รายงาน
เรื่อง Airline Passenger Satisfaction

จัดทำโดย
นายภีมภัช พจน์สุนทร

เสนอ
ผศ.ดร.ปกรณ์ ลี้สุทธิพรชัย

โครงงานนี้เป็นส่วนหนึ่งของรายวิชา
การเรียนรู้ของเครื่องจักรและการทำเหมืองข้อมูลเชิงประยุกต์
(CS345 Practical Machine Learning and Data Mining)
ภาคเรียนที่ 1 ปีการศึกษา 2564
มหาวิทยาลัยธรรมศาสตร์ ศูนย์รังสิต

สารบัญ

สารบัญภาพ

หน้า

บทนำ

ในรายงานฉบับนี้จะเป็นการนำชุดข้อมูล Airline Passenger Satisfaction มาทำการ Classification หรือ จำแนกข้อมูล โดยทำผ่าน 3 เครื่องมือ ได้แก่ 1. RapidMiner Studio 2. Weka และ 3. Python ซึ่งวิธีการทำ Classification จะใช้ Model ในการคำนวณและจำแนกข้อมูลออกมา ภายในรายงานฉบับนี้จะใช้ Model ที่นิยมใช้กัน ได้แก่ 1.KNN (k-nearest neighbors) 2.SVM (Support vector machines) และ 3.Neural Network ซึ่งจะไม่ได้ใช้ Model ทั้งมาทำ Classification แต่จะเลือก Model ที่มีประสิทธิภาพมากที่สุดมาทำ โดยวิธีการเลือก Model ที่มีประสิทธิภาพมากที่สุดคือการใช้เทคนิค Cross Validation หาก Model ไหนมีค่าประสิทธิภาพที่ใช้เทคนิค Cross Validation เยอะที่สุด จะนำ Model ดังกล่าวไปทำ Classification เพื่อดูประสิทธิภาพการทำงานต่อไป

หากสามารถจำแนกข้อมูล หรือ Classification ข้อมูลได้สำเร็จ สามารถนำผลลัพธ์มาวิเคราะห์ประสิทธิภาพของสายการบินนั้น ๆ ว่าผู้โดยสารส่วนใหญ่พึงพอใจต่อการใช้สายการบินดังกล่าวหรือไม่ อีกทั้งยังสามารถนำผลลัพธ์ไปปรับปรุงบริการของทางสายการบินเองได้อีก เพื่อให้ผู้โดยสารรู้สึกพึงพอใจในการใช้บริการของสายการบินดังกล่าว

บทที่ 1
คำอธิบายเกี่ยวกับข้อมูล

ชุดข้อมูล Airline Passenger Satisfaction [1] เป็นชุดข้อมูลที่สำรวจความพึงพอใจของผู้โดยสารสายการบินสหรัฐ โดยข้อมูลจะมีแบ่งเป็น test และ train ซึ่งรวมกันแล้วจะมีข้อมูลทั้งหมด 129,880 แถว (row) และแต่ละแถวนั้นจะมีคอลัมน์ (column) ทั้งหมด 25 คอลัมน์ โดยข้อมูลในแต่ละคอลัมน์มีดังนี้

1. id คือ ข้อมูล id ของผู้โดยสารแต่ละคน (เป็นข้อมูลประเภท Integer ไม่มี Missing)
2. Gender คือ เพศของผู้โดยสาร แบ่งเป็น เพศหญิง (Female) และ เพศชาย (Male) (เป็นข้อมูลประเภท Nominal ไม่มี Missing)
3. Customer Type คือ ประเภทของผู้โดยสาร แบ่งเป็น ผู้โดยสารประจำ (Loyal customer) และ ผู้โดยสารทั่วไป (disloyal customer) (เป็นข้อมูลประเภท Nominal ไม่มี Missing)
4. Age คือ อายุของผู้โดยสาร มีช่วยอยู่ที่ 7 ถึง 85 ปี (เป็นข้อมูลประเภท Integer ไม่มี Missing)
5. Type of Travel คือ เป้าหมายการเดินทางของผู้โดยสาร แบ่งเป็น เดินทางส่วนตัว (Personal Travel) และ เดินทางเพื่อทำธุรกิจ (Business Travel) (เป็นข้อมูลประเภท Nominal ไม่มี Missing)
6. Class คือ ชั้นโดยสารภายในเครื่องบินที่ผู้โดยสารเลือก แบ่งเป็น ชั้นธุรกิจ (Business) ชั้นประหยัด (Eco) และชั้นประหยัด พลัส (Eco Plus) (เป็นข้อมูลประเภท Nominal ไม่มี Missing)
7. Flight distance คือ ระยะทางการบิน มีช่วงอยู่ที่ 31 ถึง 4,983 กิโลเมตร (เป็นข้อมูลประเภท Integer ไม่มี Missing)

(ช่วงข้อที่ 8 จนถึงข้อที่ 21 จะเป็นระดับความพึงพอใจของผู้โดยสาร จะมีช่วงอยู่ที่ 0 ถึง 5 โดย 0 คือ ใช้ไม่ได้)

8. Inflight wifi service คือ ระดับความพึงพอใจของบริการ wifi บนเครื่องบิน (เป็นข้อมูลประเภท Integer ไม่มี Missing)
9. Departure/Arrival time convenient คือ ระดับความพึงพอใจในการออกเดินทางจนถึงปลายทาง (เป็นข้อมูลประเภท Integer ไม่มี Missing)

10. Ease of Online booking คือ ระดับความพึงพอใจของการจองออนไลน์ (เป็นข้อมูลประเภท Integer ไม่มี Missing)
11. Gate location คือ ระดับความพอใจของที่ตั้งเกท (เป็นข้อมูลประเภท Integer ไม่มี Missing)
12. Food and drink คือ ระดับความพึงพอใจของอาหารและเครื่องดื่ม (เป็นข้อมูลประเภท Integer ไม่มี Missing)
13. Online boarding คือ ระดับความพึงพอใจของการขึ้นเครื่องออนไลน์ (เป็นข้อมูลประเภท Integer ไม่มี Missing)
14. Seat comfort คือ ระดับความพอใจของความสบายของที่นั่ง (เป็นข้อมูลประเภท Integer ไม่มี Missing)
15. Inflight entertainment คือ ระดับความพึงพอใจของความบันเทิงบนเครื่องบิน (เป็นข้อมูลประเภท Integer ไม่มี Missing)
16. On-board service คือ ระดับความพึงพอใจของบริการออนบอร์ด (เป็นข้อมูลประเภท Integer ไม่มี Missing)
17. Leg room service คือ ระดับความพึงพอใจของที่รองขา (เป็นข้อมูลประเภท Integer ไม่มี Missing)
18. Baggage handling คือ ระดับความพึงพอใจในการจัดการสัมภาระ (เป็นข้อมูลประเภท Integer ไม่มี Missing)
19. Check-in service คือ ระดับความพึงพอใจของบริการเช็คอิน (เป็นข้อมูลประเภท Integer ไม่มี Missing)
20. Inflight service คือ ระดับความพึงพอใจของบริการบนเครื่องบิน (เป็นข้อมูลประเภท Integer ไม่มี Missing)
21. Cleanliness คือ ระดับความพึงพอใจของความสะอาด (เป็นข้อมูลประเภท Integer ไม่มี Missing)

22. Departure Delay in Minutes คือ เวลาที่ล่าช้าเมื่อออกเดินทาง (หน่วยนาที) (เป็นข้อมูลประเภท Integer ไม่มี Missing)

23. Arrival Delay in Minutes คือ เวลาที่ล่าช้าเมื่อถึงสถานที่ปลายทาง (หน่วยนาที) (เป็นข้อมูลประเภท Real มี Missing อยู่ 310)

24. Satisfaction คือ ระดับความพึงพอใจของสายการบิน แบ่งเป็น พึงพอใจ (satisfied) และเฉย ๆ หรือ ไม่พอใจ (neutral or dissatisfaction) (เป็นข้อมูลประเภท Nominal ไม่มี Missing)

ข้อมูลมาจากเว็บไซต์ kaggle.com ที่มีความน่าเชื่อถือระดับหนึ่ง แต่ไม่มีที่มาของข้อมูลว่ามาจากสายการบินอะไร แต่อาจเป็นเพราะการเปิดเผยข้อมูลของผู้โดยสารอาจทำให้เกิดปัญหากับสายการบินนั้น ซึ่งอาจส่งผลให้ข้อมูลนี้ไม่มีที่มาก็เป็นได้ แต่ข้อมูลชุดนี้มีผู้ใช้งานจำนวนมาก ดังนั้นข้อมูลชุดนี้แม้อาจไม่มีความน่าเชื่อถือเท่าที่ควร แต่สามารถไว้วางใจได้เพราะมีผู้ใช้งานจำนวนมาก

จากข้อมูลที่มี ทางผู้จัดทำจะนำข้อมูลไปทำ Classification หรือ การจำแนกประเภท โดยจะจำแนกตาม Satisfaction คือจากข้อมูลที่รวบรวมมา สามารถจำแนกข้อมูลเหล่านี้ว่ามีระดับความพึงพอใจได้หรือไม่ และถ้าทำได้ Model ไหนที่มีประสิทธิภาพมากที่สุด

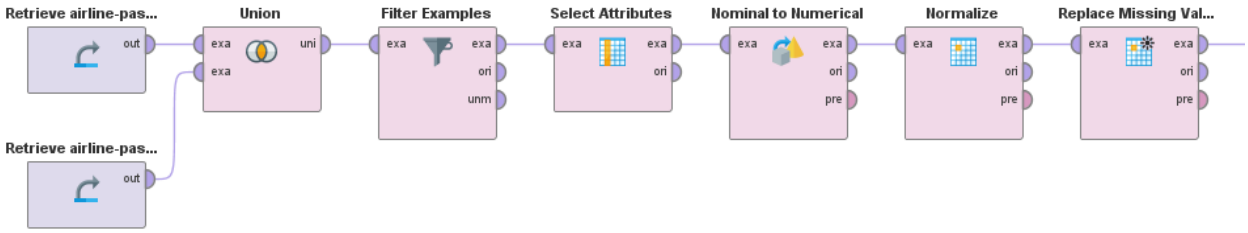
บทที่ 2
ขั้นตอนและกระบวนการทำงาน

2.1 ขั้นตอน Preprocess

ชุดข้อมูล Airline Passenger Satisfaction จะมี 2 ไฟล์ ได้แก่ 1.test.csv และ 2.train.csv ผู้จัดทำตัดสินใจที่จะรวบทั้ง 2 ไฟล์เข้าได้กันเพื่อที่ในขั้นตอนการจัดการข้อมูลสามารถทำได้พร้อมกัน และค่อยนำไปแยกเป็น train และ test ในภายหลัง

