การจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษหรือไม่มีพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง

MUSHROOM CLASSIFICATION BETWEEN POISONOUS AND EDIBLE MUSHROOMS USING MACHINE LEARNING TECHNIQUES

ภิรมย์พร เถี่ยนมิตรภาพ

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2565

การจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษหรือไม่มีพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง

ภิรมย์พร เถี่ยนมิตรภาพ

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2565

MUSHROOM CLASSIFICATION BETWEEN POISONOUS AND EDIBLE MUSHROOMS USING MACHINE LEARNING TECHNIQUES

PHIROMPORN TIANMITRAPAP

A Master's Project Submitted in Partial Fulfillment of the Requirements

for the Degree of MASTER OF SCIENCE

(Data Science)

Faculty of Science, Srinakharinwirot University

2022

สารนิพนธ์

เรื่อง

การจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษหรือไม่มีพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง

ของ

ภิรมย์พร เถี่ยนมิตรภาพ

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณบดีบัณฑิตวิทยาลัย

คณะกรรมการสอบปากเปล่าสารนิพนธ์

..... ที่ปรึกษาหลัก

(ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์)

..... ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.อัครา ประโยชน์)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.ศิริสรรพ เหล่าหะเกียรติ)

ชื่อเรื่อง	การจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษหรือไม่มีพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง
ผู้วิจัย	ภิรมย์พร เถี่ยนมิตรภาพ
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2565
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. วราภรณ์ วิยานนท์

เห็ดตามธรรมชาติมีทั้งเห็ดที่รับประทานได้และเห็ดที่มีพิษ หากคนรับประทานเห็ดพิษเข้าไปทำให้เกิดอาการเจ็บป่วยและถึงแก่ชีวิตได้ ดังนั้นจึงได้นำเทคนิคการเรียนรู้ของเครื่องมาใช้ในการจำแนกประเภทของเห็ดระหว่างเห็ดพิษและเห็ดที่รับประทานได้โดยใช้แบบจำลอง 5 แบบจำลอง ได้แก่ Logistic regression, SVM, Decision tree, Random Forest และ XGBoost ร่วมกับการใช้เทคนิคการคัดเลือกและสกัดฟีเจอร์ เพื่อค้นหาฟีเจอร์ที่สำคัญที่สามารถบ่งบอกประเภทของเห็ดได้ โดยผลการศึกษาพบว่า การคัดเลือกฟีเจอร์ทั้ง RFE และ Chi-square เมื่อจำนวนฟีเจอร์เพิ่มขึ้น ค่า cross validation score เพิ่มขึ้น และคงที่เมื่อจำนวนฟีเจอร์เท่ากับ 3 ฟีเจอร์ สำหรับเทคนิคการสกัดฟีเจอร์ด้วย PCA นั้น เมื่อจำนวน component เพิ่มขึ้น ค่า cross validation score เพิ่มขึ้น และคงที่เมื่อจำนวน component เท่ากับ 3 components เมื่อพิจารณาผลลัพธ์พบว่าการใช้แบบจำลอง Decision tree ที่ใช้เทคนิค Recursive Feature Elimination (RFE) ในการคัดเลือกฟีเจอร์โดยใช้ training size เท่ากับ 60% และใช้ฟีเจอร์เพียง 3 ฟีเจอร์ได้แก่ กลิ่นของเห็ด (odor) ขนาดครีบเห็ด (gill_size) และสีรอยพิมพ์สปอร์ (spore_print_color) มีความเหมาะสมที่สุดโดยได้ F1 score เท่ากับ 99.32% ซึ่งนำโมเดลที่ได้ไปสร้างต้นแบบเว็บแอปพลิเคชัน

คำสำคัญ : การจำแนกประเภทเห็ด, เทคนิคการเรียนรู้ของเครื่อง, การคัดเลือกฟีเจอร์, การสกัดฟีเจอร์, เว็บแอปพลิเคชัน, เทคนิคการวิเคราะห์องค์ประกอบหลัก, ต้นไม้การตัดสินใจ

Title	MUSHROOM CLASSIFICATION BETWEEN POISONOUS AND EDIBLE MUSHROOMS USING MACHINE LEARNING TECHNIQUES
Author	PHIROMPORN TIANMITRAPAP
Degree	MASTER OF SCIENCE
Academic Year	2022
Thesis Advisor	Assistant Professor Waraporn Viyanon , Ph.D.

Mushrooms are a type of natural fungi that can either be edible or poisonous. Consuming poisonous mushrooms can lead to severe illness or even death. Therefore, machine learning techniques and models have been used to classify mushrooms into edible and poisonous types. This study explores the effectiveness of five machine learning models, including Logistic regression, SVM, Decision tree, Random Forest, and XGBoost, along with feature selection techniques to identify the most significant features to indicate the type of mushroom. Recursive Feature Elimination (RFE) and Chi-square feature selection techniques were used to select important features, while Principal Component Analysis (PCA) was used for feature extraction. The results showed that both RFE and the Chi-square feature selection techniques increased the cross-validation score as the number of features increased. Additionally, the PCA technique increased the cross-validation score as the number of components increased. After comparing the results of the models, it was found that the Decision tree model, which used RFE feature selection technique with a training size of 60%, and only three features, namely odor, gill size, and spore print color, yielded the highest F1 score of 99.32%. This model was then used to develop a prototype web application for mushroom classification.

Keyword : Mushroom classification, Machine Learning, Feature selection, Recursive Feature Elimination, Feature extraction, Chi-square, Principal Component Analysis, Decision Tree, F1 score, Web application

กิตติกรรมประกาศ

สารนิพนธ์นี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จาก ผศ.ดร.วราภรณ์ วิยานนท์ อาจารย์ที่ปรึกษา ที่ให้คำปรึกษาคำแนะนำในการจัดทำและแก้ไขข้อบกพร่องสารนิพนธ์ ตลอดจนสนับสนุนข้อมูลต่างๆ สำหรับจัดทำสารนิพนธ์นี้

ขอกราบขอบพระคุณคณะกรรมการสอบสารนิพนธ์ที่ได้ให้คำแนะนำและข้อเสนอแนะในการปรับปรุงสารนิพนธ์และการใช้เทคนิคการเรียนรู้ของเครื่องเป็นเครื่องมือในการวิเคราะห์ และแก้ไขปัญหาทางข้อมูลต่อไป

ขอกราบขอบพระคุณคณาจารย์ทุกท่านในภาควิชาวิทยาการข้อมูล มหาวิทยาลัยศรีนครินทรวิโรฒ ที่ได้ให้ความรู้และเกิดการต่อยอดเป็นงานวิจัยนี้อย่างสำเร็จ

ขอขอบคุณ คีรีลักษณ์ ศิริคะรินทร์ ที่ช่วยสนับสนุนและทำให้เกิดการต่อยอดจนเกิดเป็นงานวิจัยนี้

ขอขอบคุณ ชิน เลิศวิภาดา, พัฒนเดช แจ้งศรีสุข ที่ให้คำแนะนำเกี่ยวกับเทคนิคในการเขียนคำสั่งภาษา python ต่างๆ ในงานวิจัยนี้

ภิรมย์พร เถี่ยนมิตรภาพ

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง

บทคัดย่อภาษาอังกฤษ ..... จ

กิตติกรรมประกาศ ..... ฉ

สารบัญ ..... ช

สารบัญตาราง ..... ฌ

สารบัญรูปภาพ ..... ญ

บทที่ 1 บทนำ 1

1. ภูมิหลัง 1

2. ความมุ่งหมายของงานวิจัย 3

3. ความสำคัญของการวิจัย 3

4. ขอบเขตของการวิจัย 3

ชุดข้อมูลที่ใช้ในการวิจัย 3

รายละเอียดของชุดข้อมูล 4

ตัวแปรที่ศึกษา 4

5. กรอบแนวคิดในงานวิจัย 5

6. สมมติฐานในการวิจัย 6

บทที่ 2 ทบทวนวรรณกรรม 7

1. ข้อมูลสัณฐานวิทยา (Morphology) และคุณลักษณะประจำตัวของเห็ดครีบ 7

2. เทคนิคการเรียนรู้ของเครื่อง (Machine Learning Techniques) 13

3. แบบจำลองอัลกอริธึมที่ใช้ในเทคนิคการเรียนรู้ของเครื่องที่ใช้ในงานวิจัย 14

4. การแบ่งชุดข้อมูลสำหรับการทดสอบประสิทธิภาพของแบบจำลอง 18

5. การปรับช่วงขอบเขตของฟีเจอร์ (Feature scaling) 18
6. การปรับค่าพารามิเตอร์ (Hyperparameter Tuning) 20
7. การคัดเลือกฟีเจอร์และการสกัดฟีเจอร์ (Feature Selection and Feature Extraction) 21
8. งานวิจัยที่เกี่ยวข้อง 22
บทที่ 3 วิธีดำเนินการวิจัย 32
1. ชุดข้อมูลและการแบ่งชุดข้อมูล 32
2. การออกแบบวิธีการดำเนินงานวิจัย 40
3. การประมวลผลและการวิเคราะห์ข้อมูล 41
4. การสร้างแบบจำลอง และประเมินประสิทธิภาพแบบจำลอง 60
5. การพัฒนาแบบจำลองเป็นเว็บแอปพลิเคชัน 74
บทที่ 4 ผลการศึกษา 76
1. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ข้อมูลคุณลักษณะทั้งหมดของเห็ดครีบจากชุดข้อมูล 76
2. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Recursive Feature Elimination 81
3. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี chi-square 91
4. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้เทคนิคการสกัดฟีเจอร์ด้วยวิธี Principal Component Analysis (PCA) 101
บทที่ 5 สรุป อภิปรายผล และข้อเสนอแนะ 116
1. สรุปผลการวิจัย และอภิปรายผล 116
2. ข้อเสนอแนะ 120
บรรณานุกรม 121
ประวัติผู้เขียน 128

