Computer Assignment 3

CPE 261456 (Introduction to Computational Intelligence)

โดย

นายพีรณัฐ  ธารทะเลทอง

รหัสนักศึกษา 550610530

เสนอ

ผศ.ดร. ศันสนีย์  เอื้อพันธ์วิริยะกุล

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่

**ลักษณะการทำงานของระบบ**

เริ่มต้น เป็นการกำหนด จำนวนโครโมโซม , generation ในการ train และโครงสร้าง Neuron network ที่จะใช้ให้ GA จากนั้นจะทำการ initial โครโมโซม โดยการสุ่มค่าน้ำหนัก -1 ถึง 1 โดยโครโมโซมจะมีความยาวเท่ากับจำนวนน้ำหนักในโครงสร้าง Neuron network จากนั้นทำ 10% cross validation โดยในขั้นตอนการ train ในแต่ละ fold นั้น เริ่มจาก สุ่มเลือกโครโมโซม ทีละคู่เพื่อนำมา crossover โดยใช้การ crossover แบบหนึ่งจุดจากตรงกลางของโครโมโซม โดยได้ลูกจากการทำ crossover มา 90% จากทั้งหมด จากนั้นนำโครโมโซมลูกที่ได้จากการ crossover มารวมกับกลุ่มพ่อแม่และให้ชื่อว่ากลุ่ม p1 จากนั้นเลือกมาเป็น 60% ของ generation ถัดไป โดยเลือกโครโมโซม ที่มีค่า fitness ดีที่สุดจากทั้งหมด โดยค่า fitness นั้นหาจากจำนวนความถูกต้องของคำตอบจากข้อมูลใน training set และโครโมโซมที่เหลืออีก 40% ได้จากการสุ่มโครโมโซมในกลุ่ม p1 มาทำการ mutate โดยสุ่ม node ของ neuron network มา 30% จาก node ทั้งหมด และเพิ่มค่าน้ำหนักขาเข้า node นั้น โดยสุ่มจาก -1 ถึง 1 จากนั้นนำโครโมโซมที่มีค่า fitness ที่ดีที่สุดใน generation สุดท้ายมาทดสอบกับชุดข้อมูลทดสอบ เพื่อหาค่าความผิดพลาด เมื่อครบทุก fold ในขั้นตอน cross validation แล้วสุดท้ายจะได้ค่าความผิดพลาดเฉลี่ยที่บ่งบอกถึงสมรรถนะของรูปแบบ GA นี้