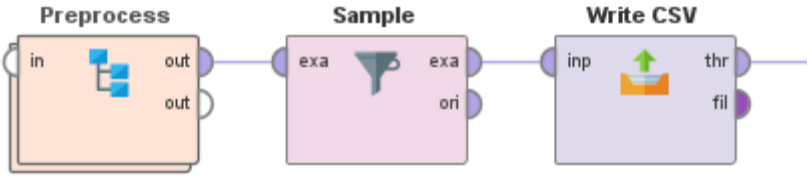
ในการเตรียมข้อมูลเพื่อที่จะนำไปทำ Classification ในภาคหลังนั้น ผู้จัดทำได้เลือกโปรแกรม RapidMiner Studio ในการจัดการเนื่องจากสะดวกและใช้ง่ายต่อการจัดการข้อมูล โดยจะจัดเตรียมข้อมูลเป็น 2 ชุดไว้สำหรับทำ Classification มีขั้นตอนการจัดการข้อมูลดังนี้

1. ชุดข้อมูลสำหรับทำ Classification แบบ 10,000 แถว



ภาพที่ 2.1 ขั้นตอนการทำ preprocess ของ Classification

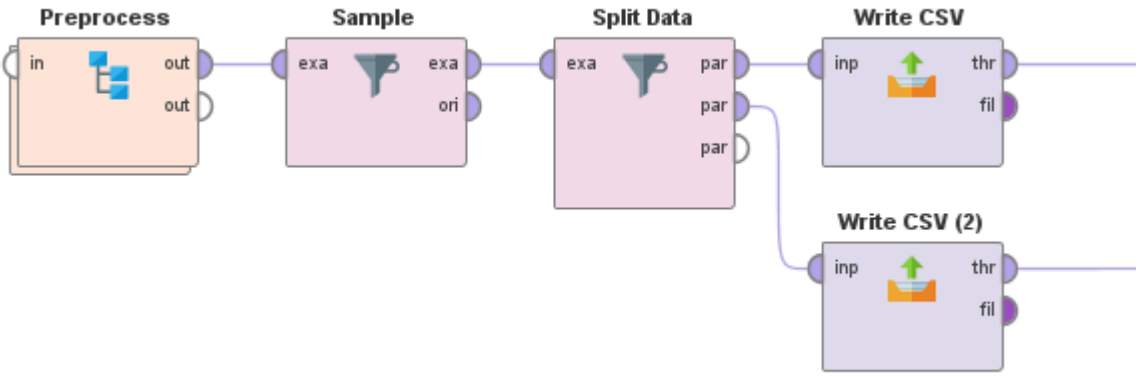
จากภาพที่ 2.1 จะนำไฟล์ test และ train มา union กัน เพื่อรวมเป็นข้อมูลเดียว จากนั้นทำการ Filter Examples โดยข้อมูลที่ Filter คือข้อมูล Arrival Delay in Minutes โดยข้อมูลนี้ในชุดข้อมูลจะมีค่าที่ไม่ได้ระบุอยู่ จึงจำเป็นต้อง Filter ออก จากนั้นทำการเลือกข้อมูลที่จะใช้ในการไปทำ Classification ด้วย Select Attributes ซึ่งข้อมูลที่เลือกออกจะเป็น id หลังจากเลือกข้อมูลเสร็จจะทำการแปลงข้อมูลที่เป็น nominal เป็น numerical และทำการ Normalize แบบ range transformation (0 ถึง 1) และมีการทำ Replace Missing Value ตามที่ทางโปรแกรมแนะนำ



ภาพที่ 2.2 ชุดข้อมูลสำหรับทำ Classification แบบ 10,000 แถว

จากภาพที่ 2.2 คือการนำกระบวนการในภาพที่ 2.1 มา group กัน สร้างเป็น subprocess โดยตั้งชื่อว่า Preprocess จากนั้นทำการ Sample ค่ามา 10,000 เนื่องจากข้อมูลที่ให้มามีถึง 100,000 แถวและมีคอลัมน์อีก 25 คอลัมน์ หากนำข้อมูลทั้งหมดนี้ไปทำสร้างเป็น model จะใช้เวลานานมาก ดังนั้นผู้จัดทำจึงกำหนดแค่ 10,000 แถวเท่านั้น โดยการ Sample ค่านั้นมีการกำหนด local random seed ไว้เท่ากับ 1 เพื่อที่เมื่อมาสร้างไฟล์ใหม่ จะยังคงได้ข้อมูลในแถวเดิมเสมอ หลังจากนั้นนำไปสร้างเป็นไฟล์ csv โดยกำหนดตัวคั่น column เป็นเครื่องหมาย semicolon (;) และตั้งชื่อไฟล์ว่า airplane_classification_preprocess_10000 เพื่อนำไปทำ Classification ต่อไป

2. ชุดข้อมูลสำหรับทำ Classification แบบแยกเป็น train 8,000 แถว และ test 2,000 แถว



ภาพที่ 2.3 ชุดข้อมูลสำหรับทำ Classification แบบแยกเป็น train 8,000 แถว และ test 2,000 แถว

จากภาพที่ 2.3 ขั้นตอนการทำ preprocess จะเหมือนกับภาพที่ 2.1 แต่ในการแยกเป็นข้อมูล 10,000 แถว มาเป็น train 8,000 แถว และ test 2,000 แถว นั้นจะมีการเรียกใช้ split data โดยกำหนด partitions ในส่วนแรกเป็น 0.8 (แยกจาก 10,000 จะได้ 8,000) และส่วนที่สองเป็น 0.2 (แยกจาก 10,000 จะได้ 2,000) หลังจากนั้นนำไปสร้างเป็นไฟล์ csv โดยกำหนดตัวคั่น column เป็นเครื่องหมาย semicolon (;) ซึ่งส่วนบน

จะเป็นข้อมูล train จึงตั้งชื่อไฟล์ว่า airplane_classification_preprocess_train และส่วนล่างเป็นข้อมูล test จึงตั้งชื่อไฟล์ว่า airplane_classification_preprocess_test เพื่อนำไปทำ Classification ต่อไป

การจัดเตรียมชุดข้อมูลแยกสำหรับทำ Classification เพราะต้องการให้ข้อมูลที่จะนำไปใช้ในโปรแกรมอื่น ๆ มีความเหมือนกัน เพื่อที่จะได้ดูประสิทธิภาพและดูความแตกต่างของแต่ละโปรแกรม ซึ่งใน Classification ที่แยกทำเป็น 2 ชุด คือ ชุดที่มีข้อมูล 10,000 แถว กับชุดที่แยกเป็น train กับ test เพราะ ในการทำ cross validation จะเป็นการดู score ของ model ที่เลือก ซึ่งจะใช้ชุดข้อมูล 10,000 แถวในการดู score และใช้ train กับ test ในการทำ model ต่าง ๆ

จากกระบวนการทำชุดข้อมูลสำหรับนำไปทำ Classification นั้นจะไม่มีการกำจัด noise หรือ outlier ออกไป เนื่องจากผู้จัดทำได้ลองดูชุดข้อมูล ดูค่า min และค่า max ของแต่ละคอลัมน์ ซึ่งพบว่าข้อมูลมีความถูกต้อง (อาจ) ไม่มีข้อมูลไหนที่มีค่าที่แปลกเกินไป เช่น ในคอลัมน์ Age อายุที่น้อยที่สุดคือ 7 ซึ่งเป็นไปได้ที่เด็กอายุ 7 ขวบขึ้นเครื่องบิน และอายุที่มากที่สุดคือ 85 ปี ซึ่งก็มีไม่แปลกที่ผู้สูงวัยขึ้นเครื่องบิน และข้อมูลอื่น ๆ ก็เป็นไปตามความเป็นไปได้จริง ๆ ดังนั้นผู้จัดทำจึงไม่ทำกระบวนการกำจัด noise หรือ กำจัด outlier

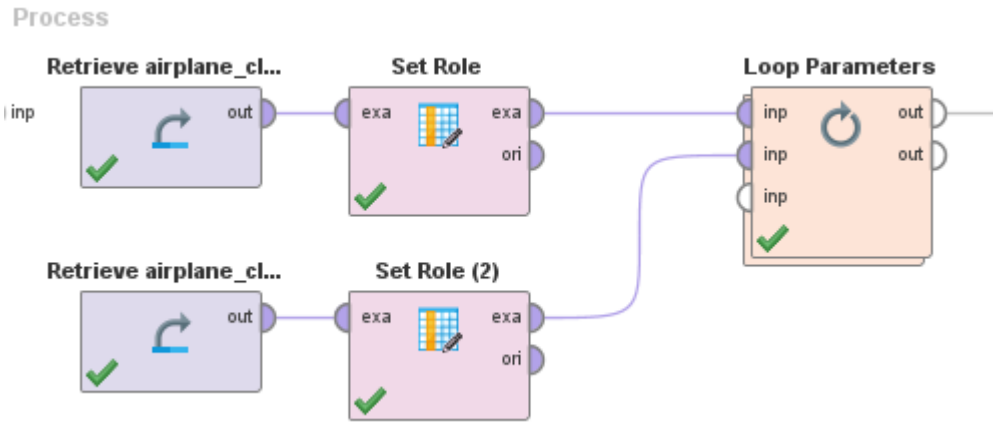
2.2 ขั้นตอน Classification

ในการทำ Classification ได้มีการเตรียมชุดข้อมูลไว้ทั้งหมด 2 ชุดนั้นคือ 1.ชุดข้อมูลแบบ 10,000 และ 2. ชุดข้อมูลแบบ train กับ test ซึ่งชุดข้อมูลที่มีนั้นเป็นมีลักษณะเป็น Class Imbalance เนื่องจาก ผลลัพธ์ของคอลัมน์ satisfaction ไม่เท่ากัน โดยผู้จัดทำจะทำ Classification ผ่านทางเครื่องมือ RapidMiner Studio และ Weka และจะมีการใช้โค้ด Python ในการทำร่วมด้วยเช่นกัน