สารบัญตาราง

หน้า

ตาราง 1 รายงานจำนวนผู้ป่วย และผู้เสียชีวิตจากการรับประทานเห็ดพิษ ปี 2559 – 2565..... 2

ตาราง 2 สรุปงานวิจัยที่เกี่ยวข้อง..... 29

ตาราง 3 รายละเอียดของชุดข้อมูลเห็ดครีบ..... 33

ตาราง 4 ประสิทธิภาพของแบบจำลองจำแนกประเภทของเห็ดโดยใช้ฟีเจอร์ทั้งหมด 77

ตาราง 5 ประสิทธิภาพของแบบจำลอง ที่ใช้เทคนิคคัดเลือกฟีเจอร์ด้วยวิธี RFE..... 87

ตาราง 6 ประสิทธิภาพของแบบจำลอง ที่ใช้เทคนิคคัดเลือกฟีเจอร์ด้วยวิธี Chi-square 98

ตาราง 7 ประสิทธิภาพของแบบจำลอง โดยใช้เทคนิคการสกัดฟีเจอร์ด้วยวิธี PCA 112

สารบัญรูปภาพ

หน้า

ภาพประกอบ 1 สัณฐานวิทยา (Morphology) ของเห็ดครีบ 7
ภาพประกอบ 2 ตัวอย่างรูปทรงของหมวกเห็ด 8
ภาพประกอบ 3 ตัวอย่างรูปทรงของหมวกเห็ด 9
ภาพประกอบ 4 ตัวอย่างลักษณะพื้นผิวบนหมวกเห็ด 9
ภาพประกอบ 5 ตัวอย่างลักษณะการยึดติดของครีบเห็ดกับก้านดอก 10
ภาพประกอบ 6 ตัวอย่างลักษณะการเรียงตัวของครีบเห็ด 10
ภาพประกอบ 7 ตัวอย่างลักษณะการเรียงตัวของผิวครีบเห็ด และขนาดของครีบเห็ด 10
ภาพประกอบ 8 ตัวอย่างรูปร่างของก้านดอก 11
ภาพประกอบ 9 วิธีการเก็บตัวอย่างของรอยพิมพ์สปอร์ 12
ภาพประกอบ 10 กราฟของสมการ Sigmoid function 14
ภาพประกอบ 11 การแบ่งกลุ่มข้อมูล 2 ประเภท ด้วยเส้นแบ่งที่มีขอบเขต (margin) ห่างจาก support vector มากที่สุด 15
ภาพประกอบ 12 หลักการทำงานของแบบจำลอง Decision tree 15
ภาพประกอบ 13 หลักการทำงานของแบบจำลอง Random Forest 17
ภาพประกอบ 14 หลักการทำงานของแบบจำลอง XGBoost 17
ภาพประกอบ 15 การแบ่งชุดข้อมูลสำหรับการทดสอบประสิทธิภาพของแบบจำลอง 18
ภาพประกอบ 16 รูปแบบของหลักการระหว่าง GridSearchCV และ RandomizedSearchCV .. 21
ภาพประกอบ 17 ภาพรวมการดำเนินการวิจัย 40
ภาพประกอบ 18 ตัวอย่างข้อมูลในตารางของชุดข้อมูลดั้งเดิมจำนวน 5 แถวแรก ด้วยคำสั่ง df.head() 41

ภาพประกอบ 19 รายละเอียดของชุดข้อมูลดั้งเดิม ได้แก่ จำนวนแถว, คอลัมน์ทั้งหมด และชนิดของข้อมูล ด้วยคำสั่ง df.info() 42

ภาพประกอบ 20 ตัวอย่างข้อมูลในตารางของชุดข้อมูลที่ผ่านการปรับแต่งให้เหมาะสม ด้วยคำสั่ง df.head() 43

ภาพประกอบ 21 รายละเอียดของชุดข้อมูลที่ผ่านการปรับแต่งให้เหมาะสม ด้วยคำสั่ง df.info() 44

ภาพประกอบ 22 ตรวจสอบจำนวนค่าว่าง และแถวข้อมูลที่ซ้ำกัน 45

ภาพประกอบ 23 ข้อมูลตัวแปรในคอลัมน์ stalk_root (ลักษณะรากที่อยู่ปลายก้านดอก) ที่มีการเติมค่าว่างด้วยคำว่า missing 45

ภาพประกอบ 24 กราฟแท่งแสดงจำนวนข้อมูลเห็ดในแต่ละประเภท 46

ภาพประกอบ 25 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 46

ภาพประกอบ 26 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะพื้นผิวของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 47

ภาพประกอบ 27 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปทรงของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 47

ภาพประกอบ 28 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของครีบเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 48

ภาพประกอบ 29 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามความแนบชิดของครีบเห็ดกับก้านดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 48

ภาพประกอบ 30 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามระยะห่างของครีบเห็ดแต่ละครีบ โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 49

ภาพประกอบ 31 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามขนาดของครีบเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 49

ภาพประกอบ 32 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปทรงของก้านดอก โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 50

ภาพประกอบ 33 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะรากที่อยู่ปลายก้านดอก โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 50

ภาพประกอบ 34 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 51

ภาพประกอบ 35 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 51

ภาพประกอบ 36 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะของพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 52

ภาพประกอบ 37 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะของพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 52

ภาพประกอบ 38 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามประเภทของเยื่อที่หุ้มดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 53

ภาพประกอบ 39 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของเยื่อที่หุ้มดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 53

ภาพประกอบ 40 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามจำนวนวงแหวนของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 54

ภาพประกอบ 41 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปร่างของวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 54

ภาพประกอบ 42 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรอยช้ำบนเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 55

ภาพประกอบ 43 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามกลิ่นของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 55

ภาพประกอบ 44 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของรอยพิมพ์สปอร์ โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 56

ภาพประกอบ 45 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะการกระจายตัว/การเกาะกลุ่มกันของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 56

ภาพประกอบ 46 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามบริเวณที่พบเจอเห็ดเจริญเติบโต โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้ 57

ภาพประกอบ 47 คำสั่งสำหรับการกำหนดตัวแปรต้น (feature) และตัวแปรตาม (Target) 57

ภาพประกอบ 48 ตัวอย่างคำสั่งการแปลงข้อมูลให้เป็นตัวเลข โดยใช้ Ordinal encoder และ Label Encoder 58

ภาพประกอบ 49 ค่าความสัมพันธ์ระหว่างข้อมูล (correlation) 59

ภาพประกอบ 50 ตัวอย่างคำสั่งการแบ่งชุดข้อมูลในอัตราส่วน 50 : 50 59

ภาพประกอบ 51 ตัวอย่างคำสั่งการปรับช่วงขอบเขตของฟีเจอร์ 60

ภาพประกอบ 52 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Logistic regression 60

ภาพประกอบ 53 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Support Vector Machine 60

ภาพประกอบ 54 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Decision tree 61

ภาพประกอบ 55 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Random Forest 61

ภาพประกอบ 56 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง XGBoost 61

ภาพประกอบ 57 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Logistic regression 63

ภาพประกอบ 58 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Decision tree 63

ภาพประกอบ 59 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Random Forest 64

ภาพประกอบ 60 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ XGBoost 64

ภาพประกอบ 61 ตัวอย่างคำสั่งสร้างกราฟแสดงค่า cross validation score ในแต่ละ n_features_to_select 65

ภาพประกอบ 62 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Logistic regression 66

ภาพประกอบ 63 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Support Vector Machine 67

ภาพประกอบ 64 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Decision tree 67

ภาพประกอบ 65 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Random Forest 68

ภาพประกอบ 66 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ XGBoost 68

ภาพประกอบ 67 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้ n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Logistic regression 69

ภาพประกอบ 68 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้ n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Support Vector Machine 70

ภาพประกอบ 69 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Decision tree 70

ภาพประกอบ 70 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Random Forest 71

ภาพประกอบ 71 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ XGBoost 71

ภาพประกอบ 72 confusion matrix 72

ภาพประกอบ 73 ตัวอย่าง ROC_AUC curves และ AUC score 74

ภาพประกอบ 74 ตัวอย่างแผนผังหน้าเว็บแอปพลิเคชัน 75

ภาพประกอบ 75 Confusion matrix ของแบบจำลองที่ใช้ฟีเจอร์ทั้งหมด 80

ภาพประกอบ 76 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Logistic regression ที่ training size ต่างๆ 82

ภาพประกอบ 77 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Decision Tree ที่ training size ต่างๆ..... 83

ภาพประกอบ 78 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Random Forest ที่ training size ต่างๆ 84

ภาพประกอบ 79 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ XGBoost ที่ training size ต่างๆ (ก) training size = 50%, ... 85

ภาพประกอบ 80 กราฟแสดงค่า chi-square ของฟีเจอร์ของเห็ดในแต่ละ training size..... 91

ภาพประกอบ 81 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Logistic regression ที่ training size ต่างๆ..... 92

ภาพประกอบ 82 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Support Vector Machine ที่ training size ต่างๆ . 93

ภาพประกอบ 83 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Decision Tree ที่ training size ต่างๆ 94

ภาพประกอบ 84 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Random Forest ที่ training size ต่างๆ..... 95

ภาพประกอบ 85 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ XGBoost ที่ training size ต่างๆ 96

ภาพประกอบ 86 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Logistic regression ที่ training size ต่างๆ..... 101

ภาพประกอบ 87 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Support Vector Machine ที่ training size ต่างๆ 102

ภาพประกอบ 88 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Decision Tree ที่ training size ต่างๆ 103

ภาพประกอบ 89 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Random Forest ที่ training size ต่างๆ 104

ภาพประกอบ 90 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ XGBoost ที่ training size ต่างๆ 105