**ตัวอย่าง output ของโปรแกรม**

```
Neuron network 30-12-2
---------------- Fold: 1 ----------------
Generation 1 best accurate: 77.58284600389864%
Generation 2 best accurate: 77.58284600389864%
Generation 3 best accurate: 80.50682261208577%
Generation 4 best accurate: 88.10916179337232%
Generation 5 best accurate: 88.69395711500975%
Generation 6 best accurate: 88.69395711500975%
Generation 7 best accurate: 89.8635477582846%
Generation 8 best accurate: 89.8635477582846%
Generation 9 best accurate: 89.8635477582846%
Generation 10 best accurate: 89.8635477582846%
Generation 11 best accurate: 89.8635477582846%
Generation 12 best accurate: 91.2280701754386%
Generation 13 best accurate: 91.81286549707602%
Generation 14 best accurate: 92.00779727095517%

Generation 15 best accurate: 92.00779727095517%

Generation 16 best accurate: 92.20272904483431%

Generation 17 best accurate: 92.20272904483431%

Generation 18 best accurate: 92.20272904483431%

Generation 19 best accurate: 92.20272904483431%

Generation 20 best accurate: 92.20272904483431%

Generation 21 best accurate: 92.20272904483431%

Generation 22 best accurate: 92.20272904483431%

Generation 23 best accurate: 92.20272904483431%

Generation 24 best accurate: 92.5925925925926%

Generation 25 best accurate: 92.5925925925926%

Generation 26 best accurate: 92.5925925925926%

Generation 27 best accurate: 92.5925925925926%

Generation 28 best accurate: 92.5925925925926%

Generation 29 best accurate: 92.5925925925926%

Generation 30 best accurate: 92.5925925925926%

Generation 31 best accurate: 92.5925925925926%

Generation 32 best accurate: 92.5925925925926%

Generation 33 best accurate: 92.78752436647173%

Generation 34 best accurate: 92.78752436647173%

Generation 35 best accurate: 92.78752436647173%

Generation 36 best accurate: 93.17738791423002%

Generation 37 best accurate: 93.17738791423002%

Generation 38 best accurate: 93.17738791423002%

Generation 39 best accurate: 93.17738791423002%

Generation 40 best accurate: 93.17738791423002%

Generation 41 best accurate: 93.17738791423002%

Generation 42 best accurate: 93.17738791423002%

Generation 43 best accurate: 93.17738791423002%

Generation 44 best accurate: 93.17738791423002%

Generation 45 best accurate: 93.17738791423002%

Generation 46 best accurate: 93.17738791423002%

Generation 47 best accurate: 93.17738791423002%
Generation 48 best accurate: 93.17738791423002%
Generation 49 best accurate: 93.17738791423002%
Generation 50 best accurate: 93.17738791423002%
Testing accurate: 85.71428571428571%

---------------- Fold: 2 ----------------
Generation 1 best accurate: 86.93957115009746%
Generation 2 best accurate: 86.93957115009746%
Generation 3 best accurate: 87.91423001949317%
Generation 4 best accurate: 87.91423001949317%
Generation 5 best accurate: 87.91423001949317%
Generation 6 best accurate: 88.30409356725146%
Generation 7 best accurate: 88.88888888888889%
Generation 8 best accurate: 88.88888888888889%
Generation 9 best accurate: 88.88888888888889%
Generation 10 best accurate: 88.88888888888889%
Generation 11 best accurate: 89.8635477582846%
Generation 12 best accurate: 89.8635477582846%
Generation 13 best accurate: 90.05847953216374%
Generation 14 best accurate: 90.05847953216374%
Generation 15 best accurate: 90.44834307992203%
Generation 16 best accurate: 90.44834307992203%
Generation 17 best accurate: 90.44834307992203%
Generation 18 best accurate: 90.83820662768031%
Generation 19 best accurate: 90.83820662768031%
Generation 20 best accurate: 90.83820662768031%
Generation 21 best accurate: 90.83820662768031%
Generation 22 best accurate: 90.83820662768031%
Generation 23 best accurate: 90.83820662768031%
Generation 24 best accurate: 91.03313840155946%
Generation 25 best accurate: 91.2280701754386%
Generation 26 best accurate: 91.61793372319688%
Generation 27 best accurate: 91.61793372319688%
Generation 28 best accurate: 91.61793372319688%
Generation 29 best accurate: 91.61793372319688%
Generation 30 best accurate: 91.61793372319688%
Generation 31 best accurate: 91.61793372319688%
Generation 32 best accurate: 91.61793372319688%
Generation 33 best accurate: 91.61793372319688%
Generation 34 best accurate: 91.61793372319688%
Generation 35 best accurate: 91.61793372319688%
Generation 36 best accurate: 91.61793372319688%
Generation 37 best accurate: 91.61793372319688%
Generation 38 best accurate: 91.61793372319688%
Generation 39 best accurate: 91.61793372319688%
Generation 40 best accurate: 91.61793372319688%
Generation 41 best accurate: 91.61793372319688%
Generation 42 best accurate: 91.61793372319688%
Generation 43 best accurate: 92.00779727095517%
Generation 44 best accurate: 92.00779727095517%
Generation 45 best accurate: 92.00779727095517%
Generation 46 best accurate: 92.00779727095517%
Generation 47 best accurate: 92.00779727095517%
Generation 48 best accurate: 92.00779727095517%
Generation 49 best accurate: 92.00779727095517%
Generation 50 best accurate: 92.00779727095517%
