**K-Nearest Neighbor   
with Mean Absolute Difference**

**จัดทำโดย**

นายปัณณวิชญ์ พันธ์วงศ์ 600610752

**เสนอ**

รศ.ดร.ศันสนีย์ เอื้อพันธ์วิริยะกุล

**รายงานเล่มนี้เป็นส่วนหนึ่งของวิชา**

**261754**

**ภาคเรียนที่ 1 ปีการศึกษา 2564**

1. **Theories and Related Method**
   1. **Mean Absolute Difference (MAD)**

ความแตกต่างแบบสัมบูรณ์เฉลี่ย โดยคำนวณจากความแตกต่างสัมบูรณ์จากค่าเฉลี่ย โดยมีความคล้ายคลึงกับการหาความแปรปรวน (Variance) ซึ่งความหมายของการคำนวณความแตกต่างแบบสมบูรณ์เฉลี่ยยิ่งมีค่าสูงยิ่งความสำคัญของข้อมูลนั้นๆ โดยคำนวณจากสมการนี้

* 1. **Euclidean distance**

ระยะทางแบบยุคลิด คือ ระยะทางปกติระหว่างจุดสองจุดในแนวเส้นตรง โดยในรายงานเล่มนี้ใช้ในการหาระยะห่างระหว่างข้อมูลใดๆ โดยมีสมการดังนี้

* 1. **K-Nearest Neighbour Algorithm (K-NN)**

ขั้นตอนวิธีการเพื่อนบ้านใกล้ที่สุด เป็นวิธีที่ใช้ในการจัดแบ่งคลาส โดยเทคนิคนี้จะตัดสินใจว่า คลาสใดที่จะแทนเงื่อนไขหรือกรณีใหม่ๆ ได้บ้าง โดยการตรวจสอบจำนวนบางจำนวน (“K” ในขั้นตอนวิธีการเพื่อนบ้านใกล้ที่สุด) ของกรณีหรือเงื่อนไขที่เหมือนกันหรือใกล้เคียงกันมากที่สุด โดยจะหาผลรวม (Count Up) ของจำนวนเงื่อนไข หรือกรณีต่างๆ สำหรับแต่ละคลาส และกำหนดเงื่อนไขใหม่ๆ ให้คลาสที่เหมือนกันกับคลาสที่ใกล้เคียงกันมากที่สุด โดยมีขั้นตอนวิธีการดังนี้

1. กำหนดขนาดของ K
2. คำนวณระยะห่างของข้อมูลที่ต้องการพิจารณากับข้อมูลตัวอย่าง โดยใช้   
   ระยะทางแบบยุคลิด เป็นต้น
3. จัดเรียงลำดับของระยะห่าง และเลือกพิจารณาชุดข้อมูลที่ใกล้จุดที่ต้องการพิจารณาตามจำนวน K ที่กาหนดไว้
4. พิจารณาข้อมูลจำนวน k ชุด และสังเกตว่ากลุ่ม (class) ไหนที่ใกล้จุดที่พิจารณาเป็นจำนวนมากที่สุด
5. กำหนด class ให้กับจุดที่พิจารณา (class) ที่ใกล้จุดพิจารณามากที่สุด
   1. **Cross-validation Test**

การวัดประสิทธิภาพด้วยวิธี Cross-validation นี้จะทําการแบ่งข้อมูลออกเป็นหลายส่วน (มักจะแสดงด้วยค่า k) เช่น 5-fold cross-validation คือ ทําการแบ่งข้อมูลออกเป็น 5 ส่วน โดยที่แต่ละส่วนมีจํานวนข้อมูลเท่ากัน หรือ 10-fold cross-validation คือ การแบ่งข้อมูลออกเป็น 10 ส่วน โดยที่แต่ละส่วนมีจํานวนข้อมูลเท่ากัน หลังจากนั้นข้อมูลหนึ่งส่วนจะใช้เป็นตัวทดสอบประสิทธิภาพของโมเดล ทําวนไปเช่นนี้จนครบจํานวนที่แบ่งไว้ ซึ่งในการทดลองนี้จะใช้ 10-fold cross-validation