2.2.1 RapidMiner Studio

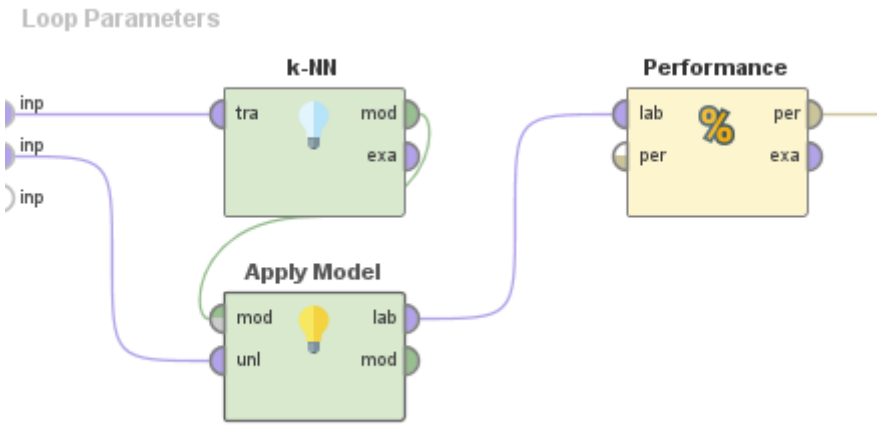
การทำ Classification นั้นจะเป็นการจำแนกประเภทของข้อมูลโดยการทำ Classification จะใช้ Model ในการคำนวณและจำแนกข้อมูลออกมา โดยแต่ละ Model จะมีการคำนวณและจำแนกแตกต่างกันออกไป Model ที่นิยมใช้กัน ได้แก่ 1.KNN (k-nearest neighbors) 2.SVM (Support vector machines) และ 3.Neural Network ซึ่งการใช้ข้อมูลมาจำแนกด้วย Model ต่าง ๆ ก็ส่งผลให้ได้ประสิทธิภาพที่แตกต่างกัน ดังนั้นในการเลือกใช้ Model มาทำ Classification จะเลือก Model ที่มีประสิทธิภาพมากที่สุด ซึ่งวิธีการหาประสิทธิภาพที่รวดเร็วนั้นคือ การทำ Cross Validation ที่จะเป็นการนำข้อมูลที่มีไปเข้า Model เพื่อดูประสิทธิภาพของ Model นั้น ๆ ดังนั้นจะเริ่มที่ Model แรก KNN

Model KNN หรือ k-nearest neighbors จะเป็นการหาข้อมูล k ตัวที่อยู่ใกล้กับจุดที่ต้องการ ซึ่ง Model นี้จำเป็นต้องระบุค่า K แต่ค่า K แต่ละค่าก็ไม่ได้ให้ประสิทธิภาพที่แม่นยำเท่ากัน ดังนั้นจึงต้องหาค่า K ที่เหมาะสมสำหรับ Model KNN โดยจะใช้ Loop Parameters ในการหา โดยจะ loop ค่า k ตั้งแต่ 1 จนถึง 50



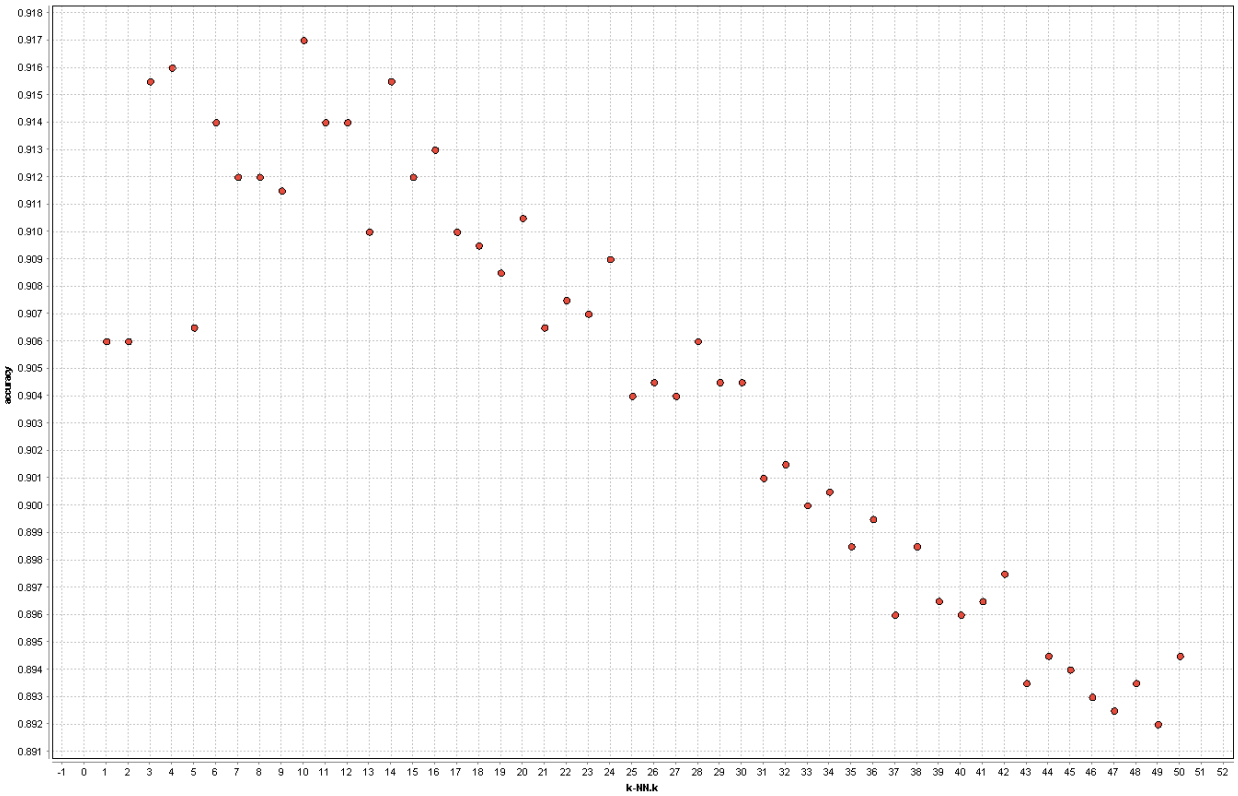
ภาพที่ 2.4 ขั้นตอนการหาค่า k (RapidMiner Studio)

จากภาพที่ 2.4 จะใช้ชุดข้อมูล train กับ test ของ Classification ในการหาค่า k หลังจากดึงข้อมูลเสร็จ จะทำการกำหนด Set Role หรือการกำหนดข้อมูล โดยจะกำหนดคอลัมน์ satisfaction เป็น label ซึ่งเป็นสิ่งที่จะให้ Model ทำนายสิ่งนี้ หลังจากนั้นทำการ Loop Parameters โดยการส่งข้อมูล train และ test เข้าไป



ภาพที่ 2.5 ขั้นตอนการหาค่า k (ต่อ) (RapidMiner Studio)

จากภาพที่ 2.5 จะใช้ข้อมูล train มาผ่าน Model KNN หรือก็คือให้ Model KNN ได้ทำการฝึกฝนผ่านชุดข้อมูล train นี้ โดยจากภาพที่ 2.4 ในขั้นตอน Loop Parameter ได้ทำการกำหนด k เริ่มที่ 1 จนถึง 50 ดังนั้น ค่า k ใน Model KNN นี้จะมีค่าตั้งแต่ 1 จนถึง 50 ซึ่งหลังจากที่ Model KNN ได้ผ่านกระบวนการฝึกฝนข้อมูลมา จะทำการ Apply Model โดยนำ Model ที่ได้มากับชุดข้อมูล test มาผ่าน Model ที่ได้ฝึกมา จากนั้นผ่านกระบวนการ Performance โดยกำหนด main criterion เป็น accuracy เพื่อดูค่าความแม่นยำหรือประสิทธิภาพ หลังจากทำ Process ทั้งหมดนี้ จะได้ผลลัพธ์มาดังนี้



ภาพที่ 2.6 กราฟแสดงความแม่นยำของค่า k ที่ 1 ถึง 50 (RapidMiner Studio)

จากภาพที่ 2.6 จะเป็นกราฟที่แสดงค่าความแม่นกับโดยสัมพันธ์กับค่า k ยิ่งค่า accuracy มีค่าเข้าใกล้ 1 มากเท่าไหร่ แปลว่าประสิทธิภาพในช่วงที่ k นั้น ๆ จะมีประสิทธิภาพมาก

Loop Parameters (50 rows, 3 columns)

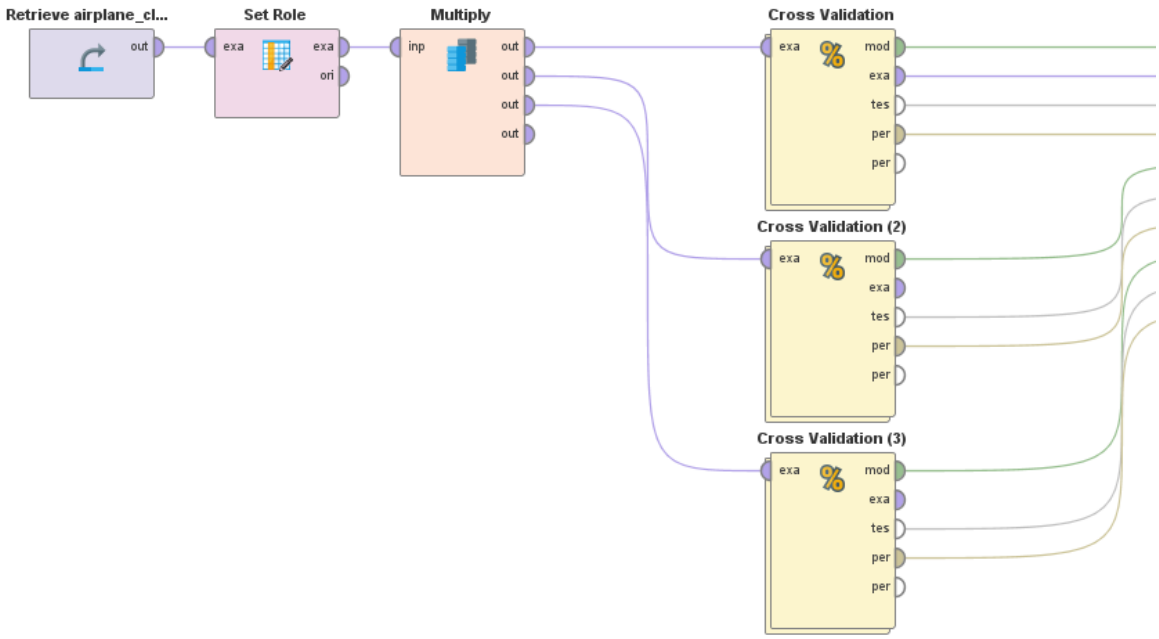
iteration	k-NN.k	accuracy ↓
10	10	0.917
4	4	0.916
14	14	0.915
3	3	0.915
6	6	0.914

ภาพที่ 2.7 ตารางแสดงความแม่นยำของค่า k ด้วยเรียงจากความแม่นยำมากสุดไปน้อยสุด (RapidMiner Studio)

จากภาพที่ 2.7 จะเป็นตารางที่บอกถึง iteration (จำนวนรอบที่วนลูป) ค่า k ที่ใช้ และ accuracy (ความแม่นยำ) จะเห็นได้ว่า ที่ iteration หรือ k ที่ 10 จะมีประสิทธิภาพสูงที่สุดนั้นคือ 0.917 ดังนั้นในการทำ Cross Validation จะเลือกใช้ k ที่ 10 ในการทำ