Testing accurate: 92.85714285714286%

---------------- Fold: 3 ----------------
Generation 1 best accurate: 78.3625730994152%
Generation 2 best accurate: 80.70175438596492%
Generation 3 best accurate: 80.70175438596492%
Generation 4 best accurate: 80.70175438596492%
Generation 5 best accurate: 84.99025341130604%
Generation 6 best accurate: 88.10916179337232%
Generation 7 best accurate: 88.88888888888889%
Generation 8 best accurate: 88.88888888888889%
Generation 9 best accurate: 88.88888888888889%
Generation 10 best accurate: 89.66861598440546%
Generation 11 best accurate: 89.66861598440546%
Generation 12 best accurate: 89.66861598440546%
Generation 13 best accurate: 89.66861598440546%
Generation 14 best accurate: 89.66861598440546%
Generation 15 best accurate: 89.66861598440546%
Generation 16 best accurate: 89.66861598440546%
Generation 17 best accurate: 89.66861598440546%
Generation 18 best accurate: 90.25341130604288%
Generation 19 best accurate: 90.25341130604288%
Generation 20 best accurate: 90.64327485380117%
Generation 21 best accurate: 90.64327485380117%
Generation 22 best accurate: 91.42300194931774%
Generation 23 best accurate: 91.42300194931774%
Generation 24 best accurate: 91.42300194931774%
Generation 25 best accurate: 91.42300194931774%
Generation 26 best accurate: 91.42300194931774%
Generation 27 best accurate: 91.42300194931774%
Generation 28 best accurate: 91.42300194931774%
Generation 29 best accurate: 91.42300194931774%
Generation 30 best accurate: 91.42300194931774%
Generation 31 best accurate: 91.42300194931774%
Generation 32 best accurate: 91.61793372319688%
Generation 33 best accurate: 91.61793372319688%
Generation 34 best accurate: 91.61793372319688%
Generation 35 best accurate: 91.61793372319688%
Generation 36 best accurate: 92.00779727095517%
Generation 37 best accurate: 92.00779727095517%
Generation 38 best accurate: 92.00779727095517%
Generation 39 best accurate: 92.00779727095517%
Generation 40 best accurate: 92.00779727095517%
Generation 41 best accurate: 92.5925925925926%
Generation 42 best accurate: 92.5925925925926%
Generation 43 best accurate: 92.5925925925926%
Generation 44 best accurate: 92.5925925925926%

Generation 45 best accurate: 92.5925925925926%

Generation 46 best accurate: 92.5925925925926%

Generation 47 best accurate: 92.5925925925926%

Generation 48 best accurate: 92.5925925925926%

Generation 49 best accurate: 92.78752436647173%

Generation 50 best accurate: 92.98245614035088%

Testing accurate: 94.64285714285714%


---------------- Fold: 4 ----------------

Generation 1 best accurate: 78.94736842105263%

Generation 2 best accurate: 88.30409356725146%

Generation 3 best accurate: 90.44834307992203%

Generation 4 best accurate: 90.44834307992203%

Generation 5 best accurate: 90.44834307992203%

Generation 6 best accurate: 90.44834307992203%

Generation 7 best accurate: 90.44834307992203%

Generation 8 best accurate: 90.44834307992203%

Generation 9 best accurate: 90.83820662768031%

Generation 10 best accurate: 90.83820662768031%

Generation 11 best accurate: 90.83820662768031%

Generation 12 best accurate: 90.83820662768031%

Generation 13 best accurate: 90.83820662768031%

Generation 14 best accurate: 90.83820662768031%

Generation 15 best accurate: 90.83820662768031%

Generation 16 best accurate: 90.83820662768031%

Generation 17 best accurate: 90.83820662768031%

Generation 18 best accurate: 90.83820662768031%

Generation 19 best accurate: 90.83820662768031%

Generation 20 best accurate: 90.83820662768031%

Generation 21 best accurate: 90.83820662768031%

Generation 22 best accurate: 90.83820662768031%

Generation 23 best accurate: 90.83820662768031%

Generation 24 best accurate: 90.83820662768031%
Generation 25 best accurate: 90.83820662768031%

Generation 26 best accurate: 91.2280701754386%

Generation 27 best accurate: 91.2280701754386%

Generation 28 best accurate: 91.2280701754386%

Generation 29 best accurate: 91.2280701754386%

Generation 30 best accurate: 91.61793372319688%

Generation 31 best accurate: 91.61793372319688%

Generation 32 best accurate: 91.61793372319688%

Generation 33 best accurate: 92.78752436647173%

Generation 34 best accurate: 92.78752436647173%

Generation 35 best accurate: 92.78752436647173%

Generation 36 best accurate: 92.78752436647173%

Generation 37 best accurate: 92.78752436647173%

Generation 38 best accurate: 92.78752436647173%

Generation 39 best accurate: 92.78752436647173%

Generation 40 best accurate: 92.78752436647173%

Generation 41 best accurate: 92.78752436647173%

Generation 42 best accurate: 92.78752436647173%

Generation 43 best accurate: 92.78752436647173%

Generation 44 best accurate: 92.78752436647173%

Generation 45 best accurate: 92.78752436647173%

Generation 46 best accurate: 92.78752436647173%

Generation 47 best accurate: 92.78752436647173%

Generation 48 best accurate: 92.78752436647173%

Generation 49 best accurate: 92.98245614035088%

Generation 50 best accurate: 92.98245614035088%