ภาพประกอบ 91 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 50% 108

ภาพประกอบ 92 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 60% 109

ภาพประกอบ 93 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 70% 110

ภาพประกอบ 94 ตัวอย่างเว็บแอปพลิเคชันต้นแบบ
(ก) ตัวอย่างผลการทำนายเห็ดรับประทานได้ (ข) ตัวอย่างผลการทำนายเห็ดพิษ 115

ภาพประกอบ 95 กราฟแสดงประสิทธิภาพของแต่ละแบบจำลอง โดยใช้ฟีเจอร์ทั้งหมด 20 ฟีเจอร์ 117

ภาพประกอบ 96 ประสิทธิภาพของแต่ละแบบจำลองโดยคัดเลือกฟีเจอร์ด้วยวิธี RFE 118

ภาพประกอบ 97 ประสิทธิภาพของแต่ละแบบจำลองโดยคัดเลือกฟีเจอร์ด้วยวิธี Chi-square .. 119

ภาพประกอบ 98 ประสิทธิภาพของแต่ละแบบจำลองโดยการสกัดฟีเจอร์ด้วยวิธี PCA 119

บทที่ 1
บทนำ

1. ภูมิหลัง

เห็ดเป็นสิ่งมีชีวิตที่จัดอยู่ในอาณาจักรเห็ดรา (Kingdom fungi) ซึ่งเห็ดจัดเป็นราชั้นสูง ที่อยู่ใน 2 ไฟลัม (phylum) คือ Ascomycota และ Basidiomycota โดยเห็ดที่ประชาชนรู้จักทั่วไปและพบเห็นได้บ่อยนั้นเป็นเห็ดในไฟลัม Basidiomycota ในอันดับ Agaricales และ Russulales ที่มีลักษณะเป็นตอกเห็ด และรูปร่างคล้ายร่ม สามารถมองเห็นได้ด้วยตาเปล่า ซึ่งมีชื่อเรียกทางวิชาการว่า “เห็ดครีบ (gilled mushroom หรือ agaric mushroom)” [1, 2]

จากสิ่งมีชีวิตในอาณาจักรเห็ดราทั้งหมดจำนวน 1.5 ล้านชนิด นักวิทยาศาสตร์ค้นพบเพียง 110,000 ชนิด และมีการจำแนกว่าเป็นเห็ด 14,000 ชนิด และถูกระบุว่าเป็นเห็ดที่สามารถรับประทานได้เพียง 3,000 ชนิดเท่านั้น [3]

โดยในประเทศไทยตั้งอยู่ในตำแหน่งใกล้เส้นศูนย์สูตรในพื้นที่เขตร้อนชื้น และมีฝนตกชุกจึงเหมาะกับการเจริญเติบโตของเห็ดได้เป็นอย่างดี [1, 4] โดยเห็ดที่ขึ้นตามธรรมชาติมีทั้งเห็ดที่รับประทานได้และเห็ดที่มีพิษ สำหรับเห็ดที่รับประทานได้นั้นให้คุณค่าทางโภชนาการมากมาย ได้แก่ โปรตีน, แร่ธาตุต่างๆ เช่น โพแทสเซียม ฟอสฟอรัส แมกนีเซียม และสังกะสี อีกทั้งยังเป็นแหล่งของเส้นใย (fiber) และวิตามินต่างๆ เช่น วิตามิน B1, วิตามิน B2, วิตามิน B3 เป็นต้น [5]

จากการที่เห็ดที่รับประทานได้มีประโยชน์หลากหลาย จึงมีการเก็บเห็ดที่เกิดขึ้นตามธรรมชาติ มาใช้ประโยชน์ต่างๆ เช่น นำมารับประทานในครัวเรือน จำหน่ายแบบสด พัฒนาและเพาะพันธุ์ต่อเพื่อนำไปผลิตในวงการอุตสาหกรรมอาหารและส่งออกต่างประเทศ เป็นต้น [6]

สำหรับเห็ดพิษคือเห็ดที่สร้างสารพิษชีวภาพที่เกิดจากกระบวนการทางสรีรวิทยา ทำให้คนที่ได้รับสารพิษดังกล่าวเกิดอาการเจ็บป่วยและถึงแก่ชีวิตได้ โดยพบว่าเห็ดพิษชนิดเดียวกันอาจมีสารพิษอยู่หลายชนิดแตกต่างกันตามพื้นที่ที่เห็ดเกิด รวมไปถึงเห็ดที่รับประทานได้ แต่เกิดในบริเวณที่มีสารพิษอยู่ในดิน อาจทำให้เกิดอันตรายได้ นอกจากนี้การปรุงอาหารจากเห็ดที่ไม่สุกก็ทำให้เกิดพิษได้เช่นกัน [5]

โดยสารพิษจากเห็ดมีการจำแนกออกเป็น 4 กลุ่มใหญ่ๆ ได้แก่

กลุ่มที่ 1 Protoplasmic poisons : เป็นกลุ่มของสารพิษที่เข้าไปทำลายเซลล์ภายในร่างกาย และทำให้เกิดการล้มเหลวของอวัยวะภายในร่างกาย สารพิษในกลุ่มนี้ ได้แก่ Amanitins ซึ่งเป็นสารพิษที่ร้ายแรงที่สุดในบรรดาสารพิษในเห็ด และมีระยะฟักตัวที่นาน (ประมาณ

6 – 24 ชั่วโมง) โดยการนำเห็ดที่มีสารพิษนี้ ไปปรุงอาหารโดยผ่านความร้อน ก็ไม่สามารถทำลายสารพิษได้ [6, 7]

กลุ่มที่ 2 Neurotoxin : เป็นกลุ่มของสารพิษที่มีผลต่อระบบประสาท เช่น มีเหงื่อออกมาก น้ำตาไหล น้ำลายไหล และอาจทำให้เกิดภาวะหัวใจเต้นช้าและถึงแก่ชีวิตในที่สุด ตัวอย่างสารพิษในกลุ่มนี้คือ อัลคาลอยด์มัสคารีน (alkaloid muscarine) [6, 7]

กลุ่มที่ 3 Gastrointestinal irritants : เป็นกลุ่มของสารพิษที่ทำให้เกิดอาการระคายเคืองต่อกระเพาะอาหาร และลำไส้ ทำให้มีอาการคลื่นไส้ อาเจียน เป็นตะคริวที่ช่องท้อง ท้องเสีย โดยมีจำนวนน้อยที่ถึงแก่ชีวิต สารพิษในกลุ่มนี้พบในเห็ดจำนวนมากที่พบในประเทศไทย [6, 7]

กลุ่มที่ 4 Disulfiram like poisoning (Coprine) : เป็นสารพิษในเห็ดที่เมื่อคนรับประทานร่วมกับแอลกอฮอล์ จะปรากฏอาการ ภายใน 24-72 ชั่วโมง หลังรับประทานเห็ด โดยมีอาการหน้าแดง ร้อน มีเหงื่อออกที่หน้ามาถึงคอและหน้าอก ปวดหัวอย่างรุนแรง คลื่นไส้ อาเจียน หายใจเร็วและลำบาก [6, 7]

สถานการณ์โรคอาหารเป็นพิษจากการรับประทานเห็ดพิษ จากข้อมูลการเฝ้าระวังโรค สำนักระบาดวิทยา กรมควบคุมโรคในอดีตที่ผ่านมานั้นมีรายงานจำนวนผู้ป่วย และผู้เสียชีวิตในช่วงเวลาต่างๆ ตั้งแต่ปี 2559 ถึง 2565 ตามตาราง 1

ตาราง 1 รายงานจำนวนผู้ป่วย และผู้เสียชีวิตจากการรับประทานเห็ดพิษ ปี 2559 – 2565

ปี พ.ศ.	ช่วงเวลาที่พบรายงาน	จำนวนผู้ป่วย (คน)	จำนวนผู้เสียชีวิต (คน)	อ้างอิงรายงาน
2559	1 มกราคม - กันยายน 2559	1,220	4	[8]
2560	1 มกราคม -13 ตุลาคม 2560	1,093	5	[9]
2561	1 มกราคม - 23 สิงหาคม 2561	1,175	6	[10]
2562	1 มกราคม - 26 พฤษภาคม 2562	141	5	[11]
	เดือนมิถุนายน - สิงหาคม 2562	989	0	[12]
2563	1 มกราคม - 24 สิงหาคม 2563	1,686	7	[12, 13]
2564	1 มกราคม - 6 มิถุนายน 2564	218	2	[14]
2565	1 มกราคม - 20 กรกฎาคม 2565	632	0	[15]

จากการข้อมูลข้างต้นเห็นได้ว่า การรับประทานเห็ดพิษนั้นก่อให้เกิดอันตรายแก่สุขภาพของประชาชน และอาจถึงแก่ชีวิตได้ ดังนั้นการที่สามารถจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษและเห็ดที่สามารถรับประทานได้จึงมีความสำคัญ ดังนั้นผู้วิจัยจึงมีความสนใจในการสร้างเทคนิคการเรียนรู้ของเครื่อง (Machine Learning Techniques) เพื่อช่วยในการจำแนกประเภทของเห็ดโดยใช้ข้อมูลคุณลักษณะของเห็ด

2. ความมุ่งหมายของงานวิจัย

ในการวิจัยครั้งนี้ผู้วิจัยได้ตั้งความมุ่งหมายไว้ดังนี้

2.1 เพื่อค้นหาคุณลักษณะที่สำคัญที่สามารถบ่งบอกและจำแนกประเภทของเห็ดมีพิษออกจากเห็ดที่สามารถรับประทานได้ และแสดงเป็นกราฟ

2.2 เพื่อสร้างเทคนิคการเรียนรู้ของเครื่องที่มีประสิทธิภาพในการจำแนกเห็ด 2 ประเภท คือ เห็ดมีพิษ และเห็ดที่สามารถรับประทานได้ ด้วยเทคนิคการเรียนรู้ของเครื่องแบบมีผู้สอน (Supervised Machine Learning)