1. **Algorithm (Flow Chart)**

อันดับแรกจะทำการ shuffle ข้อมูลทั้งหมด จากนั้นจะทำการทดลองโดยจะทำการแบ่ง 10-fold cross-validation จากข้อมูล 351 records กล่าวคือ ข้อมูลสำหรับการฝึกสอนจัดกลุ่ม ของคลาสใดๆ มี 316 และสำหรับทดสอบ 31 records โดยข้อมูลสำหรับฝึกสอนจะจัดกลุ่มโดยใช้ K-NN โดย มี K = 3 จากนั้นจะทำการวัดผลลัพธ์

|  |
| --- |
| ***Shuffle data***  ***For i, j*** *in k-fold cross validation*   * *train\_sample:* ***data [0 -> i] U data [j -> N]*** * *test\_sample:* ***data [ i -> j ]*** * *distance:* ***Calculate Euclidean distance (test\_sample, train\_sample)*** * *sort\_distance:* ***sorted(****distance****)*** * *predict:* ***KNN (****sort\_distance, k=3****)*** * ***Show Result***   ***Evaluation Result*** |

รูปที่ 1 pseudo code ของระบบ

1. **Experimental**

ข้อมูล Ionosphere data set มีทั้งหมด 351 samples โดยแบ่งออกเป็น 2 class คือ good (g) และ bad (b) โดย g class มี 224 sample และ b class มี 127 sample ใน data set ชุดนี้มี features ทั้งหมด 34 features

ผู้จัดทำได้ทำการทดลอง 2 แบบ ซึ่งแต่ละการทดลองจโดยทำการแบ่งเทรน(Train) กับเทส(Test) โดยใช้ 10-folds cross validation โดยทำตามอัลกอริทึมตาม**ข้อ 2** ซึ่งการทดลองที่ 1 จะใช้ทั้งหมด 34 Features และการทดลองที่ 2 ได้ใช้ Feature selection นั่นคือ Mean Absolute Difference ที่ได้กล่าวใน**ข้อ 1.1**

* 1. **การทดลองที่ 1**

สำหรับการทดลองที่ 1 ใช้ 4 features ในการทดลอง โดยได้ผลลัพธ์ดังนี้

รูปที่ 2 ค่าเฉลี่ยของ Features ในแต่ละ Fold

จาก**รูปที่ 2** สังเกตได้ว่า ค่าเฉลี่ยของ Feature3(สีเขียว) และ Feature4(สีแดง) ของแต่ละ class มีค่าที่ต่างกันเห็นได้อย่างชัดเจน และสำหรับ Feature1(สีฟ้า) มีค่าเฉลี่ยของแต่ละคลาสต่างกันเล็กน้อย แต่สำหรับ Feature2(สีเหลือง) มีค่าที่ใกล้เคียงกันมากๆจนแยกไม่ออก