Testing accurate: 92.85714285714286%


---------------- Fold: 5 ----------------

Generation 1 best accurate: 91.03313840155946%

Generation 2 best accurate: 91.03313840155946%

Generation 3 best accurate: 91.03313840155946%

Generation 4 best accurate: 91.03313840155946%

Generation 5 best accurate: 91.03313840155946%
Generation 6 best accurate: 91.033138401559946%
Generation 7 best accurate: 91.033138401559946%
Generation 8 best accurate: 91.033138401559946%
Generation 9 best accurate: 91.033138401559946%
Generation 10 best accurate: 91.033138401559946%
Generation 11 best accurate: 91.033138401559946%
Generation 12 best accurate: 91.033138401559946%
Generation 13 best accurate: 91.033138401559946%
Generation 14 best accurate: 91.033138401559946%
Generation 15 best accurate: 91.033138401559946%
Generation 16 best accurate: 91.033138401559946%
Generation 17 best accurate: 91.033138401559946%
Generation 18 best accurate: 91.033138401559946%
Generation 19 best accurate: 91.033138401559946%
Generation 20 best accurate: 91.2280701754386%
Generation 21 best accurate: 91.2280701754386%
Generation 22 best accurate: 91.2280701754386%
Generation 23 best accurate: 91.2280701754386%
Generation 24 best accurate: 91.2280701754386%
Generation 25 best accurate: 91.2280701754386%
Generation 26 best accurate: 91.2280701754386%
Generation 27 best accurate: 91.2280701754386%
Generation 28 best accurate: 91.2280701754386%
Generation 29 best accurate: 91.2280701754386%
Generation 30 best accurate: 91.2280701754386%
Generation 31 best accurate: 91.42300194931774%
Generation 32 best accurate: 91.42300194931774%
Generation 33 best accurate: 91.42300194931774%
Generation 34 best accurate: 91.42300194931774%
Generation 35 best accurate: 91.42300194931774%
Generation 36 best accurate: 91.42300194931774%
Generation 37 best accurate: 91.61793372319688%
Generation 38 best accurate: 91.61793372319688%
Generation 39 best accurate: 91.61793372319688%
Generation 40 best accurate: 91.61793372319688%

Generation 41 best accurate: 91.61793372319688%

Generation 42 best accurate: 91.61793372319688%

Generation 43 best accurate: 91.61793372319688%

Generation 44 best accurate: 91.61793372319688%

Generation 45 best accurate: 91.61793372319688%

Generation 46 best accurate: 91.61793372319688%

Generation 47 best accurate: 91.61793372319688%

Generation 48 best accurate: 91.61793372319688%

Generation 49 best accurate: 91.61793372319688%

Generation 50 best accurate: 91.61793372319688%

Testing accurate: 82.14285714285714%


---------------- Fold: 6 ----------------

Generation 1 best accurate: 77.77777777777777%

Generation 2 best accurate: 77.77777777777777%

Generation 3 best accurate: 81.48148148148148%

Generation 4 best accurate: 89.8635477582846%

Generation 5 best accurate: 90.25341130604288%

Generation 6 best accurate: 90.25341130604288%

Generation 7 best accurate: 90.25341130604288%

Generation 8 best accurate: 90.25341130604288%

Generation 9 best accurate: 92.20272904483431%

Generation 10 best accurate: 92.20272904483431%

Generation 11 best accurate: 92.20272904483431%

Generation 12 best accurate: 92.20272904483431%

Generation 13 best accurate: 92.20272904483431%

Generation 14 best accurate: 92.20272904483431%

Generation 15 best accurate: 92.20272904483431%

Generation 16 best accurate: 92.20272904483431%

Generation 17 best accurate: 92.20272904483431%

Generation 18 best accurate: 92.20272904483431%

Generation 19 best accurate: 92.20272904483431%

Generation 20 best accurate: 92.20272904483431%
Generation 21 best accurate: 92.20272904483431%

Generation 22 best accurate: 92.20272904483431%

Generation 23 best accurate: 92.5925925925926%

Generation 24 best accurate: 92.5925925925926%

Generation 25 best accurate: 92.5925925925926%

Generation 26 best accurate: 92.5925925925926%

Generation 27 best accurate: 92.5925925925926%

Generation 28 best accurate: 92.5925925925926%

Generation 29 best accurate: 92.5925925925926%

Generation 30 best accurate: 92.98245614035088%

Generation 31 best accurate: 92.98245614035088%

Generation 32 best accurate: 92.98245614035088%

Generation 33 best accurate: 92.98245614035088%

Generation 34 best accurate: 92.98245614035088%

Generation 35 best accurate: 92.98245614035088%

Generation 36 best accurate: 92.98245614035088%

Generation 37 best accurate: 92.98245614035088%

Generation 38 best accurate: 92.98245614035088%

Generation 39 best accurate: 92.98245614035088%

Generation 40 best accurate: 92.98245614035088%

Generation 41 best accurate: 92.98245614035088%

Generation 42 best accurate: 92.98245614035088%

Generation 43 best accurate: 92.98245614035088%

Generation 44 best accurate: 92.98245614035088%

Generation 45 best accurate: 92.98245614035088%

Generation 46 best accurate: 92.98245614035088%

Generation 47 best accurate: 92.98245614035088%

Generation 48 best accurate: 92.98245614035088%

Generation 49 best accurate: 92.98245614035088%

Generation 50 best accurate: 92.98245614035088%