ในการทำ Cross Validation ของ Model SVM (Support vector machines) ไม่ได้จำเป็นต้องมีการกำหนดค่าอะไรเหมือน KNN ดังนั้นจึงไม่มีการหาค่าสำหรับ SVM แต่ Model Neural Network เป็นการจำลองคล้าย โครงข่ายสมองของมนุษย์ ซึ่งจำเป็นต้องมีการกำหนด hidden layers (จำนวนชั้นของ Neural Network ที่จะใช้ในการคำนวณ) ซึ่งวิธีการหาว่าแต่ละชั้นจะใช้จำนวนเท่าไหร่นั้น วิธีที่ง่ายที่สุดคือนำจำนวนคอลัมน์มาหาร 2 แล้วบวกด้วย 1 ซึ่งจำนวนของคอลัมน์ที่จะนำมาใช้จำแนกประเภทออกเป็น satisfaction นั้นจะมีอยู่ด้วยกันทั้งหมด 27 คอลัมน์ ดังนั้น 27 หาร 2 จะเท่ากับ 13.5 บวกด้วย 1 เท่ากับ 14.5 ปัดขึ้นจะได้ 15 ดังนั้นในชั้นแรก จะมีขนาด 15 ซึ่ง Model Neural Network สามารถทำได้หลายชั้น ผู้จัดทำเห็นว่าจำนวนชั้นแรกยังเยอะอยู่ จึงทำการกำหนดให้ Model Neural Network ใช้ 2 ชั้น ซึ่งชั้นที่สอง หลังจากคำนวณ ชั้นที่สองจะมีขนาด 9

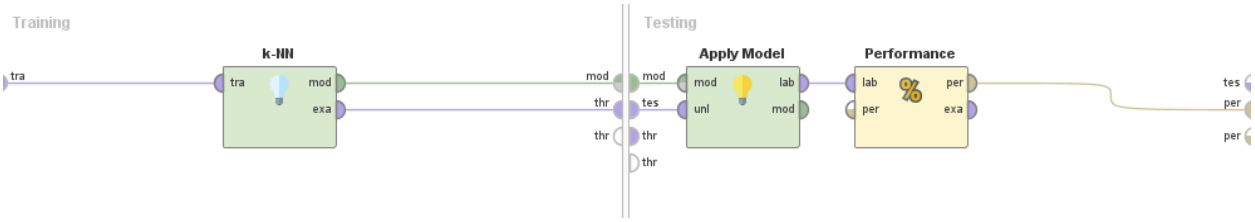
เมื่อกำหนดค่าที่จำเป็นต้องใช้หมดแล้ว ก็จะเริ่มทำ Cross Validation



ภาพที่ 2.8 การทำ Cross Validation (RapidMiner Studio)

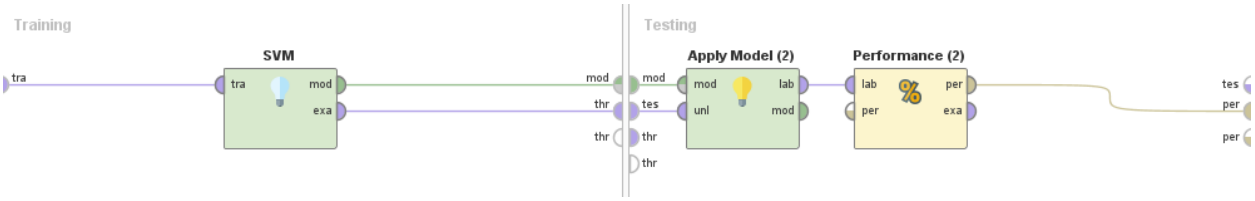
จากภาพที่ 2.8 จะใช้ชุดข้อมูล Classification แบบ 10,000 แถวในการทำ Cross Validation หลังจากทำการดึงข้อมูล จะเป็นการ Set Role โดยเลือกคอลัมน์ Satisfaction เป็น label ในการจำแนกประเภท

จากนั้นจะใช้ Multiply เพื่อทำ Cross Validation กับ Model ทั้ง 3 โดยจะส่งข้อมูลทั้งหมดที่มีคอลัมน์ Satisfaction เป็น label เข้าไปหา performance ของแต่ละ Model



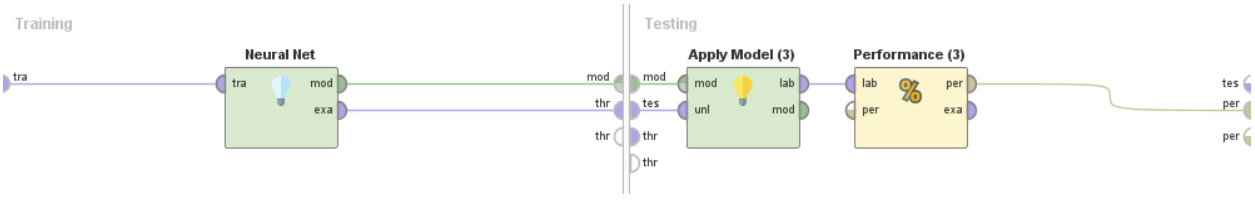
ภาพที่ 2.9 การทำ Cross Validation ด้วย Model KNN (RapidMiner Studio)

จากภาพที่ 2.9 หลังจากส่งข้อมูลเข้ามาใน Cross Validation จะนำข้อมูลที่ได้รับมา Training กับ Model KNN โดย KNN จำเป็นต้องมีการกำหนดค่า K ซึ่งก่อนหน้านี้ได้มีการหาจำนวน K แล้วว่า จำนวน K ที่เท่าไหร่ที่ให้ค่าประสิทธิภาพดีที่สุด ซึ่งได้ผลลัพธ์ออกมาเป็น 10 ดังนั้นจึงกำหนด K เท่ากับ 10 หลังจากที่ Training เสร็จ จะทำการส่ง Model ที่ได้มีการ Train แล้วไป Apply Model และหา Performance ต่อไป



ภาพที่ 2.10 การทำ Cross Validation ด้วย Model SVM (RapidMiner Studio)

จากภาพที่ 2.10 หลังจากส่งข้อมูลเข้ามาใน Cross Validation จะนำข้อมูลที่ได้รับมา Training กับ Model SVM จากนั้นเมื่อ Training เสร็จ จะทำการส่ง Model ที่ได้มีการ Train แล้วไป Apply Model และหา Performance ต่อไป



ภาพที่ 2.11 การทำ Cross Validation ด้วย Model Neural Network (RapidMiner Studio)

จากภาพที่ 2.11 หลังจากส่งข้อมูลเข้ามาใน Cross Validation จะนำข้อมูลที่ได้รับมา Training กับ Model Neural Network โดยจะกำหนด hidden layer หรือชั้นแรกมีขนาด 15 และชั้นสองมีขนาด 9 พร้อม

ทั้งกำหนดจำนวนรอบ (training cycles) ที่ 200 (default) และกำหนด learning rate ที่ 0.01 หลังจากที่ Training เสร็จ จะทำการส่ง Model ที่ได้มีการ Train แล้วไป Apply Model และหา Performance ต่อไป

accuracy: 91.04% +/- 1.04% (micro average: 91.04%)

	true neutral or dissatisfied	true satisfied	class precision
pred. neutral or dissatisfied	5337	671	88.83%
pred. satisfied	225	3767	94.36%
class recall	95.95%	84.88%	

ภาพที่ 2.12 ประสิทธิภาพของ Model KNN (RapidMiner Studio)

accuracy: 79.70% +/- 1.28% (micro average: 79.70%)

	true neutral or dissatisfied	true satisfied	class precision
pred. neutral or dissatisfied	4573	1041	81.46%
pred. satisfied	989	3397	77.45%
class recall	82.22%	76.54%	

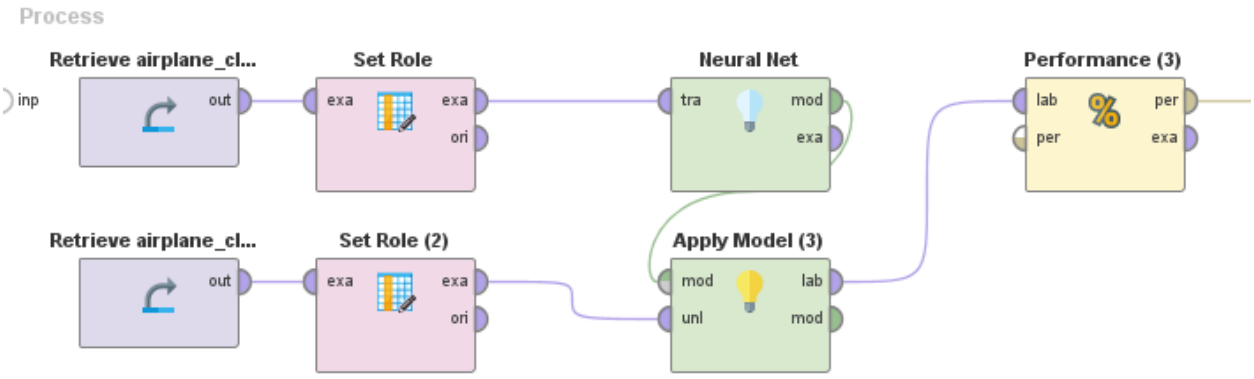
ภาพที่ 2.13 ประสิทธิภาพของ Model SVM (RapidMiner Studio)

accuracy: 93.72% +/- 0.47% (micro average: 93.72%)

	true neutral or dissatisfied	true satisfied	class precision
pred. neutral or dissatisfied	5270	336	94.01%
pred. satisfied	292	4102	93.35%
class recall	94.75%	92.43%	

ภาพที่ 2.14 ประสิทธิภาพของ Model Neural Network (RapidMiner Studio)

จากภาพที่ 2.12 จะเป็นประสิทธิภาพของ KNN ภาพที่ 2.13 จะเป็นประสิทธิภาพของ SVM และภาพที่ 2.14 จะเป็นประสิทธิภาพของ Neural Network จะเห็นได้ว่าประสิทธิภาพของ Model Neural Network ทีประสิทธิภาพดีที่สุด โดนมีประสิทธิภาพอยู่ที่ 93.72% ดังนั้นในการทำ Classification จึงเลือกใช้ Model Neural Network ในการทำ



ภาพที่ 2.15 ใช้ชุดข้อมูล train และ test กับ Model Neural Network (RapidMiner Studio)