2.3 เพื่อนำเทคนิคการเรียนรู้ของเครื่องที่มีประสิทธิภาพไปใช้ประโยชน์แก่ประชาชนที่มีพฤติกรรมในการเก็บเห็ดป่ามารับประทานหรือนำมาขาย โดยนำไปใช้งานในรูปแบบเว็บแอปพลิเคชัน เพื่อให้สาธารณชนจำแนกประเภทเห็ดเบื้องต้นได้อย่างทันที

3. ความสำคัญของการวิจัย

งานจำแนกประเภทเห็ดระหว่างเห็ดมีพิษและเห็ดที่สามารถรับประทานได้ที่พัฒนาขึ้นมาจากเทคนิคการเรียนรู้ของเครื่องนี้ คาดหวังว่าสามารถช่วยให้ประชาชนที่มีพฤติกรรมในการเก็บเห็ดป่ามารับประทานหรือนำมาขาย สามารถระมัดระวังและหลีกเลี่ยงการเก็บเห็ดที่มีพิษได้

4. ขอบเขตของการวิจัย

ชุดข้อมูลที่ใช้ในการวิจัย

ใช้ข้อมูลสาธารณะของเห็ดครีบจำนวน 23 สายพันธุ์ในตระกูล Agaricus และ Lepiota จำนวน 8,124 ชนิด ซึ่งมีการระบุว่าเป็นเห็ดพิษ หรือเห็ดที่รับประทานได้ ซึ่งได้รับมาจากเว็บไซต์ kaggle [16] โดยเป็นข้อมูลที่ Jeff Schlimmer ซึ่งเป็นผู้บริจาคให้กับ UCI Machine Learning Repository ได้ดึงมาจากข้อมูลในหนังสือ The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf [17, 18]

รายละเอียดของชุดข้อมูล

ใช้ข้อมูลสาธารณะของเห็ดครีบจำนวน 23 สายพันธุ์ในตระกูล Agaricus และ Lepiota จำนวน 8,124 ชนิด ซึ่งมีการระบุว่าเป็นเห็ดพิษ หรือเห็ดที่รับประทานได้ โดยมีข้อมูลคุณลักษณะของเห็ดทั้งหมด 22 คอลัมน์

ตัวแปรที่ศึกษา

1. ตัวแปรอิสระ แบ่งเป็นดังนี้

1.1 ข้อมูลสัณฐานวิทยาของเห็ด

1.1.1 หมวกเห็ด

- สีของหมวกเห็ด
- ลักษณะพื้นผิวบนหมวกเห็ด
- รูปทรงของหมวกเห็ด

1.1.2 ครีบเห็ด

- สีของครีบเห็ด
- ความแนบชิดของครีบกับก้านดอกเห็ด
- ระยะห่างของครีบเห็ดแต่ละครีบ
- ขนาดของครีบเห็ด

1.1.3 ก้านดอกเห็ด

- รูปทรงของก้านดอก
- ลักษณะรากที่อยู่ปลายก้านดอก
- สีของพื้นผิวก้านดอก
  - สีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน
  - สีของพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน
- ลักษณะของพื้นผิวก้านดอก
  - ลักษณะของพื้นผิวบริเวณเหนือวงแหวน
  - ลักษณะของพื้นผิวบริเวณด้านล่างวงแหวน

1.1.4 เยื่อที่หุ้มดอกเห็ด

- ประเภทของเยื่อที่หุ้มดอกเห็ดที่พบ
- สีของเยื่อที่หุ้มดอกเห็ด

1.1.5 วงแหวนของเห็ด

- จำนวนวงแหวนของเห็ด
- รูปร่างของวงแหวน

1.2 ข้อมูลคุณลักษณะประจำตัวของเห็ดอื่นๆ

1.2.1 รอยช้ำบนเห็ด

1.2.2 กลิ่นของเห็ด

1.2.3 สีของรอยพิมพ์สปอร์

1.2.4 ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด

1.2.5 บริเวณที่ที่พบเจอเห็ดเจริญเติบโต

2. ตัวแปรตาม ได้แก่ ประเภทของเห็ด ซึ่งในการวิจัยนี้มี 2 ประเภทคือ เห็ดพิษ และเห็ดที่รับประทานได้

5. กรอบแนวคิดในงานวิจัย

งานวิจัยนี้ศึกษาการจำแนกประเภทของเห็ดระหว่าง เห็ดพิษและเห็ดที่รับประทานได้ โดยใช้เทคนิคการเรียนรู้ของเครื่องเป็นเครื่องมือสำหรับสร้างแบบจำลองในการจำแนกประเภทของเห็ด ข้อมูลที่ใช้นั้นเป็นข้อมูลสาธารณะของเห็ดครีบจำนวน 23 สายพันธุ์ในตระกูล Agaricus และ Lepiota จำนวน 8,124 ชนิด ซึ่งมีการระบุว่าเป็นเห็ดพิษ หรือเห็ดที่รับประทานได้ โดยมีข้อมูลคุณลักษณะของเห็ดจำนวนทั้งหมด 22 คอลัมน์ ซึ่งได้รับมาจากเว็บไซต์ Kaggle

โดยข้อมูลที่ได้รับมานั้นอยู่ในรูปแบบ csv file หลังจากนั้นจึงนำไปเก็บในรูปแบบตาราง (DataFrame) จากนั้นทำการใช้เทคนิคการเรียนรู้ของเครื่องในการสร้างแบบจำลองเพื่อทำนายและจำแนกประเภทของเห็ด ซึ่งเขียนด้วยภาษาโปรแกรม Python โดยใช้ เทคนิคการจัดการฟีเจอร์ คือ การคัดเลือกฟีเจอร์ (Feature Selection) กับการสกัดฟีเจอร์ (Feature extraction) และใช้เทคนิคการเรียนรู้ของเครื่อง ได้แก่ Logistic Regression, Support vector machine, Decision tree, Random forest และ XGBoost ในการแก้ปัญหาประเภท Classification

ในส่วนของการประเมินประสิทธิภาพของแบบจำลอง มีการหาค่า Accuracy, F1 score และ AUC (Area Under Curve) score เพื่อเปรียบเทียบความสามารถในการจำแนกประเภทของเห็ด ในแต่ละแบบจำลอง เพื่อหาแบบจำลองที่มีประสิทธิภาพในการทำนายสูงที่สุด ประกอบกับใช้ training size และ ฟีเจอร์ที่เหมาะสมที่สุด

6. **สมมติฐานในการวิจัย**

6.1 การคัดเลือกฟีเจอร์ด้วยเทคนิค RFE ทำให้สามารถค้นหาฟีเจอร์ที่สำคัญออกมาได้โดยใช้จำนวนฟีเจอร์น้อยกว่าการคัดเลือกด้วยเทคนิค chi-square โดยยังมีประสิทธิภาพในการจำแนกประเภทเห็ดเท่ากัน

6.2 เมื่อใช้เทคนิคการคัดเลือกฟีเจอร์แล้ว แบบจำลอง Random Forest มีประสิทธิภาพในการจำแนกประเภทเห็ดมากที่สุด ที่ training size = 70% โดยใช้จำนวนฟีเจอร์น้อยที่สุด

6.3 เทคนิคการเรียนรู้ของเครื่องสามารถจำแนกประเภทของเห็ด ได้เทียบเท่ามนุษย์เป็นอย่างน้อย

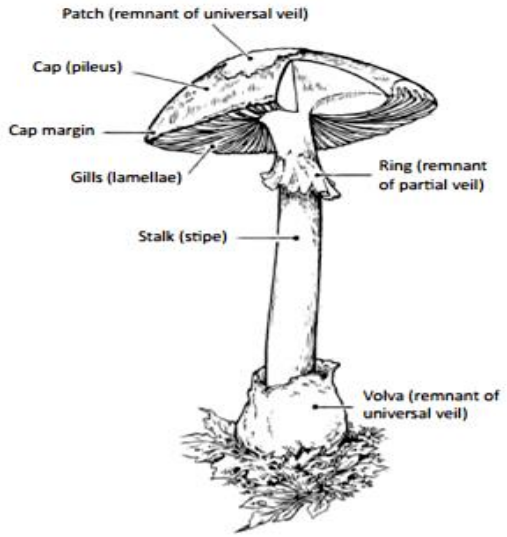
บทที่ 2
ทบทวนวรรณกรรม

ในการวิจัยครั้งนี้ ผู้วิจัยได้ศึกษาเอกสารและงานวิจัยที่เกี่ยวข้อง และได้นำเสนอตามหัวข้อต่อไปนี้

1. ข้อมูลสัณฐานวิทยา (Morphology) และคุณลักษณะประจำตัวของเห็ดครีบ
2. เทคนิคการเรียนรู้ของเครื่อง (Machine Learning Techniques)
3. แบบจำลองอัลกอริธึมที่ใช้ในเทคนิคการเรียนรู้ของเครื่องที่ใช้ในงานวิจัย
4. การแบ่งชุดข้อมูลสำหรับการทดสอบประสิทธิภาพของแบบจำลอง
5. การปรับมาตราส่วนของฟีเจอร์ (Feature scaling)
6. การปรับค่าพารามิเตอร์ (Hyperparameter Tuning)
7. การคัดเลือกฟีเจอร์และการสกัดฟีเจอร์ (Feature Selection and Feature Extraction)
8. งานวิจัยที่เกี่ยวข้อง

1. ข้อมูลสัณฐานวิทยา (Morphology) และคุณลักษณะประจำตัวของเห็ดครีบ

เห็ดครีบ (Gilled mushroom) ซึ่งมีส่วนประกอบที่สำคัญตามภาพประกอบ 1 [19]