Testing accurate: 87.5%


---------------- Fold: 7 ----------------

Generation 1 best accurate: 81.67641325536063%
Generation 2 best accurate: 81.67641325536063%

Generation 3 best accurate: 81.67641325536063%

Generation 4 best accurate: 82.06627680311891%

Generation 5 best accurate: 88.4990253411306%

Generation 6 best accurate: 88.4990253411306%

Generation 7 best accurate: 88.4990253411306%

Generation 8 best accurate: 90.44834307992203%

Generation 9 best accurate: 90.44834307992203%

Generation 10 best accurate: 90.44834307992203%

Generation 11 best accurate: 90.44834307992203%

Generation 12 best accurate: 90.44834307992203%

Generation 13 best accurate: 90.44834307992203%

Generation 14 best accurate: 90.44834307992203%

Generation 15 best accurate: 90.44834307992203%

Generation 16 best accurate: 90.44834307992203%

Generation 17 best accurate: 90.44834307992203%

Generation 18 best accurate: 90.44834307992203%

Generation 19 best accurate: 90.44834307992203%

Generation 20 best accurate: 90.44834307992203%

Generation 21 best accurate: 90.44834307992203%

Generation 22 best accurate: 90.44834307992203%

Generation 23 best accurate: 90.44834307992203%

Generation 24 best accurate: 90.44834307992203%

Generation 25 best accurate: 91.03313840155946%

Generation 26 best accurate: 91.2280701754386%

Generation 27 best accurate: 91.42300194931774%

Generation 28 best accurate: 91.42300194931774%

Generation 29 best accurate: 91.42300194931774%

Generation 30 best accurate: 91.42300194931774%

Generation 31 best accurate: 91.42300194931774%

Generation 32 best accurate: 91.61793372319688%

Generation 33 best accurate: 91.61793372319688%

Generation 34 best accurate: 91.61793372319688%

Generation 35 best accurate: 91.61793372319688%
Generation 36 best accurate: 91.61793372319688%
Generation 37 best accurate: 91.61793372319688%
Generation 38 best accurate: 91.61793372319688%
Generation 39 best accurate: 91.81286549707602%
Generation 40 best accurate: 91.81286549707602%
Generation 41 best accurate: 91.81286549707602%
Generation 42 best accurate: 91.81286549707602%
Generation 43 best accurate: 91.81286549707602%
Generation 44 best accurate: 91.81286549707602%
Generation 45 best accurate: 91.81286549707602%
Generation 46 best accurate: 91.81286549707602%
Generation 47 best accurate: 91.81286549707602%
Generation 48 best accurate: 91.81286549707602%
Generation 49 best accurate: 92.00779727095517%
Generation 50 best accurate: 92.00779727095517%
Testing accurate: 92.85714285714286%

---------------- Fold: 8 ----------------
Generation 1 best accurate: 83.23586744639377%
Generation 2 best accurate: 86.1598440545809%
Generation 3 best accurate: 86.1598440545809%
Generation 4 best accurate: 86.1598440545809%
Generation 5 best accurate: 86.1598440545809%
Generation 6 best accurate: 86.1598440545809%
Generation 7 best accurate: 91.03313840155946%
Generation 8 best accurate: 91.03313840155946%
Generation 9 best accurate: 91.03313840155946%
Generation 10 best accurate: 91.03313840155946%
Generation 11 best accurate: 91.03313840155946%
Generation 12 best accurate: 91.03313840155946%
Generation 13 best accurate: 91.03313840155946%
Generation 14 best accurate: 91.03313840155946%
Generation 15 best accurate: 91.03313840155946%
Generation 16 best accurate: 91.03313840155946%
Generation 17 best accurate: 91.03313840155946%
Generation 18 best accurate: 91.03313840155946%
Generation 19 best accurate: 91.03313840155946%
Generation 20 best accurate: 91.03313840155946%
Generation 21 best accurate: 91.03313840155946%
Generation 22 best accurate: 91.03313840155946%
Generation 23 best accurate: 91.03313840155946%
Generation 24 best accurate: 91.03313840155946%
Generation 25 best accurate: 91.2280701754386%
Generation 26 best accurate: 91.2280701754386%
Generation 27 best accurate: 91.2280701754386%
Generation 28 best accurate: 91.2280701754386%
Generation 29 best accurate: 91.2280701754386%
Generation 30 best accurate: 91.2280701754386%
Generation 31 best accurate: 91.2280701754386%
Generation 32 best accurate: 91.2280701754386%
Generation 33 best accurate: 91.2280701754386%
Generation 34 best accurate: 91.2280701754386%
Generation 35 best accurate: 91.2280701754386%
Generation 36 best accurate: 91.2280701754386%
Generation 37 best accurate: 91.2280701754386%
Generation 38 best accurate: 91.2280701754386%
Generation 39 best accurate: 91.2280701754386%
Generation 40 best accurate: 91.42300194931774%
Generation 41 best accurate: 91.42300194931774%
Generation 42 best accurate: 91.42300194931774%
Generation 43 best accurate: 91.42300194931774%
Generation 44 best accurate: 91.42300194931774%
Generation 45 best accurate: 91.61793372319688%
Generation 46 best accurate: 91.61793372319688%
Generation 47 best accurate: 91.61793372319688%
Generation 48 best accurate: 91.61793372319688%
Generation 49 best accurate: 91.61793372319688%
Generation 50 best accurate: 91.61793372319688%
```

```