จากภาพที่ 2.15 จะใช้ชุดข้อมูลแบบ train กับ test หลังจากที่ดึงข้อมูลเสร็จจะทำการ Set Role โดย set คอลัมน์ satisfaction เป็น label จากนั้นจะนำชุดข้อมูล train ไปผ่าน Model Neural Network โดยมีการกำนหด hidden layer ชั้นแรกขนาด 15 และชั้นสองขนาด 9 พร้อมทั้งกำหนดจำนวนรอบ (training cycles) ที่ 200 (default) และกำหนด learning rate ที่ 0.01 หลังจาก Model ฝึกฝนเสร็จ จะทำการ Apply Model โดยใช้ Model ที่ได้ฝึกมาและใช้ชุดข้อมูล test เมื่อ apply เสร็จจะทำการแสดงประสิทธิภาพของ Model

accuracy: 93.20%

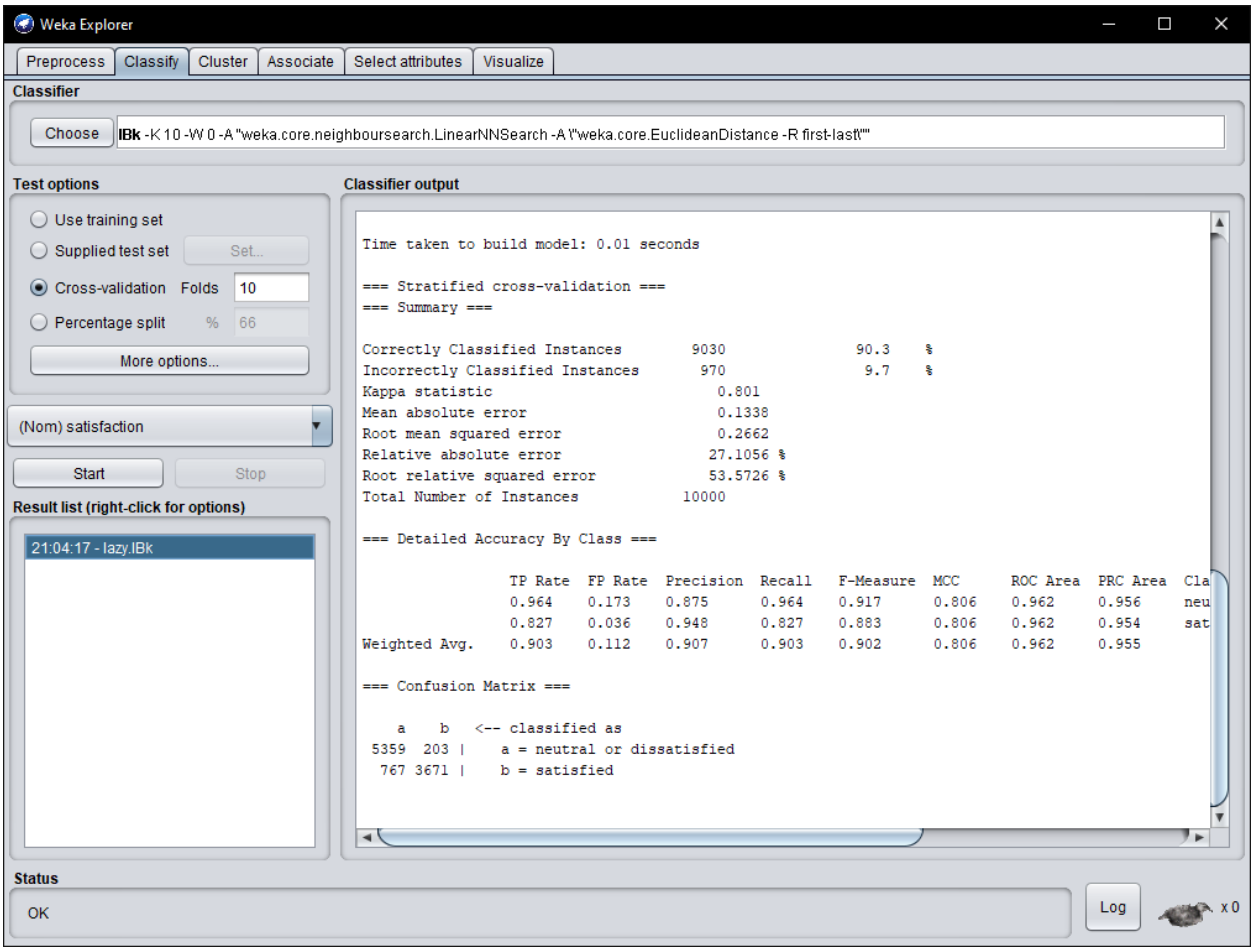
	true neutral or dissatisfied	true satisfied	class precision
pred. neutral or dissatisfied	1030	70	93.64%
pred. satisfied	66	834	92.67%
class recall	93.98%	92.26%	

ภาพที่ 2.16 ประสิทธิภาพของ Model Neural Network (RapidMiner Studio)

จากภาพที่ 2.16 Model Neural Network จะมีประสิทธิภาพอยู่ที่ 93.20% โดย Model จะทำการคาดเดาความพึงพอใจของผู้โดยสาร (satisfaction) โดยอ้างอิงจากสิ่งที่ได้ฝึกฝนมา พบว่า จากผู้โดยสารที่รู้สึกเฉย ๆ หรือไม่พอใจ Model ทายถูกถึง 1,030 จากทั้งหมด 1,096 แถว ทายผิดเพียงแค่ 66 แถว และผู้โดยสารที่รู้สึกพึงพอใจ Model ทายถูกถึง 834 จากทั้งหมด 904 ทายผิดเพียงแค่ 70 แถว แต่จากการทำ Cross Validation ซึ่งให้ประสิทธิภาพอยู่ที่ 93.72% จะเห็นได้ว่าประสิทธิภาพลดลงเล็กน้อย จึงกล่าวได้ว่า Model Neural Network ที่ได้ฝึกมานั้นมีประสิทธิภาพดี แต่ Model อาจ fit เล็กน้อยเนื่องจากค่าประสิทธิภาพได้ลดลง

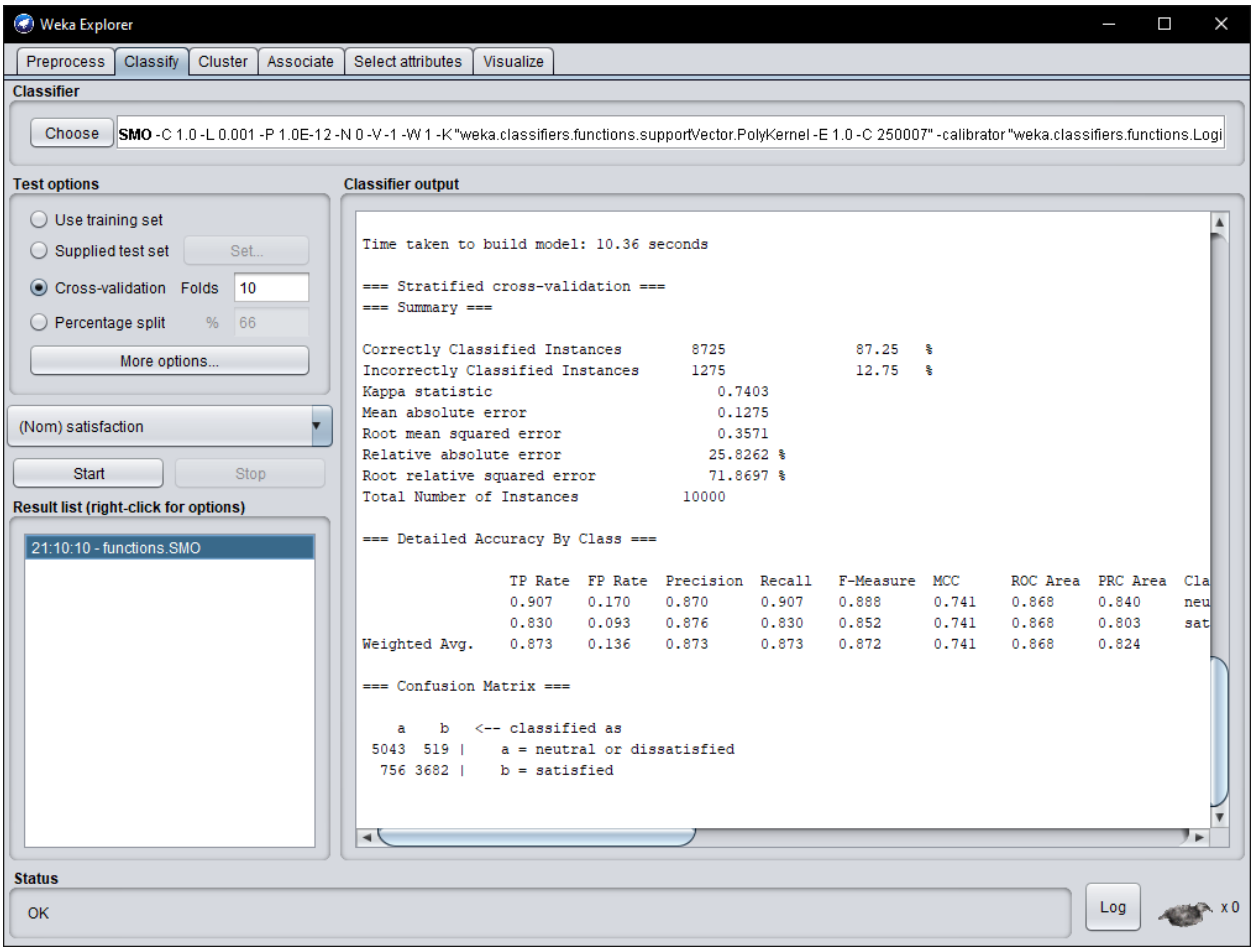
2.2.2 Weka

ถัดจาก RapidMiner Studio ผู้จัดทำเลือกใช้ Weka มาทำต่อ โดยกระบวนการทำจะคล้ายกับ RapidMiner Studio นั้นคือ หา Model ที่มีประสิทธิภาพมากที่สุดด้วยการทำ Cross Validation โดยใช้ชุดข้อมูล 10,000 แถว และเมื่อได้ Model ที่มีประสิทธิภาพมากที่สุดแล้ว จะนำ Model นั้นมา Train ด้วยชุดข้อมูล train และ Test ด้วยชุดข้อมูล test จะเริ่มต้นกันด้วย Model KNN