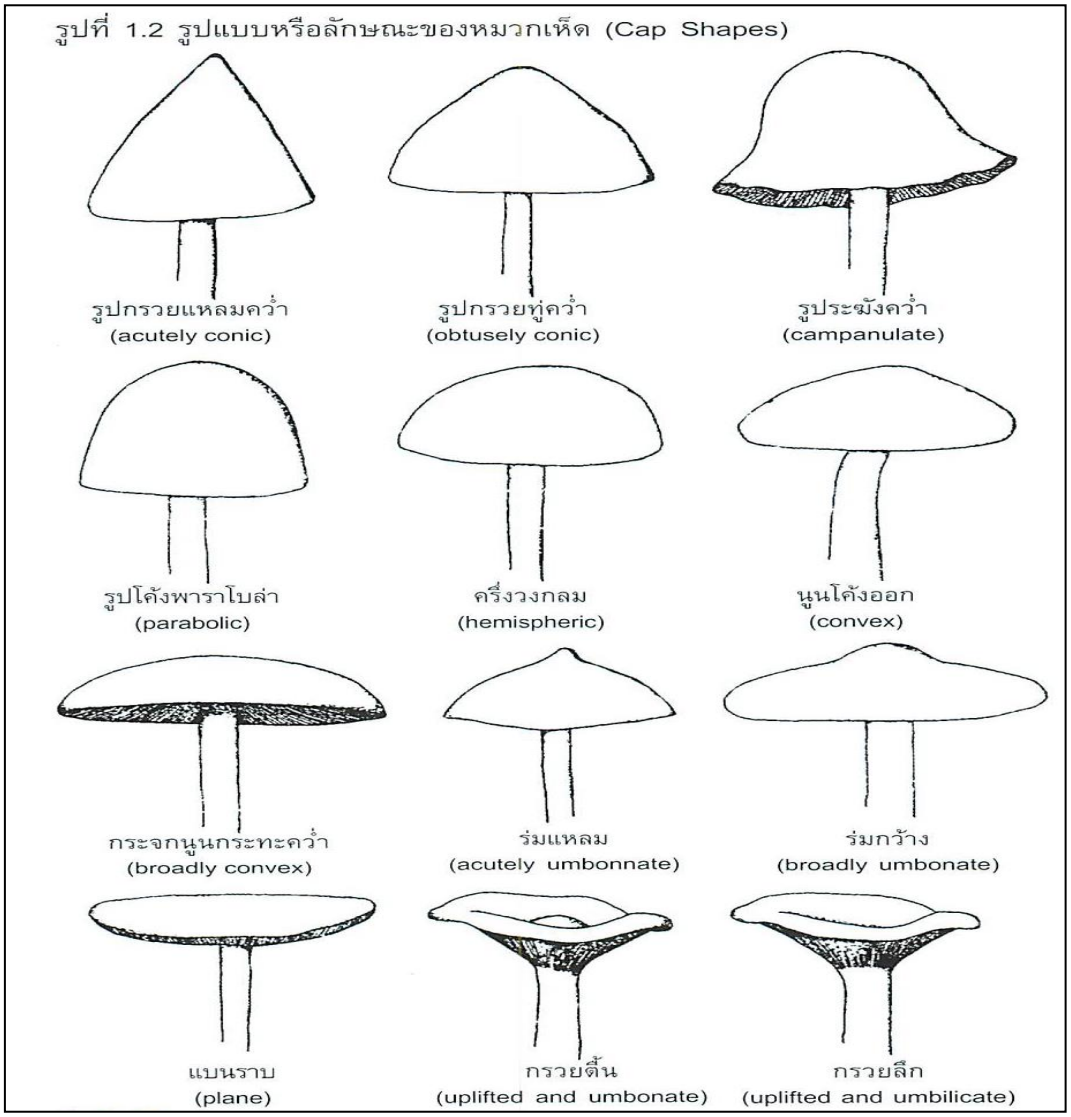


ภาพประกอบ 1 สัณฐานวิทยา (Morphology) ของเห็ดครีบ

ที่มา : https://www.researchgate.net/figure/General-morphology-of-Agaricus-spp-fruiting-body_fig1_275179411

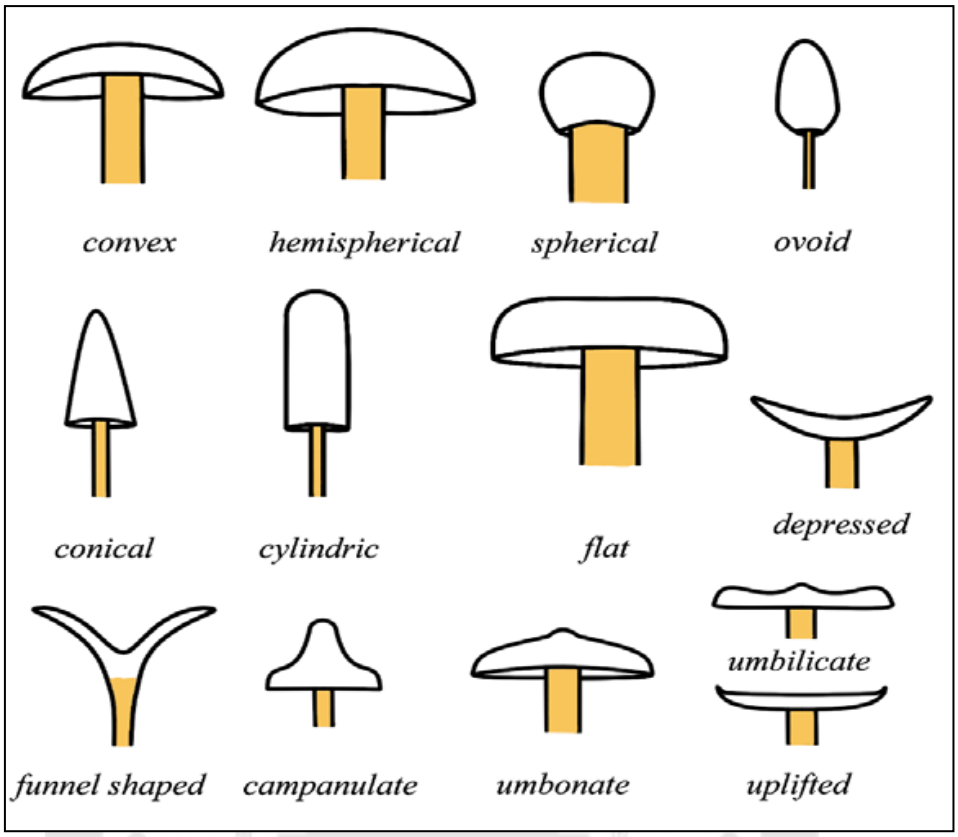
โดยมีส่วนประกอบดังต่อไปนี้ [2, 4-7, 20]

- หมวกเห็ด (Cap) เป็นส่วนที่อยู่ด้านบนสุดของดอกเห็ด ที่เมื่อดอกเห็ดเจริญเติบโตเต็มที่แล้ว ก็จะบานออก โดยอาจมีรูปทรงที่แตกต่างกันออกไปในเห็ดแต่ละชนิด ดังแสดงในภาพประกอบ 2 - 3 เช่น ทรงกรวยคว่ำ (conical), ทรงระฆัง (bell), แผ่นแบนราบ (flat), ทรงกรวยหงาย (sunken), ทรงร่มแหลม (knobbed หรือ umbonate) เป็นต้น โดยบนผิวหมวกเห็ดก็สามารถมีได้หลากหลายรูปแบบ ดังแสดงในภาพประกอบ 4 ได้แก่ เป็นสะเก็ด ผิวเรียบ เป็นต้น และยังมีสีที่แตกต่างกันออกไปได้ ได้แก่ ขาว แดง ส้ม เทา เขียว ม่วง เป็นต้น



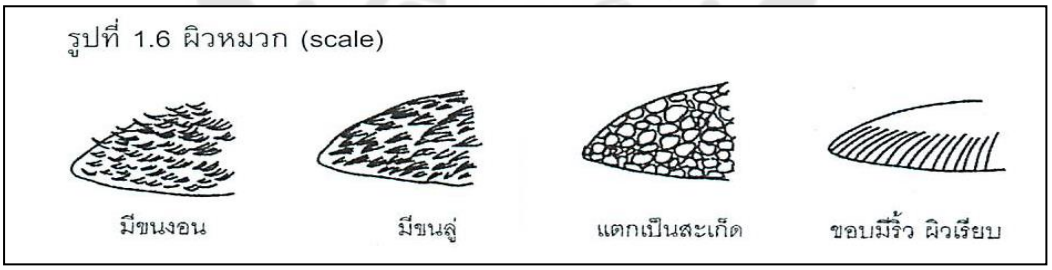
ภาพประกอบ 2 ตัวอย่างรูปทรงของหมวกเห็ด

ที่มา: นันทนา แต้ประเสริฐ (2552), เห็ดพิษอันตรายที่ควรรู้จัก



ภาพประกอบ 3 ตัวอย่างรูปทรงของหมวกเห็ด

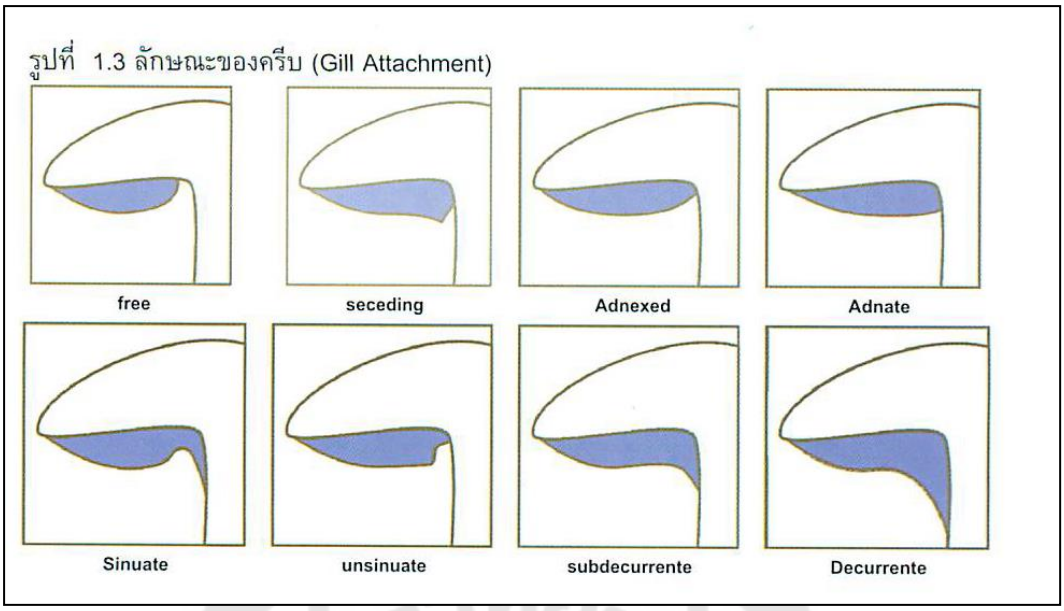
ที่มา: Denchev, C. M., Denchev, T. T., Polemis, E., and Venturella, G. (2013)