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 2** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.314 | -0.023 | 0.021 | -0.011 | ***Fea. 1*** | 0.312 | -0.022 | 0.017 | -0.012 |
| ***Fea. 2*** | -0.024 | 0.116 | -0.009 | 0.003 | ***Fea. 2*** | -0.02 | 0.118 | -0.012 | 0.005 |
| ***Fea. 3*** | 0.021 | -0.007 | 0.166 | 0.016 | ***Fea. 3*** | 0.018 | -0.013 | 0.148 | 0.014 |
| ***Fea. 4*** | -0.009 | 0.004 | 0.017 | 0.04 | ***Fea. 4*** | -0.011 | 0.004 | 0.014 | 0.04 |
| **Fold 3** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 4** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.313 | -0.007 | 0.026 | -0.006 | ***Fea. 1*** | 0.301 | -0.023 | 0.015 | -0.015 |
| ***Fea. 2*** | -0.009 | 0.114 | -0.005 | 0.002 | ***Fea. 2*** | -0.023 | 0.119 | -0.008 | 0.006 |
| ***Fea. 3*** | 0.025 | -0.004 | 0.162 | 0.016 | ***Fea. 3*** | 0.015 | -0.009 | 0.152 | 0.01 |
| ***Fea. 4*** | -0.009 | 0.001 | 0.015 | 0.038 | ***Fea. 4*** | -0.015 | 0.006 | 0.01 | 0.038 |
| **Fold 5** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 6** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.282 | -0.032 | 0.019 | -0.013 | ***Fea. 1*** | 0.324 | -0.02 | 0.032 | -0.011 |
| ***Fea. 2*** | -0.033 | 0.113 | -0.011 | 0.006 | ***Fea. 2*** | -0.019 | 0.119 | -0.005 | 0.006 |
| ***Fea. 3*** | 0.017 | -0.011 | 0.145 | 0.018 | ***Fea. 3*** | 0.033 | -0.005 | 0.143 | 0.011 |
| ***Fea. 4*** | -0.014 | 0.006 | 0.018 | 0.041 | ***Fea. 4*** | -0.009 | 0.007 | 0.012 | 0.041 |
| **Fold 7** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 8** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.317 | -0.023 | 0.043 | -0.015 | ***Fea. 1*** | 0.306 | -0.014 | 0.038 | -0.012 |
| ***Fea. 2*** | -0.023 | 0.11 | 0 | 0.011 | ***Fea. 2*** | -0.013 | 0.104 | -0.001 | 0.003 |
| ***Fea. 3*** | 0.044 | 0 | 0.144 | 0.017 | ***Fea. 3*** | 0.04 | -0.002 | 0.158 | 0.018 |
| ***Fea. 4*** | -0.016 | 0.01 | 0.017 | 0.036 | ***Fea. 4*** | -0.013 | 0.002 | 0.017 | 0.039 |
| **Fold 9** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 10** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.326 | -0.017 | 0.024 | -0.015 | ***Fea. 1*** | 0.276 | -0.016 | 0.031 | -0.006 |
| ***Fea. 2*** | -0.02 | 0.099 | -0.018 | 0.01 | ***Fea. 2*** | -0.014 | 0.117 | -0.007 | 0.006 |
| ***Fea. 3*** | 0.025 | -0.015 | 0.157 | 0.017 | ***Fea. 3*** | 0.034 | -0.007 | 0.162 | 0.015 |
| ***Fea. 4*** | -0.013 | 0.011 | 0.017 | 0.04 | ***Fea. 4*** | -0.008 | 0.005 | 0.013 | 0.039 |

รูปที่ 3 Covariance Matrix ของ Class 1 ในแต่ละ Fold

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 2** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.353 | 0.022 | 0 | 0.002 | ***Fea. 1*** | 0.369 | -0.002 | -0.01 | -0.009 |
| ***Fea. 2*** | 0.02 | 0.097 | 0.011 | -0.001 | ***Fea. 2*** | -0.001 | 0.106 | 0.009 | -0.006 |
| ***Fea. 3*** | 0.001 | 0.012 | 0.187 | 0.028 | ***Fea. 3*** | -0.007 | 0.01 | 0.182 | 0.025 |
| ***Fea. 4*** | 0.003 | 0 | 0.028 | 0.078 | ***Fea. 4*** | -0.006 | -0.005 | 0.027 | 0.075 |
| **Fold 3** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 4** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.369 | 0.004 | -0.003 | 0 | ***Fea. 1*** | 0.373 | 0.01 | -0.02 | -0.012 |
| ***Fea. 2*** | 0.007 | 0.105 | 0.014 | -0.002 | ***Fea. 2*** | 0.012 | 0.105 | 0.016 | -0.004 |
| ***Fea. 3*** | 0.002 | 0.014 | 0.194 | 0.025 | ***Fea. 3*** | -0.017 | 0.016 | 0.2 | 0.031 |
| ***Fea. 4*** | 0.003 | -0.001 | 0.026 | 0.079 | ***Fea. 4*** | -0.011 | -0.004 | 0.031 | 0.08 |
| **Fold 5** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 6** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.389 | 0.005 | -0.013 | 0 | ***Fea. 1*** | 0.387 | 0 | -0.011 | -0.009 |
| ***Fea. 2*** | 0.005 | 0.107 | 0.011 | -0.006 | ***Fea. 2*** | 0.001 | 0.104 | 0.003 | -0.008 |
| ***Fea. 3*** | -0.012 | 0.011 | 0.197 | 0.03 | ***Fea. 3*** | -0.009 | 0.003 | 0.175 | 0.024 |
| ***Fea. 4*** | -0.001 | -0.006 | 0.029 | 0.078 | ***Fea. 4*** | -0.007 | -0.007 | 0.025 | 0.081 |
| **Fold 7** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 8** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.366 | 0.001 | -0.01 | -0.001 | ***Fea. 1*** | 0.338 | 0.02 | -0.031 | -0.011 |
| ***Fea. 2*** | 0.001 | 0.103 | 0.013 | -0.004 | ***Fea. 2*** | 0.02 | 0.096 | 0.015 | -0.002 |
| ***Fea. 3*** | -0.013 | 0.012 | 0.19 | 0.024 | ***Fea. 3*** | -0.028 | 0.016 | 0.168 | 0.033 |
| ***Fea. 4*** | -0.003 | -0.005 | 0.023 | 0.081 | ***Fea. 4*** | -0.012 | -0.003 | 0.031 | 0.078 |
| **Fold 9** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** | **Fold 10** | ***Fea. 1*** | ***Fea. 2*** | ***Fea. 3*** | ***Fea. 4*** |
| ***Fea. 1*** | 0.36 | -0.002 | 0.007 | 0 | ***Fea. 1*** | 0.383 | -0.012 | -0.022 | -0.007 |
| ***Fea. 2*** | -0.005 | 0.097 | 0.004 | -0.009 | ***Fea. 2*** | -0.01 | 0.095 | 0.017 | 0.004 |
| ***Fea. 3*** | 0.006 | 0.006 | 0.181 | 0.025 | ***Fea. 3*** | -0.023 | 0.016 | 0.195 | 0.021 |
| ***Fea. 4*** | 0.001 | -0.007 | 0.027 | 0.08 | ***Fea. 4*** | -0.008 | 0.003 | 0.02 | 0.073 |

