**Computer Assignment 2**

ธนกิจ เหล่ายาวิระ600632014

**1.รายละเอียดของทฤษฎีหรือวิธีการต่าง ๆ ที่ใช้**

1.  ;Euclidean Distant

2. ใช้ NodeJS ในการพัฒนาซอฟท์แวร์

**2.Flowchart**

No

Start

แบ่งข้อมูลตาม cross validation

คำนวณค่าระยะทาง

สร้าง Confusion matrix

ถ้าเป็นข้อมูล validate สุดท้าย

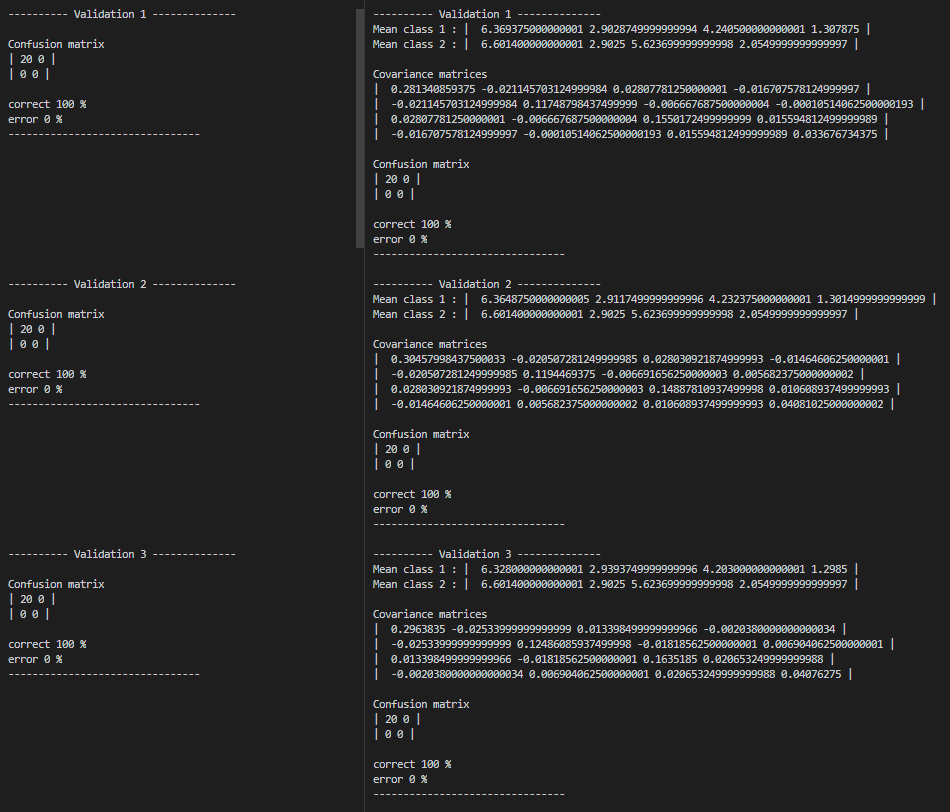
yes

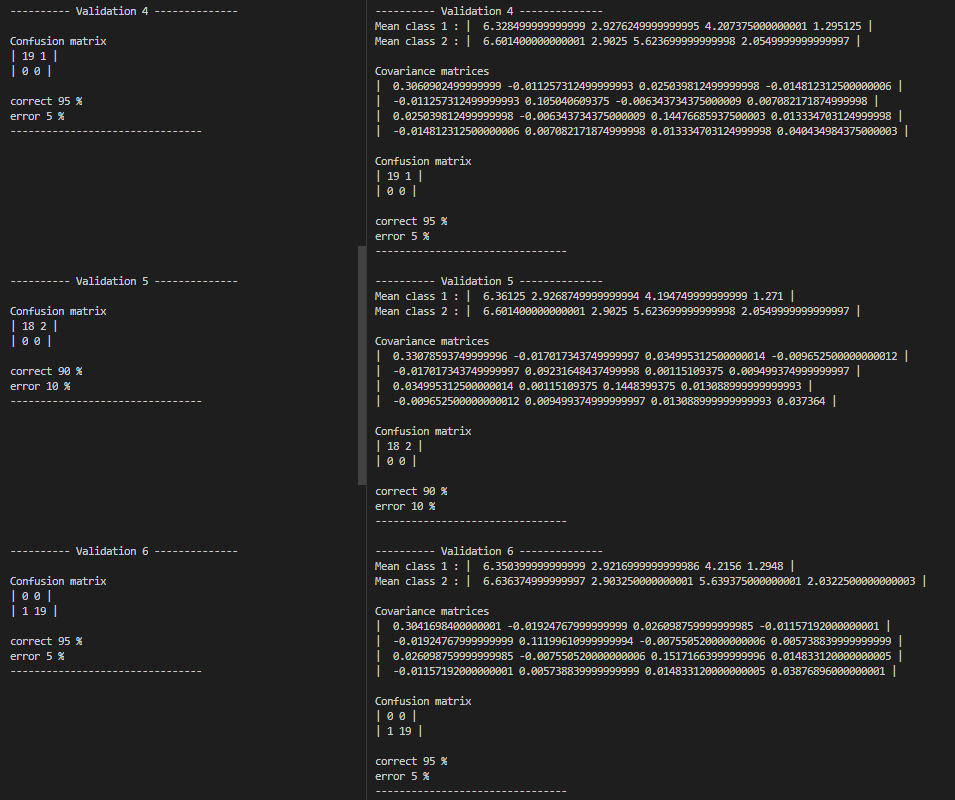
เปรียบเทียบผล

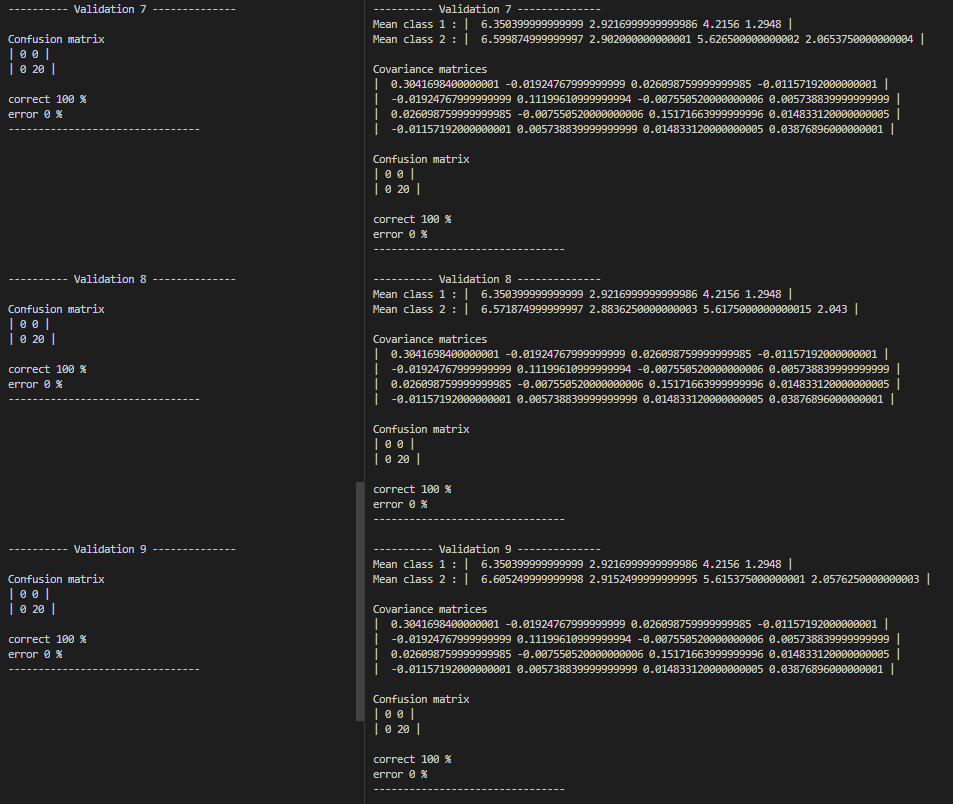
End

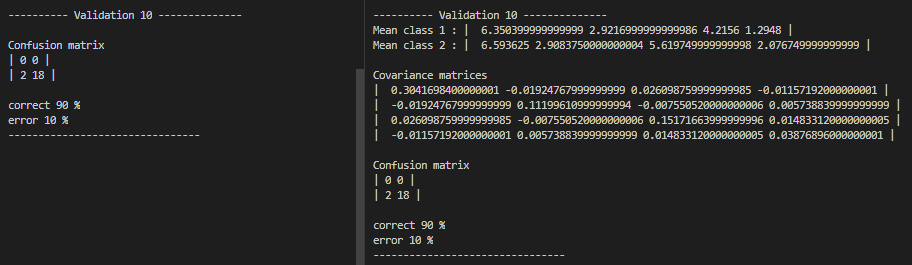
**3. ผลการทดลอง**

เมื่อทดลองกับ k = 4 กับ Bayes Classifier 4 feature









**4. สรุปผลการทดลอง** เมื่อใช้จำนวน K และ จำนวน feature เท่ากัน ผลของการเลือกคลาสจะเหมือนกัน

**ภาคผนวก**

Code: NodeJS

let fs = require('fs')

const math = require('mathjs')

let input = fs.readFileSync('TWOCLASS.dat', 'utf8')

const fNum = 4 //Change number of features here

const percentValidate = 10

const kValue = 14

const fetchData = Promise.resolve(

input.trim().split('\r\n').map(x => x.split('\t'))

)

const setUpTestData = (percentValidate, data) => {

const testDataNum = data.length / percentValidate

const round = data.length / testDataNum

let testDatas = []

for (i = 0; i < round; i++) {

testDatas.push(data.slice((i \* testDataNum), (i \* testDataNum + testDataNum)))