Testing accurate: 92.85714285714286%

---------------- Fold: 9 ----------------
Generation 1 best accurate: 84.60038986354776%
Generation 2 best accurate: 86.1598440545809%
Generation 3 best accurate: 86.93957115009746%
Generation 4 best accurate: 88.69395711500975%
Generation 5 best accurate: 88.88888888888889%
Generation 6 best accurate: 88.88888888888889%
Generation 7 best accurate: 88.88888888888889%
Generation 8 best accurate: 90.25341130604288%
Generation 9 best accurate: 91.2280701754386%
Generation 10 best accurate: 91.2280701754386%
Generation 11 best accurate: 91.2280701754386%
Generation 12 best accurate: 91.2280701754386%
Generation 13 best accurate: 91.2280701754386%
Generation 14 best accurate: 91.2280701754386%
Generation 15 best accurate: 91.2280701754386%
Generation 16 best accurate: 91.2280701754386%
Generation 17 best accurate: 91.42300194931774%
Generation 18 best accurate: 91.42300194931774%
Generation 19 best accurate: 91.61793372319688%
Generation 20 best accurate: 91.61793372319688%
Generation 21 best accurate: 91.61793372319688%
Generation 22 best accurate: 91.61793372319688%
Generation 23 best accurate: 91.61793372319688%
Generation 24 best accurate: 91.61793372319688%
Generation 25 best accurate: 91.61793372319688%
Generation 26 best accurate: 91.61793372319688%
Generation 27 best accurate: 91.81286549707602%
Generation 28 best accurate: 92.39766081871345%
Generation 29 best accurate: 92.39766081871345%
Generation 30 best accurate: 92.39766081871345%
Generation 31 best accurate: 92.39766081871345%
Generation 32 best accurate: 92.39766081871345%
Generation 33 best accurate: 92.39766081871345%
Generation 34 best accurate: 92.5925925925926%
Generation 35 best accurate: 92.5925925925926%
Generation 36 best accurate: 92.5925925925926%
Generation 37 best accurate: 92.5925925925926%
Generation 38 best accurate: 92.5925925925926%
Generation 39 best accurate: 92.5925925925926%
Generation 40 best accurate: 92.5925925925926%
Generation 41 best accurate: 92.5925925925926%
Generation 42 best accurate: 92.5925925925926%
Generation 43 best accurate: 92.5925925925926%
Generation 44 best accurate: 92.5925925925926%
Generation 45 best accurate: 92.5925925925926%
Generation 46 best accurate: 92.5925925925926%
Generation 47 best accurate: 92.78752436647173%
Generation 48 best accurate: 92.78752436647173%
Generation 49 best accurate: 92.78752436647173%
Generation 50 best accurate: 92.78752436647173%
Testing accurate: 89.28571428571429%

---------------- Fold: 10 ----------------
Generation 1 best accurate: 86.1598440545809%
Generation 2 best accurate: 86.1598440545809%
Generation 3 best accurate: 86.54970760233918%
Generation 4 best accurate: 89.47368421052632%
Generation 5 best accurate: 89.47368421052632%
Generation 6 best accurate: 89.47368421052632%
Generation 7 best accurate: 89.47368421052632%
Generation 8 best accurate: 89.47368421052632%
Generation 9 best accurate: 89.47368421052632%
Generation 10 best accurate: 89.47368421052632%
Generation 11 best accurate: 89.47368421052632%
Generation 12 best accurate: 89.66861598440546%
Generation 13 best accurate: 91.03313840155946%
Generation 14 best accurate: 91.03313840155946%
Generation 15 best accurate: 91.03313840155946%
Generation 16 best accurate: 91.03313840155946%
Generation 17 best accurate: 92.20272904483431%
Generation 18 best accurate: 92.20272904483431%
Generation 19 best accurate: 92.20272904483431%
Generation 20 best accurate: 92.20272904483431%
Generation 21 best accurate: 92.20272904483431%
Generation 22 best accurate: 92.20272904483431%
Generation 23 best accurate: 92.20272904483431%
Generation 24 best accurate: 92.20272904483431%
Generation 25 best accurate: 92.20272904483431%
Generation 26 best accurate: 92.20272904483431%
Generation 27 best accurate: 92.20272904483431%
Generation 28 best accurate: 92.78752436647173%
Generation 29 best accurate: 92.78752436647173%
Generation 30 best accurate: 92.78752436647173%
Generation 31 best accurate: 92.78752436647173%
Generation 32 best accurate: 92.78752436647173%
Generation 33 best accurate: 92.78752436647173%
Generation 34 best accurate: 92.78752436647173%
Generation 35 best accurate: 92.78752436647173%
Generation 36 best accurate: 92.78752436647173%
Generation 37 best accurate: 92.78752436647173%
Generation 38 best accurate: 92.78752436647173%
Generation 39 best accurate: 92.78752436647173%
Generation 40 best accurate: 92.78752436647173%
Generation 41 best accurate: 92.78752436647173%
Generation 42 best accurate: 92.78752436647173%
Generation 43 best accurate: 92.78752436647173%
Generation 44 best accurate: 92.78752436647173%
Generation 45 best accurate: 92.78752436647173%
Generation 46 best accurate: 92.78752436647173%
Generation 47 best accurate: 92.98245614035088%

Generation 48 best accurate: 92.98245614035088%

Generation 49 best accurate: 92.98245614035088%

Generation 50 best accurate: 92.98245614035088%

Testing accurate: 92.857142857142856%

Error average: 9.642857142857142%
```