รูปที่ 4 Covariance Matrix ของ Class 2 ในแต่ละ Fold

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Predict*** | | **Acc** | **Fold 2** | ***Predict*** | | **Acc** |
| ***Actual*** | 10 | 0 | 100 | ***Actual*** | 7 | 1 | 95 |
| 0 | 10 | 0 | 12 |
| **Fold 3** | ***Predict*** | | **Acc** | **Fold 4** | ***Predict*** | | **Acc** |
| ***Actual*** | 12 | 0 | 100 | ***Actual*** | 10 | 0 | 100 |
| 0 | 8 | 0 | 10 |
| **Fold 5** | ***Predict*** | | **Acc** | **Fold 6** | ***Predict*** | | **Acc** |
| ***Actual*** | 11 | 0 | 100 | ***Actual*** | 9 | 0 | 95 |
| 0 | 9 | 1 | 10 |
| **Fold 7** | ***Predict*** | | **Acc** | **Fold 8** | ***Predict*** | | **Acc** |
| ***Actual*** | 9 | 2 | 90 | ***Actual*** | 13 | 0 | 100 |
| 0 | 9 | 0 | 7 |
| **Fold 9** | ***Predict*** | | **Acc** | **Fold 10** | ***Predict*** | | **Acc** |
| ***Actual*** | 9 | 0 | 100 | ***Actual*** | 7 | 0 | 95 |
| 0 | 11 | 1 | 12 |
| ***Mean Acc.*** | | 97.5 | | | | | |

รูปที่ 5 Confusion Matrix Test 1 ในแต่ละ fold

จาก**รูปที่ 5** พบว่าโดยเฉลี่ยทั้ง 10 folds มีค่าความแม่นยำถึง 97.5% โดยค่าความแม่นยำ มีค่าต่ำที่สุดอยู่ใน fold 7 ซึ่งมีค่า 90%

* 1. **การทดลองที่ 2**

สำหรับการทดลองที่ 1 ใช้ 2 features ในการทดลอง โดยเลือกใช้ Feature1 และ Feature2 ได้ผลลัพธ์ดังนี้

รูปที่ 6 ค่าเฉลี่ยของ Features ในแต่ละ Fold

จาก**รูปที่ 6** สังเกตได้ว่า ค่าเฉลี่ยของทั้ง 2 Features ในแต่ละคลาส มีค่าใกล้เคียงกันมากๆ โดยมีเพียงแค่ Feature1 ที่มีค่าต่างกันเล็กน้อย