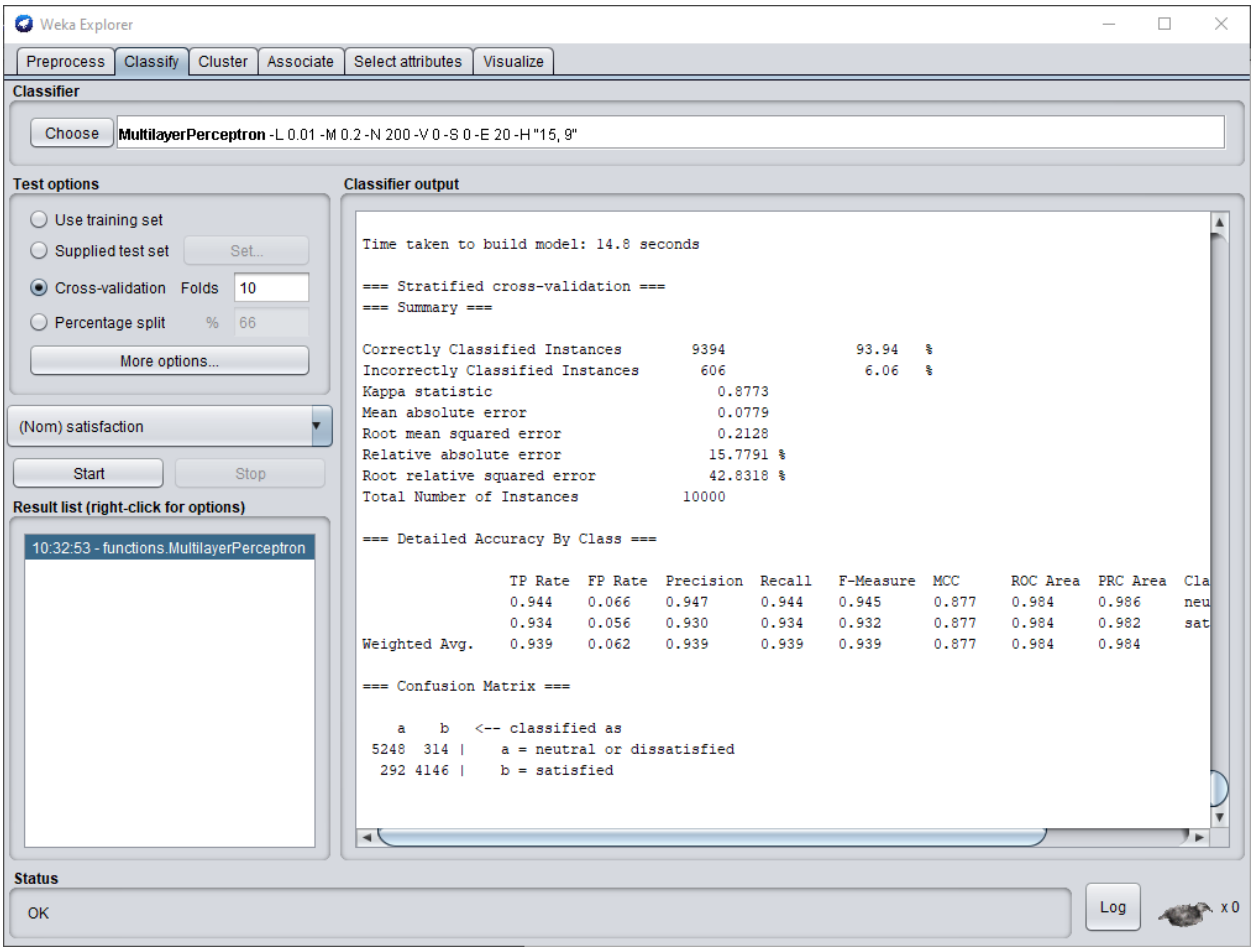
ภาพที่ 2.17 การทำ Cross Validation ด้วย Model KNN (Weka)

จากภาพที่ 2.17 ทำการเลือก Classifier เป็น Model KNN หรือใน Weka จะเป็น IBK โดยทำการกำหนดค่า k ตามผลลัพธ์ในภาพที่ 2.9 นั้นคือ k เท่ากับ 10 จากนั้นเลือก Test options เป็น Cross-validation ที่ Folds เท่ากับ 10 จากนั้นกด Start จะได้ผลลัพธ์ตามภาพ นั้นคือ Model KNN นั้นมีประสิทธิภาพอยู่ที่ 90.3% หรือจากข้อมูล 10,000 แถว สามารถทายถูก 9,030 แถว



ภาพที่ 2.18 การทำ Cross Validation ด้วย Model SVM **(Weka)**

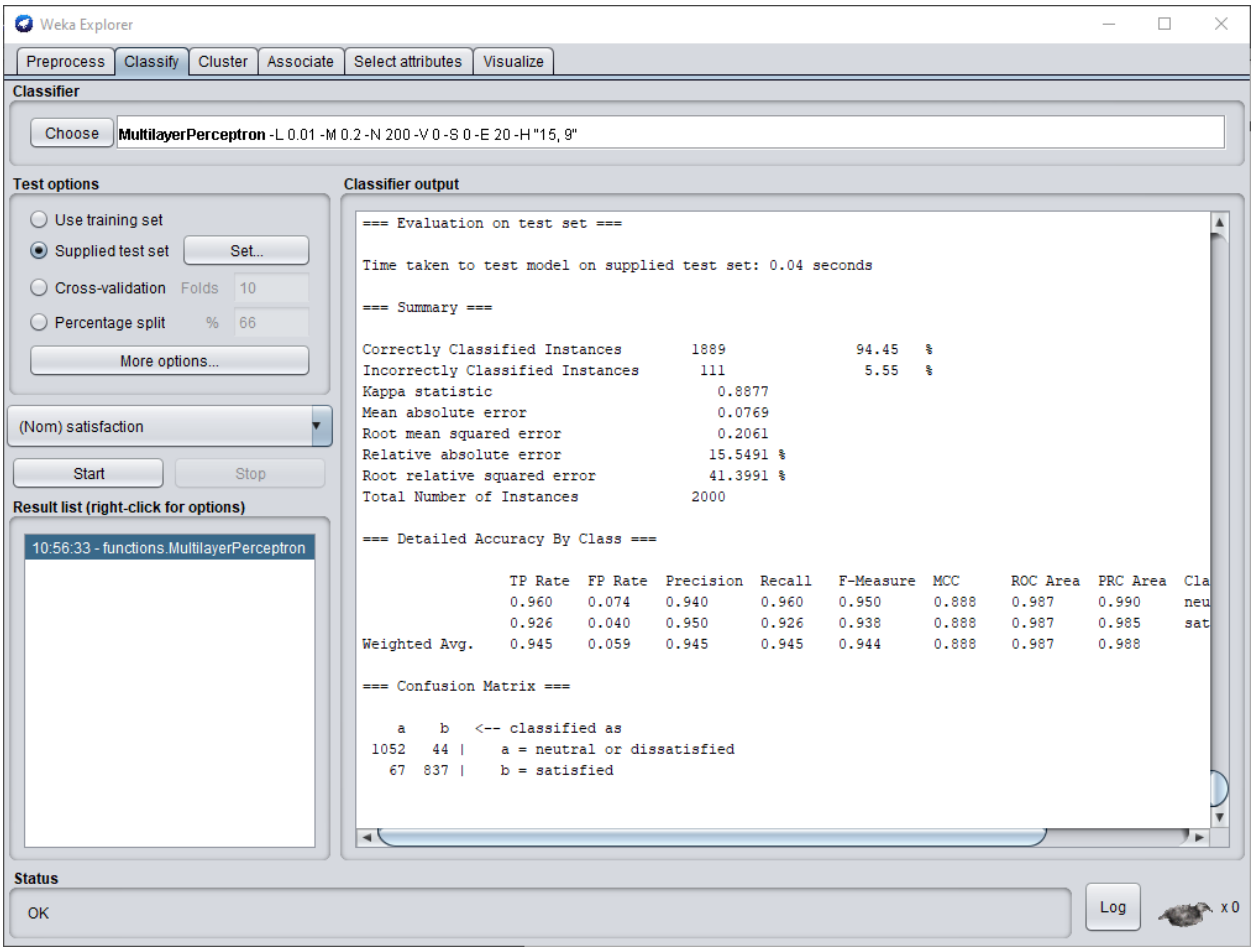
จากภาพที่ 2.18 ทำการเลือก Classifier เป็น Model SVM หรือใน Weka จะเป็น SMO โดยจะไม่ทำการกำหนดค่าอะไรเป็นพิเศษ จากนั้นเลือก Test options เป็น Cross-validation ที่ Folds เท่ากับ 10 จากนั้นกด Start จะได้ผลลัพธ์ตามภาพ นั้นคือ Model SVM นั้นมีประสิทธิภาพอยู่ที่ 87.25% หรือจากข้อมูล 10,000 แถว สามารถทายถูก 8,725 แถว



ภาพที่ 2.19 การทำ Cross Validation ด้วย Model Neural Network (Weka)

จากภาพที่ 2.19 ทำการเลือก Classifier เป็น Model Neural Network หรือใน Weka จะเป็น Multilayer Perceptron โดยจะกำหนด hidden layer หรือชั้นแรกมีขนาด 15 และชั้นสองมีขนาด 9 พร้อมทั้งกำหนดจำนวนรอบที่ 200 และกำหนด learning rate ที่ 0.01 จากนั้นเลือก Test options เป็น Cross-validation ที่ Folds เท่ากับ 10 จากนั้นกด Start จะได้ผลลัพธ์ตามภาพ นั้นคือ Model Neural Network นั้นมีประสิทธิภาพอยู่ที่ 93.94% หรือจากข้อมูล 10,000 แถว สามารถทายถูก 9,394 แถว

จากภาพที่ 2.17, 2.18 และ 2.19 ประสิทธิภาพของ Model Neural Network มีประสิทธิภาพดีที่สุด นั่นคือ 94.06% ดังนั้น จึงเลือกใช้ Model Neural Network ในการทำ Model ต่อไป



ภาพที่ 2.20 การทำ Supplied test set ด้วย Model Neural Network (Weka)

จากภาพที่ 2.20 หลังจากที่ได้ว่า Model Neural Network มีประสิทธิภาพมากที่สุด จะทำการ Supplied test set ด้วย Model Neural Network โดยก่อนการทำนั้นจะใช้ทำการเลือกชุดข้อมูล train แล้วมาเลือก Test options เป็น Supplied test set โดย Set ชุดข้อมูล test จากนั้นกด Start จะได้ผลลัพธ์ดังภาพ นั่นคือ จากการฝึกฝน Model Neural Network ด้วยชุดข้อมูล train จำนวนข้อมูล 8,000 แถว และใช้ชุดข้อมูล test มาทดสอบประสิทธิภาพ จำนวนข้อมูล 2,000 แถว ได้ว่า Model Neural Network มีประสิทธิภาพอยู่ที่ 94.45% หรือจาก 2,000 ข้อมูล ทายถูกทั้งหมด 1,889 และจากการทำ Cross Validation ที่ Model Neural Network ได้ค่าประสิทธิภาพอยู่ที่ 93.94% ซึ่งผลลัพธ์หลังจากการ test เพิ่มขึ้น ทำให้สามารถสรุปได้ว่า Model Neural Network ที่ได้ฝึกฝนมานั้น มีประสิทธิภาพที่ดีเยี่ยม ไม่มีปัญหาในเรื่อง overfiiting และ underfitting เลย เนื่องจากค่าประสิทธิภาพที่ได้นั้นมีค่าเพิ่มขึ้น

