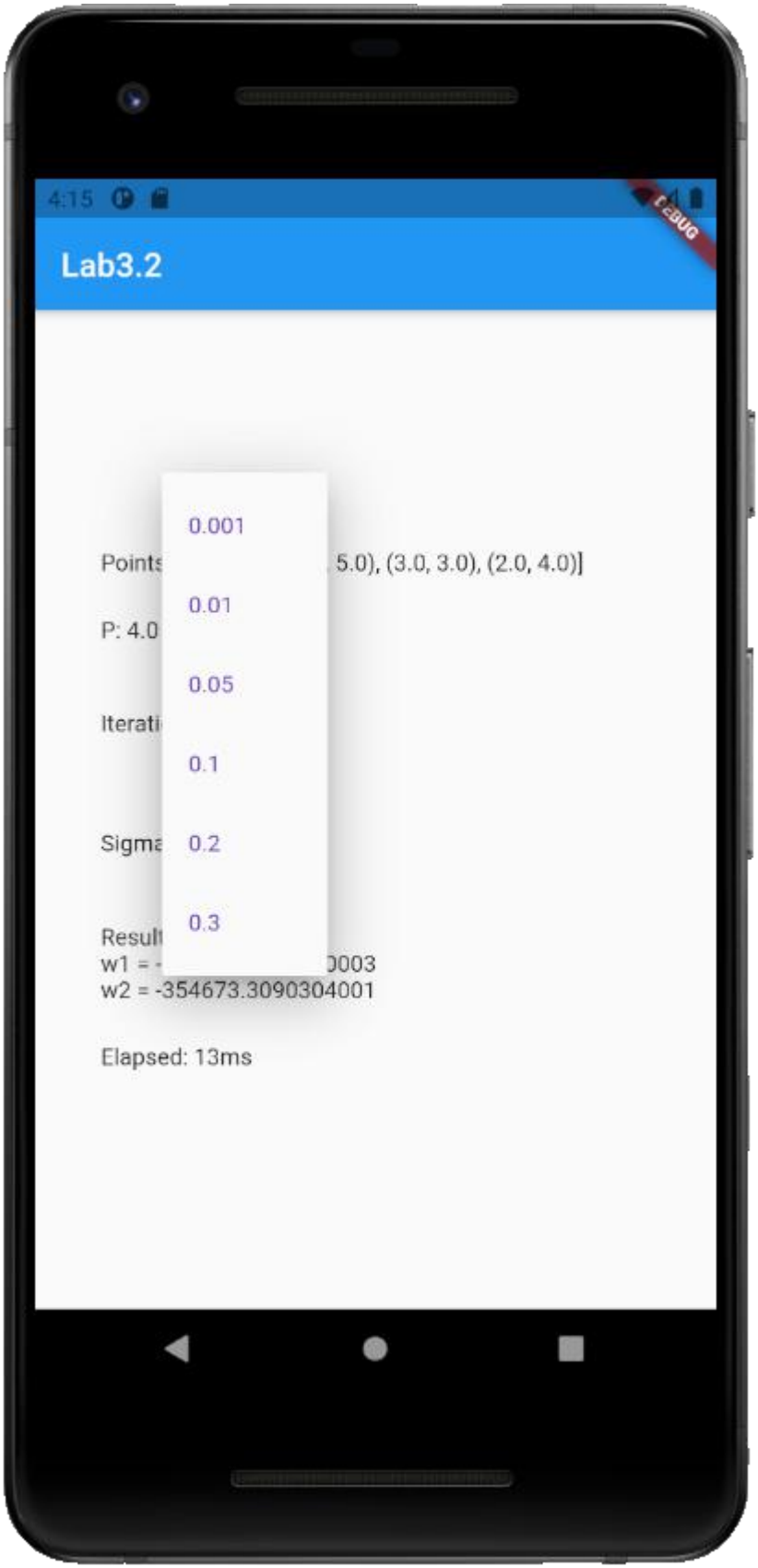
```
void run() {
var intiters = int.parse(iters);
var doublesigma = double.parse(sigma);
Stopwatch stopwatch = new Stopwatch()..start();
var network = Network(intiters, p, doublesigma, points);
result = network.run();
elapsed = "${stopwatch.elapsed.inMilliseconds}";
}
}
```

```
Widget buildDropDown(List<String> list, String value, Function callback)
{
return DropdownButton<String>(
value: value,
elevation: 16,
style: const TextStyle(color: Colors.deepPurple),
underline: Container(
height: 2,
color: Colors.deepPurpleAccent,
),
onChanged: callback,
```

```
items: list.map<DropdownMenuItem<String>>((String value) {
return DropdownMenuItem<String>(value: value, child: Text(value));
}).toList(),
);
}
```

3. Результати виконання кожної програми.


4:15
DEBUG
Lab3.2
Points: [(0.0, 6.0), (1.0, 5.0), (3.0, 3.0), (2.0, 4.0)]
P: 4.0
Iterations: 10
Sigma: 0.2
Result:
w1 = -92790.00110080003
w2 = -354673.3090304001
Elapsed: 13ms


4:15
DEBUG
Lab3.2
0.001
0.01
0.05
0.1
0.2
0.3
5.0), (3.0, 3.0), (2.0, 4.0)]
P: 4.0
w2 = -354673.3090304001
Elapsed: 13ms


4:16
DEBUG
Lab3.2
Points: [(0.0, 6.0), (1.0, 5.0), (3.0, 3.0), (2.0, 4.0)]
P: 4.0
Iterations: 10
Sigma: 0.001
Result:
w1 = 0.047337098139598355
w2 = 0.16876511515317805
Elapsed: 4ms