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Fea. 1*** | ***Fea. 2*** | **Fold 2** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.314 | -0.023 | ***Fea. 1*** | 0.312 | -0.022 |
| ***Fea. 2*** | -0.024 | 0.116 | ***Fea. 2*** | -0.02 | 0.118 |
| **Fold 3** | ***Fea. 1*** | ***Fea. 2*** | **Fold 4** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.313 | -0.007 | ***Fea. 1*** | 0.301 | -0.023 |
| ***Fea. 2*** | -0.009 | 0.114 | ***Fea. 2*** | -0.023 | 0.119 |
| **Fold 5** | ***Fea. 1*** | ***Fea. 2*** | **Fold 6** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.282 | -0.032 | ***Fea. 1*** | 0.324 | -0.02 |
| ***Fea. 2*** | -0.033 | 0.113 | ***Fea. 2*** | -0.019 | 0.119 |
| **Fold 7** | ***Fea. 1*** | ***Fea. 2*** | **Fold 8** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.317 | -0.023 | ***Fea. 1*** | 0.306 | -0.014 |
| ***Fea. 2*** | -0.023 | 0.11 | ***Fea. 2*** | -0.013 | 0.104 |
| **Fold 9** | ***Fea. 1*** | ***Fea. 2*** | **Fold 10** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.326 | -0.017 | ***Fea. 1*** | 0.276 | -0.016 |
| ***Fea. 2*** | -0.02 | 0.099 | ***Fea. 2*** | -0.014 | 0.117 |

รูปที่ 7 Covariance Matrix ของ Class 1 ในแต่ละ Fold

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Fea. 1*** | ***Fea. 2*** | **Fold 2** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.353 | 0.022 | ***Fea. 1*** | 0.369 | -0.002 |
| ***Fea. 2*** | 0.02 | 0.097 | ***Fea. 2*** | -0.001 | 0.106 |
| **Fold 3** | ***Fea. 1*** | ***Fea. 2*** | **Fold 4** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.369 | 0.004 | ***Fea. 1*** | 0.373 | 0.01 |
| ***Fea. 2*** | 0.007 | 0.105 | ***Fea. 2*** | 0.012 | 0.105 |
| **Fold 5** | ***Fea. 1*** | ***Fea. 2*** | **Fold 6** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.389 | 0.005 | ***Fea. 1*** | 0.387 | 0 |
| ***Fea. 2*** | 0.005 | 0.107 | ***Fea. 2*** | 0.001 | 0.104 |
| **Fold 7** | ***Fea. 1*** | ***Fea. 2*** | **Fold 8** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.366 | 0.001 | ***Fea. 1*** | 0.338 | 0.02 |
| ***Fea. 2*** | 0.001 | 0.103 | ***Fea. 2*** | 0.02 | 0.096 |
| **Fold 9** | ***Fea. 1*** | ***Fea. 2*** | **Fold 10** | ***Fea. 1*** | ***Fea. 2*** |
| ***Fea. 1*** | 0.36 | -0.002 | ***Fea. 1*** | 0.383 | -0.012 |
| ***Fea. 2*** | -0.005 | 0.097 | ***Fea. 2*** | -0.01 | 0.095 |

รูปที่ 8 Covariance Matrix ของ Class 1 ในแต่ละ Fold

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold 1** | ***Predict*** | | **Acc** | **Fold 2** | ***Predict*** | | **Acc** |
| ***Actual*** | 6 | 4 | 50 | ***Actual*** | 6 | 2 | 65 |
| 6 | 4 | 5 | 7 |
| **Fold 3** | ***Predict*** | | **Acc** | **Fold 4** | ***Predict*** | | **Acc** |
| ***Actual*** | 7 | 5 | 60 | ***Actual*** | 7 | 3 | 50 |
| 3 | 5 | 7 | 3 |
| **Fold 5** | ***Predict*** | | **Acc** | **Fold 6** | ***Predict*** | | **Acc** |
| ***Actual*** | 5 | 6 | 45 | ***Actual*** | 7 | 2 | 65 |
| 5 | 4 | 5 | 6 |
| **Fold 7** | ***Predict*** | | **Acc** | **Fold 8** | ***Predict*** | | **Acc** |
| ***Actual*** | 7 | 4 | 65 | ***Actual*** | 8 | 5 | 65 |
| 3 | 6 | 2 | 5 |
| **Fold 9** | ***Predict*** | | **Acc** | **Fold 10** | ***Predict*** | | **Acc** |
| ***Actual*** | 6 | 3 | 55 | ***Actual*** | 4 | 3 | 30 |
| 6 | 5 | 11 | 2 |
| ***Mean Acc.*** | | 55 | | | | | |