}

return testDatas

}

const setUpTrainData = (percentValidate, data) => {

const testDataNum = data.length / percentValidate

const round = data.length / testDataNum

let trainDatas = []

for (i = 0; i < round; i++) {

let trainData = []

for (j = 0; j < data.length; j++) {

if (j < (i \* testDataNum) || j >= (i \* testDataNum + testDataNum)) {

trainData.push(data[j])

}

}

trainDatas.push(trainData)

}

return trainDatas

}

const calDistance = (sources, destinations) => {

return destinations.map(

(destination) => {

let res = []

for (let j = 0; j < (destination.length - 1); j++) {

res.push(

Math.pow(

(parseFloat(sources[j]) - parseFloat(destination[j]))

, 2)

)

}

let resSq = Math.sqrt(res.reduce((prev, curr) => prev + curr))

res = []

res.push(resSq, destination[4])

return res

}

)

}

const main = async () => {

let sourceData = await fetchData.then((value) => {

value.shift()

return value

})

const testDatas = setUpTestData(percentValidate, sourceData)

const trainDatas = setUpTrainData(percentValidate, sourceData)

for (let i = 0; i < 10; i++) {

let a = 0, b = 0, c = 0, d = 0

for (let j = 0; j < testDatas[i].length; j++) {

let dist = await calDistance(testDatas[i][j], trainDatas[i])

const distSorteds = dist.sort((a, b) => a[0] - b[0])

const distSliceds = distSorteds.slice(0, kValue)

const classes = [0, 0]

distSliceds.forEach(distSliced => {

if (distSliced[1] === '1') { classes[0]++ }

else if (distSliced[1] === '2') { classes[1]++ }

})

let classChoose

if (classes[0] === classes[1]) { classChoose = Math.floor((Math.random() \* 2) + 1) }

else if (classes[0] > classes[1]) { classChoose = 1 }

else if (classes[0] < classes[1]) { classChoose = 2 }

if (parseInt(testDatas[i][j][4]) === 1 && classChoose === 1) { a = a + 1 }

else if (parseInt(testDatas[i][j][4]) === 1 && classChoose === 2) { b = b + 1 }

else if (parseInt(testDatas[i][j][4]) === 2 && classChoose === 1) { c = c + 1 }

else if (parseInt(testDatas[i][j][4]) === 2 && classChoose === 2) { d = d + 1 }

}

console.log(`---------- Validation ${i + 1} --------------`)

console.log(`\nConfusion matrix`)

console.log(`| ${a} ${b} |`)

console.log(`| ${c} ${d} |`)

let correct = 100 \* (a + d) / (a + b + c + d)

let error = 100 - correct

console.log(`\ncorrect ${correct} %`)

console.log(`error ${error} %`)

console.log(`--------------------------------\n\n\n\n\n\n\n\n\n`)

}

}

main()