**การทดลอง**

1. ทำการทดลองโดยเปลี่ยน hidden layer ของ Neuron network เป็น 30-40-20-2, 30-15-7-2, 30-12-2, 30-10-5-2 โดยมีโครโมโซมใน GA เป็น 50 และ training สิ้นสุดใน generation ที่ 50 ได้ผลการทดลองดังนี้

   Neuron network 30-40-20-2

   ความผิดพลาดเฉลี่ยในการทำ 10% cross validation คือ 8.57142857142857%

   Neuron network 30-15-7-2

   ความผิดพลาดเฉลี่ยในการทำ 10% cross validation คือ 9.10714285714286%

   Neuron network 30-12-2

   ความผิดพลาดเฉลี่ยในการทำ 10% cross validation คือ 9.107142857142858%

   Neuron network 30-10-5-2

   ความผิดพลาดเฉลี่ยในการทำ 10% cross validation คือ 8.571428571428571%

2. ทำการทดลองโดยใช้ Neuron network 30-10-5-2 โดยการเพิ่มโครโมโซมเป็น 200 และเพิ่ม generation การ train เป็น 200 ได้ผลการทดลองดังนี้

   การเพิ่มโครโมโซมเป็น 200 ได้ความผิดพลาดเฉลี่ยคคือ 9.464285714285714%

   การเพิ่ม generation การ train เป็น 200 ได้ความผิดพลาดเฉลี่ยคคือ 8.928571428571427%

**วิเคราะห์ผลการทดลอง**

จากการทดลองการเปลี่ยน hidden layer ของ Neuron network จะเห็นได้ว่าความผิดพลาดเฉลี่ยนั้นมีค่าใกล้เคียงกันและโครงสร้างที่ดีที่สุดในการทดลองนี้คือ Neuron network 30-40-20-2 และในการทดลองการเพิ่มโครโมโซมและเพิ่ม generation การ train นั้น จากการทดลองแรก ใช้จำนวนโครโมโซมเป็น 50 และ generation การ train เป็น 50 ความผิดพลาดเฉลี่ยคคือ 8.571428571428571% จะเห็นได้ว่าการเพิ่มจำนวนโครโมโซมและ generation การ train นั้นจะทำให้ความผิดพลาดเฉลี่ย เพิ่มขึ้น เพราะเป็นการเรียนรู้มากเกินไป (overtrain) ค่าความถูกต้องในขั้นตอน training จะมีค่ามากขึ้น แต่เมื่อนำข้อมูลทดสอบที่ต่างจากข้อมูล training มาทดสอบจะทำให้เกิดค่าความผิดพลาดมากกว่า