รูปที่ 9 Confusion Matrix Test 2 ในแต่ละ fold

จาก**รูปที่ 9** สำหรับการทดลอง 2 พบว่า มีความแม่นยำเพียงแค่ 55% โดย Fold ที่ 10 มีความแม่นยำต่ำที่สุดเพียงแค่ 30% เท่านั้น

1. **Analyze the experiment**

จากการทดลองพบว่า การทดลองที่ 1 มีความแม่นยำโดยเฉลี่ยถึง 90% และการทดลองที่ 2 มีความแม่นยำ 55% ซึ่งสามารถสังเกตได้จาก ค่าเฉลี่ยของ Feature ในแต่ละ Folds ในการทดลองที่ 1 พบว่า Feature3 และ Feature4 มีค่าที่ต่างกันเห็นได้อย่างชัดเจนในแต่ละ Class ลำดับถัดมาคือ Feature1 มีค่าต่างกันเพียงเล็กน้อย และลำดับสุดท้าย Feature2 มีค่าเฉลี่ยที่แทบจะคล้ายคลึงกัน ดังนั้น Feature ที่เหมาะสมคือ Feature3 และ Feature4 ซึ่งการทดลองที่ 2 ผู้ทดลองได้ทดลองใช้ Feature1 และ Feature2 จึงทำให้ผลลัพธ์ออกมามีค่าที่ไม่แม่นยำ

1. **Appendix**

**import** numpy **as** np

**from** numpy **import** linalg **as** LA

**import** pandas **as** pd

# mean

**def** mean**(**x**):** **return** np**.round(**x**.**mean**(**axis**=**0**),**3**)**

# std

**def** std**(**x**):** **return** np**.round(**x**.**std**(**axis**=**0**),**3**)**

# covarian-matrix

**def** cov\_matrix**(**x**):**

fact **=** x**.**shape**[**0**]** **-** 1

**return** np**.round(**np**.**dot**((**x**-**mean**(**x**)).**T**,(**x**-**std**(**x**)))\*(**1**/**fact**),**3**)**

# multivariate normal distribution

**def** multi\_distribution**(**X**,**cov**,**mean**):**

const **=** **((**2**\***np**.**pi**)\*\*(**cov**.**shape**[**1**]/**2**))**

cov\_norm **=** LA**.**norm**(**cov**)\*\*(**0.5**)**

exp **=** np**.**array**(list(map(lambda** x**:** np**.**exp**(-**0.5**\***np**.**dot**(**np**.**dot**((**x**-**mean**),**LA**.**inv**(**cov**)),(**x**-**mean**).**T**)),**X**)))**

**return** **((**1**/(**const**\***cov\_norm**))\***exp**)**

# cross\_validations

**def** cross\_validations\_split**(**shape**,**folds**):**

fold\_size **=** **int(**shape **\*** folds**/**100**)**

k **=** 0

index **=** **[]**

**for** i **in** **range(**1**,**folds**+**1**):**

index**.**append**([**k**,**i**\***fold\_size**])** **if** **(**i **<** folds**)** **else** index**.**append**([**k**,**shape**])**

k **=** i**\***fold\_size

**return** index

# probability of Wi

**def** prob\_of\_p**(**n**,**N**):**

**return** n**/**N

# for 2 classes

**def** bayes\_rules**(**f1**,**f2**,**p1**,**p2**):**

likelihood\_ratio **=** f1**/**f2

threshold **=** p2**/**p1

decision\_matrix **=** **(**likelihood\_ratio **>** threshold**)**

**return** np**.**where**(**decision\_matrix**,**np**.**float64**(**1**),**np**.**float64**(**2**)).**reshape**(-**1**)**