ภาพประกอบ 4 ตัวอย่างลักษณะพื้นผิวบนหมวกเห็ด

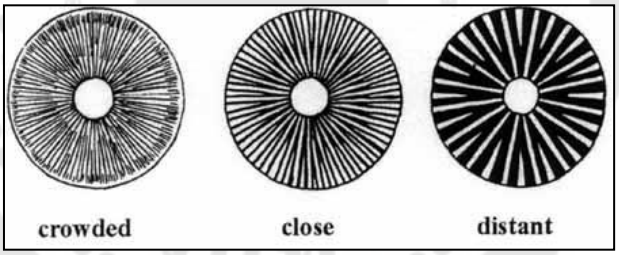
ที่มา: นันทนา แต้ประเสริฐ (2552), เห็ดพิษอันตรายที่ควรรู้จัก

- ครีบเห็ด (Gills) เป็นส่วนที่อยู่ด้านล่างของหมวกเห็ด มีลักษณะเป็นซี่ๆที่เรียงเป็นรัศมีรอบก้านดอก เป็นที่เกิดของสปอร์ของดอกเห็ด โดยเห็ดแต่ละชนิดนั้นมีครีบเห็ดที่แตกต่างกันทั้งในด้านของสี ความหนา การเรียงตัว การยึดติดกับก้านดอก ดังแสดงในภาพประกอบ 5 - 7



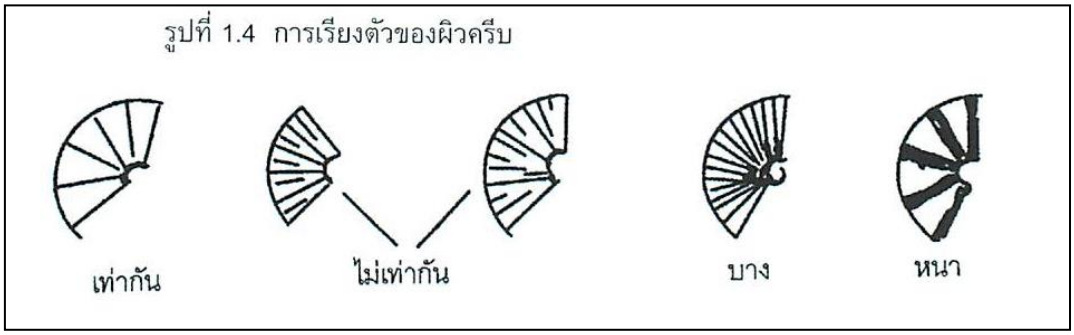
ภาพประกอบ 5 ตัวอย่างลักษณะการยึดติดของครีบเห็ดกับก้านดอก

ที่มา: นันทนา แต้ประเสริฐ (2552), เห็ดพิษอันตรายที่ควรรู้จัก



ภาพประกอบ 6 ตัวอย่างลักษณะการเรียงตัวของครีบเห็ด

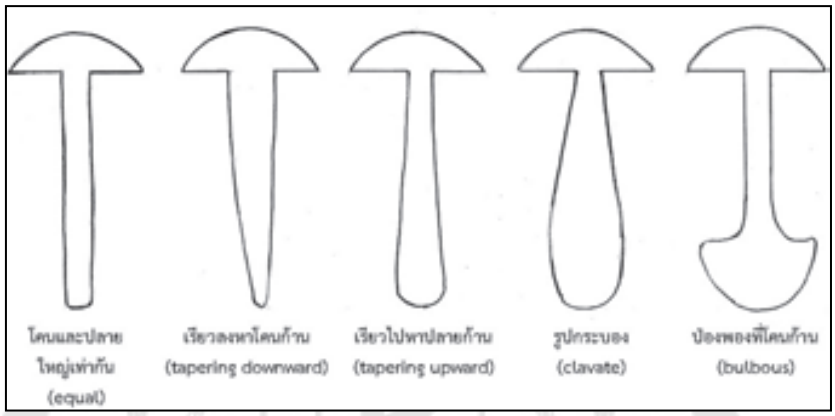
ที่มา: https://laptrinhx.com/mushrooms-classification-part-1-196322162/



ภาพประกอบ 7 ตัวอย่างลักษณะการเรียงตัวของผิวครีบเห็ด และขนาดของครีบเห็ด

ที่มา: นันทนา แต้ประเสริฐ (2552), เห็ดพิษอันตรายที่ควรรู้จัก

- ก้านดอก (stalk) หรืออาจเรียกได้ว่าเป็นลำต้นของดอกเห็ด เป็นส่วนที่ยกหมวกเห็ดให้สูง เพื่อให้สามารถปล่อยสปอร์ที่อยู่ที่ครีบเห็ดนั้นกระจายและปลิวไปได้ไกลๆ โดยก้านเห็ดสามารถมีรูปร่างได้หลายแบบ ได้แก่ รูปทรงกระบอก, รูปทรงเรียวลงหาโคนก้าน อีกทั้งก้านของเห็ดแต่ละชนิดก็สามารถมีลักษณะกับสีที่บริเวณพื้นผิวของก้านดอกที่หลากหลายได้ ดังแสดงในภาพประกอบ 8 นอกจากนี้ที่บริเวณก้านดอกสามารถพบส่วนของเยื่อหุ้มดอกเห็ด (veil) ได้



ภาพประกอบ 8 ตัวอย่างรูปร่างของก้านดอก

ที่มา: อุทัยวรรณ แสงวณิช และคณะ (2556), หนังสือบัญชีรายการทรัพย์สินชีวภาพเห็ด

- เยื่อหุ้มดอกเห็ด (veil) เป็นเนื้อเยื่อบางๆที่ช่วยยึดก้านดอกและขอบของหมวกเห็ดตอนที่เห็ดยังเป็นดอกเล็กๆ โดยเมื่อดอกเห็ดเจริญเติบโตขึ้นจนหมวกเห็ดกางออก เยื่อหุ้มนี้จะขาดออก แต่ยังมีเศษที่ยึดติดอยู่ได้ โดยส่วนที่อยู่ตรงกลางก้านดอกนั้นเรียกว่า วงแหวน (Ring หรือ partial veil) หรืออยู่ที่ส่วนปลายของก้านเรียกว่า Volva (หรือ Universal veil) หรือสามารถพบเยื่อหุ้มดอกเห็ดได้ทั้งสองแบบในเห็ดชนิดเดียวกัน [21]

- กลิ่น (Odor) : เห็ดมีกลิ่นเฉพาะตัวที่แตกต่างกัน หรือไม่มีกลิ่นก็ได้ โดยเห็ดที่รับประทานได้บางชนิดมีกลิ่นเครื่องเทศเมื่อมาทำให้แห้ง (ได้แก่ Lactarius camphoratus (Bull.) Fr.) หรือบางชนิดมีกลิ่นอัลมอนด์ (เช่น เห็ดหล่มกลิ่นอัลมอนด์) ส่วนเห็ดพิษบางชนิดมีกลิ่นเหม็น (เช่น Amanita virosa (Fr.) Bertill) [6]

- รอยช้ำ (Bruise) : รอยช้ำบนเห็ดสามารถเกิดขึ้นได้จากการที่ดอกเห็ดเกิดการฉีกขาด ทำให้มีน้ำยาง (latex) ที่ไม่มีสี, มีสีขาว, สีส้ม หรือสีอื่นๆ ไหลออกมา โดยหากทิ้งไว้นานๆ สีอาจมีการเปลี่ยนแปลงไปได้ [6]

- รอยพิมพ์สปอร์ (Spore print) ติดอยู่ที่ครีบเห็ด โดยสีของรอยพิมพ์สปอร์สามารถมีสีที่แตกต่างไปจากสีของครีบเห็ดและหมวกเห็ด จึงต้องมีการเก็บข้อมูลจากสีของรอยพิมพ์สปอร์เพิ่มเติม โดยวิธีการตรวจสีของรอยพิมพ์สปอร์ เป็นไปดังวิธีต่อไปนี้ ดังแสดงในภาพประกอบ 9 [6]

ขั้นตอนที่ 1 นำดอกเห็ดที่โตเต็มที่และยังสดอยู่ มาตัดแยกส่วนหมวกออกจากก้านด้วยใบมีดที่คม

ขั้นตอนที่ 2 คว่ำหมวกเห็ดทั้งอัน หรือบางส่วนของหมวกเห็ดลงบนกึ่งกลางของแผ่นกระดาษที่ข้างหนึ่งเป็นสีดำและอีกข้างหนึ่งเป็นสีขาว