2.2.3 Python

เครื่องมือสุดท้าย นั่นคือการทำ Classification ด้วยการใช้โค้ด Python โดยกระบวนการทำจะคล้ายกับ RapidMiner Studio และ Weka นั้นคือ หา Model ที่มีประสิทธิภาพมากที่สุดด้วยการทำ Cross Validation และเมื่อได้ Model ที่มีประสิทธิภาพมากที่สุดแล้ว จะนำ Model นั้นมา Train และ Test เพื่อดูประสิทธิภาพอีกรอบ โดยในโค้ด Python นั้นจะแบ่งออกเป็น 4 ขั้นตอน ดังนี้

2.2.3.1 ขั้นตอนการ import ข้อมูล

```
train = pd.read_csv('/content/drive/MyDrive/CS/CS345/Dataset/airplane_classification_preprocess_train.csv', sep=';')
test = pd.read_csv('/content/drive/MyDrive/CS/CS345/Dataset/airplane_classification_preprocess_test.csv', sep=';')
```

ภาพที่ 2.21 การ import ข้อมูล (Python)

จากภาพที่ 2.21 จะเป็นขั้นตอนการ import ข้อมูลเข้ามา โดยจะเก็บข้อมูลชุด train ในตัวแปล train และเก็บข้อมูลชุด test ในตัวแปร test

```
train['satisfaction'].replace({'satisfied':0, 'neutral or dissatisfied':1}, inplace = True)
test['satisfaction'].replace({'satisfied':0, 'neutral or dissatisfied':1}, inplace = True)
```

ภาพที่ 2.22 แปลง satisfaction จาก nominal เป็น numerical (Python)

จากภาพที่ 2.22 เป็นการแปลงข้อมูลภายในคอมลัมน์ satisfaction ที่เป็น nominal ไปเป็น numerical ซึ่งในการทำ Model ค่าจำเป็นต้องเป็นตัวเลข จะแปลงโดยใช้คำสั่ง replace แปลงแถวที่มีค่าเป็น satisfied แปลงเป็น 0 และแปลงแถวที่มีค่าเป็น neutral or dissatisfied แปลงเป็น 1

2.2.3.2 ขั้นตอนการเตรียมข้อมูล

```
X_train = train.iloc[:, 0:26].values
Y_train = train.iloc[:, 27].values

X_test = test.iloc[:, 0:26].values
Y_test = test.iloc[:, 27].values
```

ภาพที่ 2.23 เตรียมข้อมูล (Python)

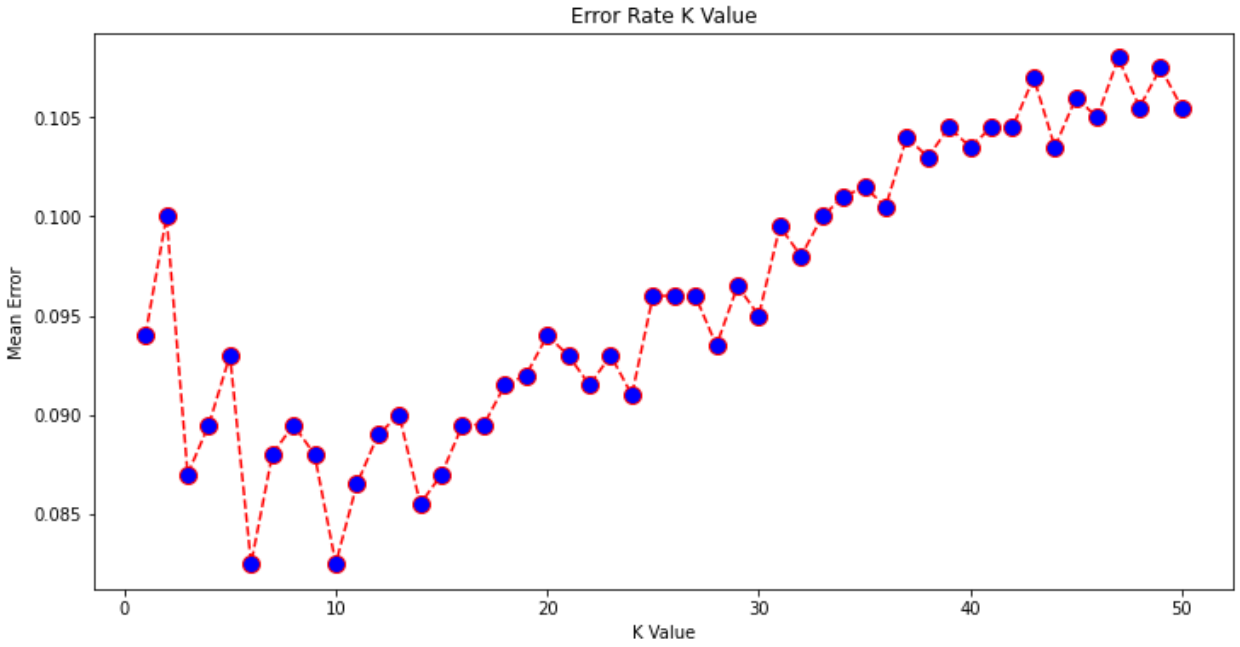
จากภาพที่ 2.23 ทำการแบ่งข้อมูลเป็นข้อมูลสำหรับ train และ test โดยเฉพาะ โดยข้อมูลชุด train จะแบ่งได้ออกมาเป็น X_train ไว้สำหรับให้ Model ฝึกฝน และ Y_train ไว้สำหรับเฉลยข้อมูลที่

Model ฝึกมา และข้อมูลชุด test จะแบ่งได้ออกมาเป็น X_test ไว้สำหรับให้ Model ทาย และ Y_test ไว้สำหรับให้ผลลัพธ์ที่ Model ทายมา มาเปรียบเทียบเพื่อดูประสิทธิภาพ

2.2.3.3 ขั้นตอนการเลือก model

อย่างที่บอกไปข้างต้น นั่นคือจะเลือก model จากการดูประสิทธิภาพสูงสุดที่วัดจากการทำ Cross Validation ซึ่งก่อนที่จะเริ่มทำ Cross Validation นั้น Model บางตัวจำเป็นต้องมีการกำหนดค่าก่อนถึงจะได้ประสิทธิภาพที่ดีที่สุด นั่นคือ Model KNN

Model KNN จำเป็นต้องมีค่า K สำหรับการทำ Model ซึ่งค่า K แต่ละค่า จะส่งผลให้มีประสิทธิภาพไม่เท่ากัน โดยวิธีการหาค่า K รูปแบบหนึ่งนั่นคือการดูจากค่า Error Rate ยิ่งค่า Error Rate มีน้อยค่า K ดังกล่าวก็จะมีประสิทธิภาพมาก ดังนั้นจึงทำการหาค่า K โดยการวนลูปตั้งแต่ค่า K ที่ 1 จนถึง ค่า K ที่ 50 เพื่อนำมาเปรียบเทียบค่า Error Rate หากค่า K ไหนที่ได้ Error Rate น้อยจะทำการเลือกค่า K นั้นไปทำ Cross Validation ต่อไป



ภาพที่ 2.24 Error Rate (Python)

จากภาพที่ 2.24 เป็นกราฟ Error Rate ของค่า K ตั้งแต่ค่าที่ 1 จนถึงค่าที่ 50 โดยที่ K ไหนมีค่า Error Rate น้อย จะเลือก K นั้นมาทำ Cross Validation แต่จะเห็นได้ว่า K ที่ 6 และ 10 มีค่า Error Rate เท่ากัน ดังนั้นจึงเลือกค่า K ทั้งสองไปทำ Cross Validation

หลังจากได้ค่า K ของ Model KNN แล้ว จึงทำการหา Cross Validation ทั้ง 3 Model

```
models = []
```

```
for i in number_of_kmeans:
  name = 'KNN at ' + str(i)
  models.append((name, KNeighborsClassifier(i)))
```

```
models.append(('SVM', SVC(kernel='linear')))
```

```
models.append(('Neural Network', MLPClassifier(solver='lbfgs', max_iter = 200, hidden_layer_sizes=(15, 9), random_state=1)))
```

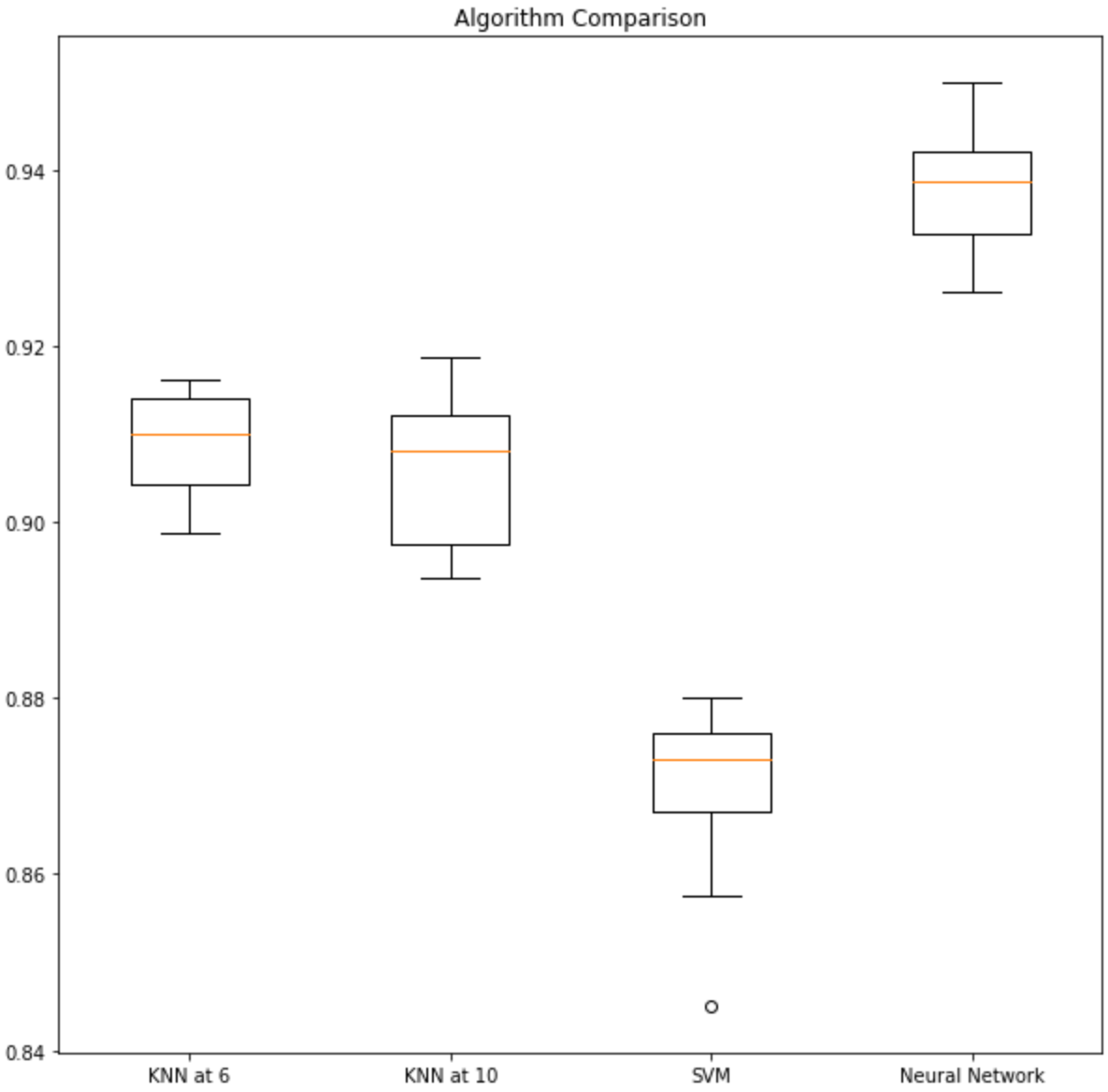
ภาพที่ 2.25 ขั้นตอนก่อนการทำ Cross Validation (Python)