# confusion matrix

**def** confusion\_matrix**(**y\_pred**,**y\_true**,**err **=** **False):**

**if** y\_true**.**shape **!=** y\_pred**.**shape **:** **return**

**def** \_condition**(**y\_pred**,**y\_true**):**

**if** y\_pred **==** y\_true **and** y\_true **==** 1**:**

**return** "TN"

**elif** y\_pred **!=** y\_true **and** y\_true **==** 2**:**

**return** "FP"

**elif** y\_pred **!=** y\_true **and** y\_true **==** 1**:**

**return** "FN"

**return** "TP"

matrix **=** np**.**array**([[**0**,** 0**],** **[**0**,** 0**]])**

**for** i **in** **range(**y\_true**.**shape**[**0**]):**

result **=** \_condition**(**y\_pred**[**i**],**y\_true**[**i**])**

**if** result **==** "TN"**:**

matrix**[**0**][**0**]** **+=** 1

**elif** result **==** "FN"**:**

matrix**[**0**][**1**]** **+=** 1

**elif** result **==** "FP"**:**

matrix**[**1**][**0**]** **+=** 1

**else:**

matrix**[**1**][**1**]** **+=** 1

**if** err**:**

**return** matrix**,**100**-(**matrix**[**0**][**0**]+**matrix**[**1**][**1**])\***100**/**y\_true**.**shape**[**0**]**

**return** matrix

**def** preprocess\_data**(**data**,**i**,**j**):**

population **=** np**.**concatenate**((**data**[:**i**],**data**[**j**:]))**

samples **=** data**[**i**:**j**]**

x\_class1 **=** population**[**population**[:,-**1**]** **==** 1**][:,:-**1**]**

x\_class2 **=** population**[**population**[:,-**1**]** **==** 2**][:,:-**1**]**

# calculate P(Wi)

p1 **=** prob\_of\_p**(**population**[**population**[:,-**1**]** **==** 1**][:,:-**1**].**shape**[**0**],**population**.**shape**[**0**])**

p2 **=** prob\_of\_p**(**population**[**population**[:,-**1**]** **==** 2**][:,:-**1**].**shape**[**0**],**population**.**shape**[**0**])**

# calculate COV(Wi)

cov\_1 **=** cov\_matrix**(**x\_class1**)**

cov\_2 **=** cov\_matrix**(**x\_class2**)**

# calculate mean(Wi)

mean\_1 **=** mean**(**x\_class1**)**

mean\_2 **=** mean**(**x\_class2**)**

pre\_data **=** **{**

'population' **:** population**,**

'x\_sample' **:** samples**[:,:-**1**],**

'x\_class1' **:** x\_class1**,** # separate the data to class 1

'x\_class2' **:** x\_class2**,** # separate the data to class 2

'p1'**:** p1**,**

'p2'**:** p2**,**

'y\_sample'**:** samples**[:,-**1**],**

'cov1'**:** cov\_1**,**

'cov2'**:** cov\_2**,**

'mean1'**:** mean\_1**,**

'mean2'**:** mean\_2**,**

**}**

**return** pre\_data

# main

**if** \_\_name\_\_ **==** "\_\_main\_\_"**:**

# split features and classes to two classes

data **=** np**.**genfromtxt**(**'TWOCLASS.csv'**,**delimiter**=**','**)[**1**:,:]**

np**.**random**.**shuffle**(**data**)** # shuffle data

k **=** 1

**for** i**,**j **in** cross\_validations\_split**(**data**.**shape**[**0**],**10**):**

# \* --------------- preprocess data ---------------

x1 **=** preprocess\_data**(**data**,**i**,**j**)** # for test 1

x2 **=** preprocess\_data**(**data**[:,[**0**,**1**,-**1**]],**i**,**j**)** # for test 2

# calculate multivariate normal distribution test 1

fx1\_1 **=** multi\_distribution**(**x1**[**'x\_sample'**],**x1**[**'cov1'**],**x1**[**'mean1'**])**

fx1\_2 **=** multi\_distribution**(**x1**[**'x\_sample'**],**x1**[**'cov2'**],**x1**[**'mean2'**])**

# calculate multivariate normal distribution test 2

fx2\_1 **=** multi\_distribution**(**x2**[**'x\_sample'**],**x2**[**'cov1'**],**x2**[**'mean1'**])**

fx2\_2 **=** multi\_distribution**(**x2**[**'x\_sample'**],**x2**[**'cov2'**],**x2**[**'mean2'**])**

# evaluate test 1

y\_pred1 **=** bayes\_rules**(**fx1\_1**,**fx1\_2**,**x1**[**'p1'**],**x1**[**'p2'**])**

y\_true1 **=** x1**[**'y\_sample'**]**

# evaluate test 1

y\_pred2 **=** bayes\_rules**(**fx2\_1**,**fx2\_2**,**x2**[**'p1'**],**x2**[**'p2'**])**

y\_true2 **=** x2**[**'y\_sample'**]**

**print(**"############### K="**,** k **,**" #################"**)**

k**+=**1