**Code** (https://github.com/porpeeranut/Computational_Intelligence_Assignment3 )

```
//		main.java

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collections;

public class main {

	public static void main(String[] args) {
		String path = "..\\wdbc.data.txt";
		File file = new File(path);
		// feature 0-29, output 30
		ArrayList<Double[]> trainingSet = new ArrayList<Double[]>();
		try {
			BufferedReader br = new BufferedReader(new FileReader(file));
			String line;
			while ((line = br.readLine()) != null) {
				String[] data = line.split(",");
				Double[] feature = new Double[31];
				if (data[1].equals("M"))
					feature[30] = 0.0;
				else
					feature[30] = 1.0;
				for (int j = 2;j < data.length;j++) {
					feature[j-2] = Double.parseDouble(data[j]);
				}
				trainingSet.add(feature);
			}
			Collections.shuffle(trainingSet);
			System.out.println(trainingSet.size());
			br.close();
		} catch (IOException e) {
			e.printStackTrace();
		}

		GA ga = new GA();
```

```
        //int[] MLP_struct = {30,40,20,2};
        //int[] MLP_struct = {30,15,7,2};
        //int[] MLP_struct = {30,12,2};
        int[] MLP_struct = {30,10,5,2};

        System.out.print("Neuron network ");
        for (int i = 0;i < MLP_struct.length;i++) {
            if (i != 0)
                System.out.print('-');
            System.out.print(MLP_struct[i]);
        }
        ga.initChromosome(200, MLP_struct);
        double eav = 0;
        for (int c = 0;c < 10;c++) {
            System.out.println("\n---------------- Fold: "+(c+1)+" ----------------");
            int i = (int) (c*trainingSet.size()*0.1);
            ArrayList<Double[]> test = new ArrayList<Double[]>(trainingSet.subList(i, (int)
(i+(trainingSet.size()*0.1))));
            ArrayList<Double[]> train = (ArrayList<Double[]>) trainingSet.clone();
            train.subList(i, (int) (i+(train.size()*0.1))).clear();
            ga.train(50, (ArrayList<Double[]>) train);
            eav += ga.test(test);
        }
        System.out.println("Error average: "+eav/10+"%");
    }
}
```

```
//          GA.java

import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.LinkedHashSet;
import java.util.Random;
import java.util.Set;

public class GA {

        ArrayList<Chromosome> initChromosome = new ArrayList<Chromosome>();
        ArrayList<Chromosome> chromosomeList = new ArrayList<Chromosome>();
        ArrayList<Double[]> trainingSet;
        int[] MLP_struct;
        double wMin = -1.0;
        double wMax = 1.0;

        public void initChromosome(int amount, int[] MLP_struct){
                this.MLP_struct = MLP_struct;
                int chromoLen = 0;
                for (int i = 1;i < MLP_struct.length;i++) {
                        chromoLen += MLP_struct[i-1]*MLP_struct[i];
                }
                for (int a = 0;a < amount;a++) {
                        Double[] gene = new Double[chromoLen];
                        for (int g = 0;g < gene.length;g++) {
                                gene[g] = wMin + (wMax - wMin)*new Random().nextDouble();
                        }
                        initChromosome.add(new Chromosome(gene));
                }
        }

        public void train(int maxGeneration, ArrayList<Double[]> trainingSet){
                this.trainingSet = trainingSet;
                chromosomeList = (ArrayList<Chromosome>) initChromosome.clone();
                computeFitnessInList(chromosomeList);
                for (int g = 0;g < maxGeneration;g++) {
                        ArrayList<Chromosome> selected = randomSelect((int) (chromosomeList.size()*0.9));
                        ArrayList<Chromosome> crossed = crossover(selected, (int) (selected.size()*0.9));
```

```
            ArrayList<Chromosome> pool = (ArrayList<Chromosome>) chromosomeList.clone();
            pool.addAll(crossed);
            //ArrayList<Chromosome> mutated = mutate(pool, (int) (pool.size()*0.7));
            int mutateAmount = (int) (chromosomeList.size()*0.4);
            ArrayList<Chromosome> mutated = mutate(pool, mutateAmount);
            //pool.addAll(mutated);
            Collections.sort(pool, new Comparator<Chromosome>() {
          public int compare(Chromosome c1, Chromosome c2) {
             return (int) (c2.fitness - c1.fitness);
          }
         });
            chromosomeList = new ArrayList<Chromosome>(pool.subList(0, chromosomeList.size()-
mutateAmount));
            //chromosomeList = new ArrayList<Chromosome>(pool.subList(0, chromosomeList.size()));
            chromosomeList.addAll(mutated);
            System.out.println("Generation "+(g+1)+" best accurate:
"+(chromosomeList.get(0).fitness*100/trainingSet.size())+"%");
        }

        Collections.sort(chromosomeList, new Comparator<Chromosome>() {
      public int compare(Chromosome c1, Chromosome c2) {
         return (int) (c2.fitness - c1.fitness);
      }
     });
        for (Chromosome chromosome : chromosomeList) {
            //System.out.println(chromosome.fitness+" "+(chromosome.fitness*100/trainingSet.size()));
        }
    }

    public double test(ArrayList<Double[]> testSet){
        MLP mlp = new MLP(MLP_struct);
        Chromosome bestChomos = chromosomeList.get(0);
        mlp.setWeightFromChromosome(bestChomos);
        bestChomos.fitness = 0.0;
        for (Double[] data : testSet) {
            if (mlp.computeForward(data))
                bestChomos.fitness += 1;
        }
        //computeFitness(chromosomeList.get(0));
        System.out.println("Testing accurate: "+(bestChomos.fitness*100/testSet.size())+"%");
```

```
		return 100-(bestChomos.fitness*100/testSet.size());
	}

	private void computeFitnessInList(ArrayList<Chromosome> chromosomeList) {
		for (Chromosome chromosome : chromosomeList) {
			computeFitness(chromosome);
		}
	}

	private void computeFitness(Chromosome chromosome) {
		MLP mlp = new MLP(MLP_struct);
		mlp.setWeightFromChromosome(chromosome);
		chromosome.fitness = 0.0;
		for (Double[] data : trainingSet) {
			if (mlp.computeForward(data))
				chromosome.fitness += 1;
		}
	}

	private ArrayList<Chromosome> randomSelect(int amount) {
		ArrayList<Chromosome> selected = new ArrayList<Chromosome>();
		for (int a = 0;a < amount;a++) {
			int i = new Random().nextInt(chromosomeList.size());
			selected.add(chromosomeList.get(i));
		}
		return selected;
	}

	private ArrayList<Chromosome> crossover(ArrayList<Chromosome> selected, int amount) {
		ArrayList<Chromosome> crossed = new ArrayList<Chromosome>();
		for (int a = 0;a < amount;a++) {
			int i1 = new Random().nextInt(selected.size());
			int i2 = new Random().nextInt(selected.size());
			Chromosome daddy = selected.get(i1);
			Chromosome mommy = selected.get(i2);
			int chromosomeLen = selected.get(0).gene.length;
			int helfLen = chromosomeLen/2;
			Double[] gene = new Double[chromosomeLen];
			for (int i = 0;i < helfLen;i++) {
				gene[i] = daddy.gene[i];
```

```
                }
                for (int i = helfLen;i < chromosomeLen;i++) {
                    gene[i] = mommy.gene[i];
                }
                crossed.add(new Chromosome(gene));
            }
            computeFitnessInList(crossed);
            return crossed;
    }

    private ArrayList<Chromosome> mutate(ArrayList<Chromosome> pool, int amount) {
            ArrayList<Chromosome> mutated = new ArrayList<Chromosome>();
            Random rng = new Random();
            Set<Integer> index = new LinkedHashSet<Integer>();
            while (index.size() < amount)
            {
                //        random index
              Integer next = rng.nextInt(pool.size());
              index.add(next);
            }
            int k = 0;
            for (int i : index) {
                Double[] gene = pool.get(i).gene.clone();
                int mutateRate = (int) (pool.get(i).gene.length * 0.3);
                for (int m = 0;m < mutateRate;m++) {
                    int w = rng.nextInt(pool.get(i).gene.length);
                    gene[w] += wMin + (wMax - wMin)*new Random().nextDouble();
                    if (gene[w] > wMax)
                        gene[w] = wMax;
                    if (gene[w] < wMin)
                        gene[w] = wMin;
                }
                mutated.add(new Chromosome(gene));
                computeFitness(mutated.get(k++));
            }
            return mutated;
    }
}
```

```
//	MLP.java

public class MLP {

    int[] MLP_struct;
    double[][][] weight;  // layer, maxnode, maxnode
    int maxNode = 0;

    public MLP(int[] MLP_struct) {
        this.MLP_struct = MLP_struct;
        int allNode = 0;
        for (int struc : MLP_struct) {
            if (maxNode < struc)
                maxNode = struc;
            allNode += struc;
        }
        weight = new double[MLP_struct.length-1][maxNode][maxNode];
    }

    public void setWeightFromChromosome(Chromosome chromosome) {
        int g = 0;
        for (int l = 0;l < MLP_struct.length-1;l++) {
            for (int n1 = 0;n1 < MLP_struct[l];n1++) {
                for (int n2 = 0;n2 < MLP_struct[l+1];n2++) {
                    weight[l][n1][n2] = chromosome.gene[g];
                    g++;
                }
            }
        }
    }

    public boolean computeForward(Double[] data) {
        Double[][] y = new Double[MLP_struct.length][maxNode];    // layer, node
        y[0] = data;
        for (int L = 1;L < MLP_struct.length;L++) {
            for (int n2 = 0;n2 < MLP_struct[L];n2++) {
                Double v = 0.0;
                for (int n1 = 0;n1 < MLP_struct[L-1];n1++) {
                    v += y[L-1][n1] * weight[L-1][n1][n2];
                }
                //y[L][n2] = sigmoid(v);
                y[L][n2] = Math.tanh(v);
            }
        }
        if (data[30] == 1.0) {
            //      "B" 0 1
            if (y[MLP_struct.length-1][0] < y[MLP_struct.length-1][1])
                return true;
            else
                return false;
        } else {
            //      "M" 1 0
            if (y[MLP_struct.length-1][0] > y[MLP_struct.length-1][1])
                return true;
            else
                return false;
        }
    }

    private Double sigmoid(Double v) {
        return 1.0 / (1.0 + Math.exp(-v));
    }
}
```

```
//      Chromosome.java

public class Chromosome {

    public Double[] gene;
    public Double fitness = 0.0;

    public Chromosome(Double[] gene) {
        this.gene = gene;
    }
}
```