จากภาพที่ 2.25 ทำการสร้าง models ที่เป็น list ไว้เก็บ Model ต่าง ๆ จากนั้นทำการเพิ่ม Model KNN ตามค่า k ที่ได้มา จากนั้นเพิ่ม Model SVM และเพิ่ม Movel Neural Network โดย Model นี้จะมีการกำหนด รอบอยู่ที่ 200 และ hidden_layer_sizes อยู่ที่ (15, 9) หรือก็คือที hidden layer อยู่ 2 ชั้น ชั้นแรกจะมีขนาด 15 และชั้นที่สองมีขนาด 9

```
KNN at 6: 0.909000 (0.005750)
KNN at 10: 0.906000 (0.008888)
SVM: 0.869500 (0.010265)
Neural Network: 0.937875 (0.007204)
```

ภาพที่ 2.26 ผลลัพธ์ Cross Validation (Python)

จากภาพที่ 2.25 จะนำ Model ทั้งหมดไปวนลูปหาค่า Cross Validation โดยได้ผลลัพธ์ดังภาพที่ 2.26 โดยผลลัพธ์คือ Model Neural Network มีประสิทธิภาพมากที่สุด ดังนั้นจึงเลือกใช้ Model Neural Network ไปทำ Model ต่อไป



ภาพที่ 2.27 เปรียบเทียบประสิทธิภาพของแต่ละ Model (Python)

จากภาพที่ 2.27 ได้มีการนำผลลัพธ์ที่ได้จากการทำ Cross Validation มาทำเป็น Box Plot เพื่อเปรียบเทียบกัน จะเห็นได้ว่า Model Neural Network มีประสิทธิภาพดีที่สุด นั่นคือมีค่าเข้าใกล้ 1 มากที่สุด

2.2.3.4 ขั้นตอนการจำแนกข้อมูล

```
model = MLPClassifier(solver='lbfgs', hidden_layer_sizes=(15, 9), max_iter = 200, random_state=1)
model.fit(X_train, Y_train)
Y_pred = model.predict(X_test)
```

ภาพที่ 2.28 ทำนายข้อมูลด้วย Neural Network (Python)

จากภาพที่ 2.28 จะทำการสร้าง Model Neural Network ที่มี hidden_layer_sizes เท่ากับ (15, 9) และมีจำนวนรอบอยู่ที่ 200 จากนั้นให้ Model ฝึกฝนผ่านข้อมูล X_train กับ Y_train เมื่อฝึกฝนเสร็จจะให้ Model ทำการทำนายข้อมูลที่อยู่ใน X_test และเก็บผลทำนายไว้ใน Y_pred

```
Accuracy Score:  0.939

Confusion Matrix
[[ 836   68]
 [  54 1042]]

Classification Report
              precision    recall  f1-score   support

           0       0.94      0.92      0.93       904
           1       0.94      0.95      0.94      1096

    accuracy                           0.94      2000
   macro avg       0.94      0.94      0.94      2000
weighted avg       0.94      0.94      0.94      2000
```

ภาพที่ 2.29 ประสิทธิภาพของ Model (Python)

จากภาพที่ 2.29 จะเป็นการนำผลที่ Model ทำนายไว้ หรือตัวแปร Y_pred มาทำการหาประสิทธิภาพ โดยดูเฉลยจาก Y_test ทำให้ได้ผลลัพธ์ดังภาพที่ 2.29 จะเห็นได้ว่า จาก 2,000 ข้อมูล Model สามารถทำนายถูกได้ถึง 1,878 ข้อมูล หรือมีความแม่นยำอยู่ที่ 0.94 ซึ่งจากผลลัพธ์ประสิทธิภาพที่ทำตอน Cross Validation ที่มีค่าประสิทธิภาพอยู่ที่ 0.93 ทำให้สรุปได้ว่า Model Neural Network ที่ได้ฝึกฝนมานั้น มีประสิทธิภาพที่ยอดเยี่ยม สามารถทำนายข้อมูลให้มีประสิทธิภาพมากกว่าตอนทำ Cross Validation ได้

บทสรุป

ชุดข้อมูล Airline Passenger Satisfaction เป็นชุดข้อมูลที่สำรวจความพึงพอใจของผู้โดยสารสายการบินสหรัฐ โดยข้อมูลจะมีแบ่งเป็น test และ train ซึ่งรวมกันแล้วจะมีข้อมูลทั้งหมด 129,880 แถว (row) และแต่ละแถวนั้นจะมีคอลัมน์ (column) ทั้งหมด 25 คอลัมน์ ได้มีการนำข้อมูลชุดนี้ไปทำ Classification หรือจัดแจกข้อมูล ผ่านเครื่องมือทั้ง 3 เครื่องมือ ซึ่งในขั้นตอนวิธีการทำนั้น ในทุกเครื่องมือจะทำการหา Model ที่มีประสิทธิภาพที่ดีที่สุดจากการทำ Cross Validation หาก Model ไหนที่มีประสิทธิภาพมากที่สุด จะถูกนำไปทำเป็น Model ในการทำ train กับ test ต่อไป โดยจะได้ผลลัพธ์ของแต่ละเครื่องมือดังนี้

1. RapidMiner Studio จากการทำ Cross Validation โดย Model KNN ที่ค่า K เท่ากับ 10 ได้ประสิทธิภาพอยู่ที่ 91.04% Model SVM ได้ประสิทธิภาพอยู่ที่ 79.70% และ Model Neural Network ที่มี learning rate เท่ากับ 0.01 จำนวนรอบอยู่ที่ 200 และมี hidden layer แบบ (15, 9) ได้ประสิทธิภาพอยู่ที่ 93.72% ซึ่ง Model Neural Network มีประสิทธิภาพดีที่สุดจึงนำมาทำ train กับ test ต่อ ซึ่งหลังจากผ่านการ train และมาทำนายข้อมูลใน test นำผลลัพธ์มาเปรียบเทียบกับเฉลย ได้ประสิทธิภาพอยู่ที่ 93.20% ทำให้สรุปได้ว่า Model ที่ได้ฝึกมานั้นมีประสิทธิภาพที่ดี แต่อาจจะยัง fit เล็กน้อยเนื่องจากได้ประสิทธิภาพที่ลดลงมา

2. Weka จากการทำ Cross Validation โดย Model KNN ที่ค่า K เท่ากับ 10 (ตามค่า K ใน RapidMiner Studio) ได้ประสิทธิภาพอยู่ที่ 90.3% Model SVM ได้ประสิทธิภาพอยู่ที่ 87.25% และ Model Neural Network ที่มี learning rate เท่ากับ 0.01 จำนวนรอบอยู่ที่ 200 และมี hidden layer แบบ (15, 9) ได้ประสิทธิภาพอยู่ที่ 93.94% ซึ่ง Model Neural Network มีประสิทธิภาพดีที่สุดจึงนำมาทำ train กับ test ต่อ ซึ่งหลังจากผ่านการ train และมาทำนายข้อมูลใน test นำผลลัพธ์มาเปรียบเทียบกับเฉลย ได้ประสิทธิภาพอยู่ที่ 94.45% ทำให้สรุปได้ว่า Model ที่ได้ฝึกมานั้นมีประสิทธิภาพที่ดีเยี่ยม

3. Python จากการทำ Cross Validation โดย Model KNN ที่ค่า K เท่ากับ 6 กับ 10 ได้ประสิทธิภาพอยู่ที่ 90.9% และ 90.6% ตามลำดับ Model SVM ได้ประสิทธิภาพอยู่ที่ 86.95% และ Model Neural Network ที่มี จำนวนรอบอยู่ที่ 200 และมี hidden layer แบบ (15, 9) ได้ประสิทธิภาพอยู่ที่ 93.78% ซึ่ง Model Neural Network มีประสิทธิภาพดีที่สุดจึงนำมาทำ train กับ test ต่อ ซึ่งหลังจากผ่านการ train และมาทำนายข้อมูลใน test นำผลลัพธ์มาเปรียบเทียบกับเฉลย ได้ประสิทธิภาพอยู่ที่ 94% ทำให้สรุปได้ว่า Model ที่ได้ฝึกมานั้นมีประสิทธิภาพที่ดีเยี่ยม

จากผลลัพธ์ของทั้ง 3 เครื่องมือ พบว่า Model Neural Network มีประสิทธิภาพที่ดีที่สุดในทั้ง 3 เครื่องมือ จากการทำ Cross Validation ของ Model Neural Network ค่าประสิทธิภาพจะอยู่ที่ 93.72%, 93.94% และ 93.78% ตามลำดับเครื่องมือที่ใช้ และจากการฝึกฝนด้วยข้อมูล train และทำนายข้อมูลใน test ได้

ประสิทธิภาพจากเครื่องมือทั้ง 3 RapidMiner Studio, Weka, และ Python อยู่ที่ 93.20%, 94.45% และ 94% ตามลำดับ จะมีเพียง RapidMiner Studio เท่านั้นที่ได้ค่าประสิทธิภาพลดลง ในขณะที่ Weka และ Python นั้นได้ประสิทธิภาพที่เพิ่มขึ้น

ดังนั้นจากผลลัพธ์ที่ได้มาทั้งหมด สามารถตอบคำถามที่ตั้งไว้ในตอนแรก นั้นคือ Model เหล่านี้สามารถจำแนกข้อมูลได้ โดยได้ประสิทธิภาพแตกต่างกันออกไป และ Model ที่มีประสิทธิภาพมากที่สุดนั้นคือ Model ของ Neural Network

รายการอ้างอิง

[1] T. Klein. (2019). “Airline Passenger Satisfaction” [Online]. Available: https://www.kaggle.com/teejmahal20/airline-passenger-satisfaction. [Accessed: Nov. 17, 2021].