ขั้นตอนที่ 3 นำฝาจานแก้วหรือถ้วยแก้วมาครอบหมวกเห็ด โดยอาจทิ้งไว้ข้ามคืนเพื่อให้สปอร์ที่อยู่บนผิวครีบตกลงบนแผ่นกระดาษได้เต็มที่และชัดเจน

ขั้นตอนที่ 4 เมื่อครบกำหนดเวลา จึงเปิดภาชนะที่ครอบและหยิบหมวกเห็ดออก จึงเห็นสปอร์ตกอยู่บนกระดาษตามรอยครีบ (กลุ่มเห็ดครีบ) สีสปอร์ที่อ่อน เช่น สีขาว สีครีม สีเหลืองอ่อน สามารถมองเห็นอย่างชัดเจนบนกระดาษสีดำ ส่วนสีสปอร์ที่เข้ม เช่น สีดำ สีเทา สีน้ำตาล สามารถมองเห็นอย่างชัดเจนบนกระดาษสีขาว

สำหรับการบอกสีสปอร์ของเห็ดนั้น ให้ใช้สีรอยพิมพ์สปอร์ที่ตกอยู่บนกระดาษสีขาวและสังเกตภายใต้แสงจากธรรมชาติ สีของรอยพิมพ์สปอร์ในกลุ่มเห็ดครีบ แบ่งออกเป็น 5 กลุ่มสีดังนี้ กลุ่มสีขาวได้แก่ ขาว ครีม เหลืองอ่อน จนถึงสีเหลือง และเขียวอ่อน กลุ่มสีชมพูได้แก่ สีชมพูอ่อนจนถึงชมพูแก่ และน้ำตาลอมชมพู กลุ่มสีน้ำตาล ได้แก่ สีน้ำตาลปนเหลืองจนถึงสีน้ำตาล และสีน้ำตาลปนแดงหรือ สีสนิมเหล็ก กลุ่มสีน้ำตาลปนม่วงจนถึงสีน้ำตาลปนสีชอคโกแลต และกลุ่มสีเทาปนดำจนถึงสีดำ

นอกจากนี้การตรวจสอบรอยพิมพ์ของสปอร์นั้น ยังช่วยให้สังเกตเห็นลักษณะการเรียงตัวของครีบเห็ดได้อีกด้วย

ภาพประกอบ 9 วิธีการเก็บตัวอย่างของรอยพิมพ์สปอร์

ที่มา: อุทัยวรรณ แสงวณิช และคณะ (2556), หนังสือบัญชีรายการทรัพย์สินชีวภาพเห็ด

- ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด (Population): เห็ดสามารถเจริญเติบโตได้หลากหลายรูปแบบ ได้แก่ เกิดเป็นดอกเดี่ยว หรือเป็นกลุ่มเล็กๆ หรือเป็นกระจุกใหญ่ๆ รวมตัวกัน [6]

- บริเวณที่ที่พบเจอเห็ดเจริญเติบโต (Habitat): เห็ดที่ต่างชนิดกันมีการเจริญเติบโตในที่แตกต่างกัน เนื่องจากเห็ดต้องการสารอาหารในการเติบโตและแพร่พันธุ์ที่แตกต่างกัน เช่น เห็ดบางชนิดจะอาศัยอยู่บนลำต้นและตอไม้ที่ตายแล้ว, อาศัยบนดินในป่าผลัดใบและป่าสน, บนพื้นดินในป่าเต็งรัง, บนซากใบไม้ที่ทับถม, บนกิ่งไม้และท่อนไม้ของไม้ผลัดใบ, บนซากใบไม้ เป็นต้น [6]

2. เทคนิคการเรียนรู้ของเครื่อง (Machine Learning Techniques)

เทคนิคการเรียนรู้ของเครื่องเป็นศาสตร์ที่ใช้สร้างการเรียนรู้ให้กับเครื่อง โดยส่วนใหญ่มักใช้หลักการทางคณิตศาสตร์และสถิติ ในการสร้างแบบจำลอง จากข้อมูลต่างๆ (ตัวแปรต้น) นำไปสู่การทำนายผลลัพธ์ (ตัวแปรตาม) ซึ่งเทคนิคการเรียนรู้ของเครื่องนี้แบ่งเป็น 3 ประเภทใหญ่ๆ คือ [22, 23]

2.1 Supervised Learning คือการเรียนรู้แบบมีผู้สอน เป็นเทคนิคที่มนุษย์ซึ่งมีข้อมูลทั้งตัวแปรต้น (input data) และ ข้อมูลผลลัพธ์ (Target) ที่ถูกต้อง แล้วนำไปสอนให้แบบจำลองเพื่อให้แบบจำลองสามารถคำนวณและทำนายผลลัพธ์ได้เมื่อมี input data ที่ไม่รู้จักได้ในอนาคต โดยสามารถแบ่งออกเป็น 2 ลักษณะคือ

- แบบ Classification (ปัญหาเชิงแบ่งประเภท) เมื่อข้อมูลผลลัพธ์มีลักษณะที่เป็นประเภท เช่น yes / no หรือ ชื่อประเภท (Class) ที่ต้องการทำนาย

- แบบ Regression (ปัญหาเชิงถดถอย) เมื่อข้อมูลผลลัพธ์มีลักษณะที่เป็นตัวเลขต่อเนื่อง เช่น การทำนายราคาบ้าน, การทำนายค่าความเข้มข้นของก๊าซ CO_2 ในอากาศ เป็นต้น

2.2 Unsupervised learning คือการเรียนรู้แบบไม่มีผู้สอน เป็นเทคนิคที่มนุษย์ซึ่งมีข้อมูลแค่ input data โดยไม่มีข้อมูลผลลัพธ์ (Target) โดยอาศัยการที่คอมพิวเตอร์ได้เรียนรู้ข้อมูลจากคุณสมบัติหรือลักษณะจำเพาะของข้อมูลที่คล้ายกัน แล้วทำการจัดกลุ่มให้อยู่ด้วยกัน และสร้างกฎและขอบเขตเพื่อจำแนกข้อมูล เช่น งาน Clustering ที่ต้องการแบ่งกลุ่มลูกค้าที่มีพฤติกรรมคล้ายกันให้อยู่ในกลุ่มเดียวกัน เพื่อนำเสนอโปรโมชั่นการขายสินค้า

2.3 Reinforcement learning คือการเรียนรู้แบบเสริมกำลัง มีหลักการทำงานเสมือนกับการที่มนุษย์เรียนรู้บางสิ่งบางอย่างด้วยการลองผิดลองถูก และมีการเรียนรู้เกิดขึ้นระหว่างทางว่าการกระทำไหนดีหรือไม่ดี หรืออาจมีคะแนนเข้ามาเกี่ยวข้อง โดยคำนวณว่าในสถานการณ์นั้นควร

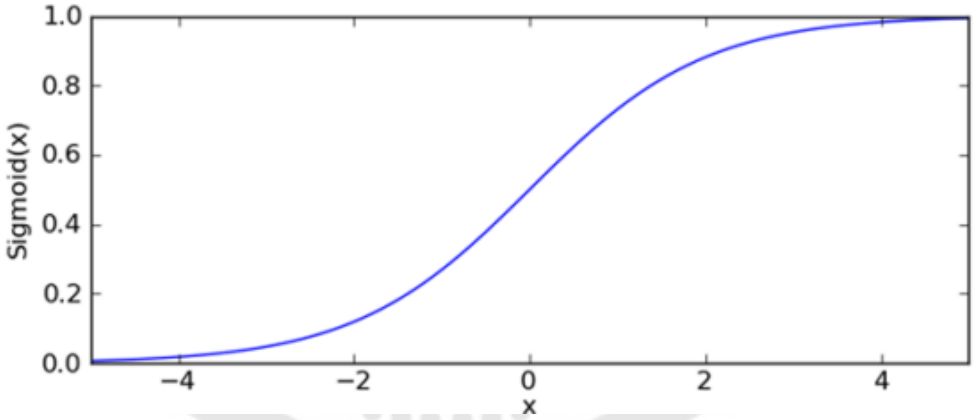
ทำอย่างไรเพื่อให้ได้คะแนนที่สูงที่สุด เช่น AlphaGo ที่ถูกพัฒนาขึ้นมาจนสามารถชนะแชมป์โกะของโลก เป็นต้น [24]

3. แบบจำลองอัลกอริธึมที่ใช้ในเทคนิคการเรียนรู้ของเครื่องที่ใช้ในงานวิจัย

แบบจำลองเทคนิคการเรียนรู้ของเครื่องมีหลายแบบจำลอง โดยขอกล่าวถึงเพียง 5 แบบจำลองที่ใช้ในงานวิจัยนี้ ซึ่งเป็น Supervised Learning (การเรียนรู้แบบมีผู้สอน) ที่ใช้ในงาน Classification ได้แก่ Logistic Regression, Support Vector Machine (SVM), Decision tree, Random Forest และ XGBoost เท่านั้น ดังต่อไปนี้

3.1 Logistic Regression เป็นแบบจำลองที่ใช้การคำนวณทางคณิตศาสตร์โดยการหาค่าความน่าจะเป็น โดยใช้สมการ Sigmoid function (สมการที่ 1) และตามภาพประกอบ 10 [23]

$$\sigma(X) = \frac{1}{1+ e^{-X}} \quad (1)$$

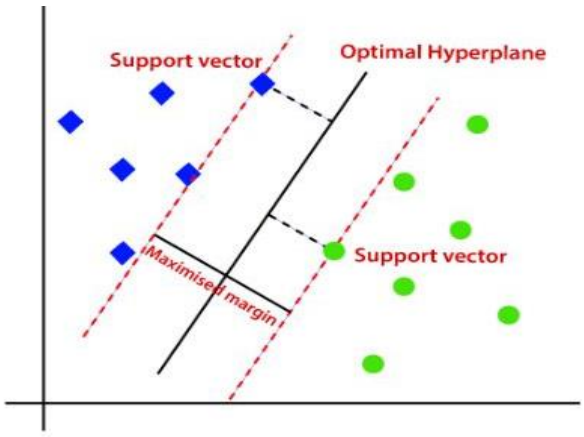


ภาพประกอบ 10 กราฟของสมการ Sigmoid function

ที่มา: Harrington P, 2012

โดยหากค่าความน่าจะเป็น $\sigma(X)$ คำนวณออกมาได้ตั้งแต่ 0.5 จะเป็น class 1 หากน้อยกว่า 0.5 จะเป็น class 0

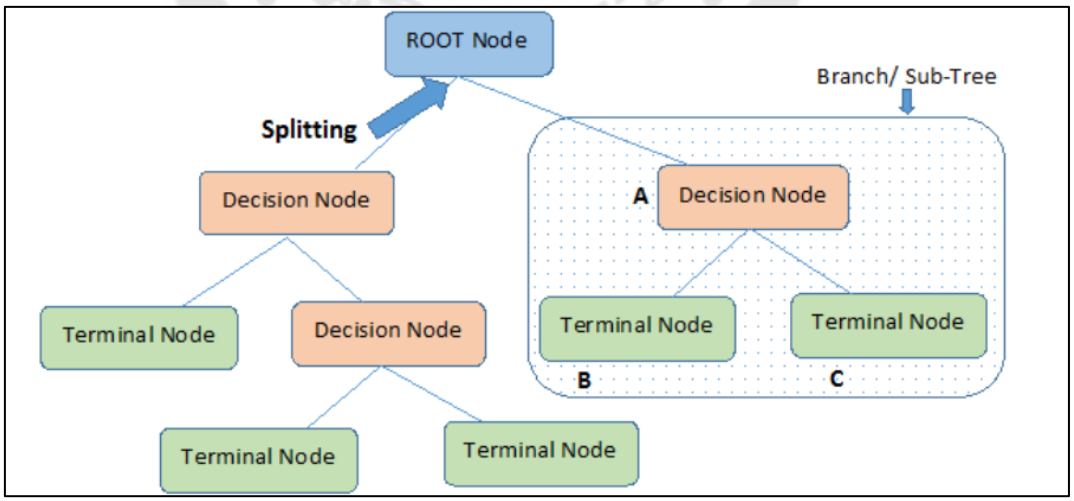
3.2 Support Vector Machine (SVM) เป็นแบบจำลองที่มีหลักการพื้นฐานคือพยายามหาเส้นแบ่งระหว่าง class ต่างๆ (เรียกว่าเส้น hyperplane) ของข้อมูลที่กระจายบน feature space เพื่อแบ่งข้อมูลออกเป็นกลุ่มๆ โดยเป็นเส้นที่ทำให้มีขอบเขต (margin) ห่างจาก support vector ที่มากที่สุด ดังภาพประกอบ 11 [25]



ภาพประกอบ 11 การแบ่งกลุ่มข้อมูล 2 ประเภท ด้วยเส้นแบ่งที่มีขอบเขต (margin) ห่างจาก support vector มากที่สุด

ที่มา: https://www.analyticsvidhya.com/blog/2021/04/insight-into-svm-support-vector-machine-along-with-code/

3.3 Decision tree หรือเรียกว่าต้นไม้ตัดสินใจ เป็นแบบจำลองที่ใช้ทั้งงาน classification และ regression โดยหลักการการทำงานของแบบจำลองนี้เกิดจากสร้างลำดับชั้นของการตัดสินใจขึ้นมาจากข้อมูลในตัวแปรต้น โดยเริ่มต้นจาก Root node (คือเงื่อนไขแรกของการตัดสินใจ) แล้วแตกกิ่งการตัดสินใจ (branch) ไปที่เงื่อนไขลำดับถัดไป (Decision node หรือ Internal node) จนสามารถทำนายผลลัพธ์สุดท้ายออกมาได้ (Leaf node หรือ Terminal node) ตามที่แสดงในภาพประกอบ 12 ซึ่งแบบจำลองนี้จะแตกกิ่งจนผลลัพธ์ที่ได้มีความถูกต้องสูงที่สุด [26, 27]



ภาพประกอบ 12 หลักการทำงานของแบบจำลอง Decision tree

ที่มา : https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/

โดยหลักการในการจัดลำดับ node ของงาน Classification มีหลายวิธี ได้แก่ [22]

- ID3 (Iterative Dichotomiser 3) เป็นวิธีที่คำนวณจากค่า Entropy และ Information gain ซึ่งหากเมื่อคำนวณค่าในแต่ละฟีเจอร์แล้ว node ที่มีค่า information gain สูง จะถูกเลือกก่อนเป็นลำดับแรก การคำนวณเป็นไปตามสมการที่ 2 – 3 [22]

$$\text{Entropy(node)} = \sum P(\text{node}) * \sum(-P(b_i) * \log_2 P(b_i)) \quad (2)$$

$$\text{Information gain (node)} = \text{Entropy(previous node)} - \text{Entropy(node)} \quad (3)$$

เมื่อ b_i คือ แต่ละกิ่งของ node นั้น

$P(b_i)$ คือ ความน่าจะเป็นในแต่ละกิ่ง ซึ่งได้จากการคำนวณจำนวน class ของผลลัพธ์ของแต่ละกลุ่ม แล้วหารด้วยจำนวน class ทั้งหมดในกิ่งนั้น

- CART (Classification And Regression Tree) เป็นวิธีที่คำนวณจากค่า Gini impurity index ซึ่งหากเมื่อคำนวณค่าในแต่ละฟีเจอร์แล้ว node ที่มีค่า Gini impurity index ต่ำ จะถูกเลือกก่อนเป็นลำดับแรก การคำนวณเป็นไปตามสมการที่ 4 – 5 [22]

$$\text{Gini(bi)} = 1 - \left(\frac{c_1|b_i}{c_1+c_2+\cdots}\right)^2 - \left(\frac{c_2|b_i}{c_1+c_2+\cdots}\right)^2 - \cdots \quad (4)$$

$$\text{Gini(node)} = \left(\frac{c|b_1}{c|node}\right) * \text{Gini(b1)} + \left(\frac{c|b_2}{c|node}\right) * \text{Gini(b2)} + \ldots \quad (5)$$

เมื่อ b_i คือ แต่ละกิ่งของ node นั้น

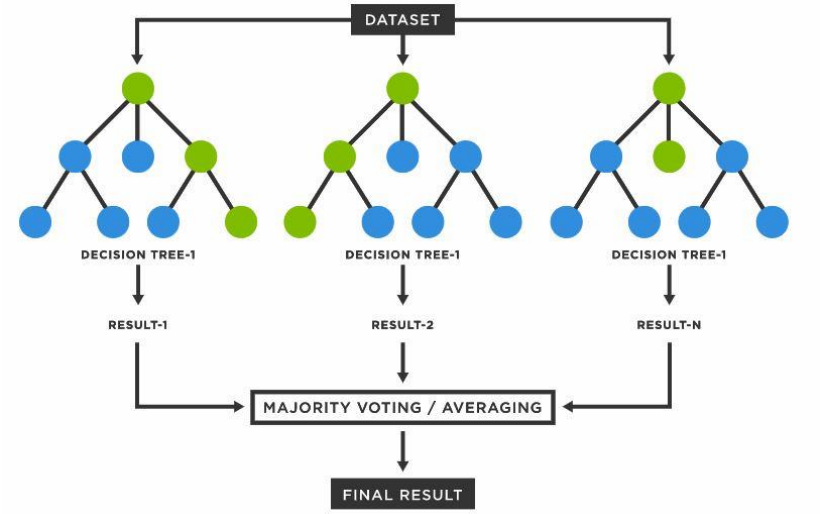
$c_j \mid b_i$ คือ จำนวน class ของแต่ละกลุ่มในกิ่งที่ i

$c \mid b_i$ คือ จำนวน class ทั้งหมดในกิ่งที่ i

c | node คือ จำนวน class ทั้งหมดของ node นั้น

3.4 Random Forest เป็นแบบจำลองแบบ Ensemble model ของแบบจำลองต้นไม้ตัดสินใจ (Decision Tree) คือนำ Decision Tree หลายๆ อัน มาช่วยกันทำนายผล ซึ่งใช้เทคนิคที่เรียกว่า Bagging (Bootstrap aggregating) โดยแต่ละ Decision Tree นั้นมีการสุ่มฟีเจอร์ที่

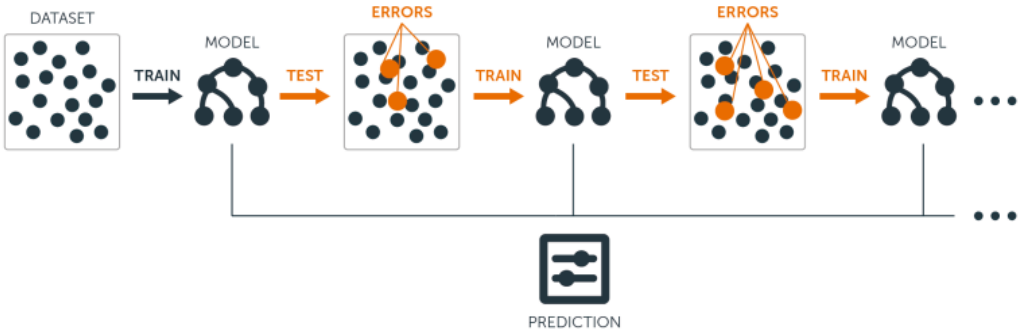
ไม่เหมือนกันและการสุ่มตัวอย่างก็ไม่เหมือนกัน (โดยสุ่มตัวอย่างประมาณ 2/3 ของ ชุดข้อมูลทั้งหมด) ผลลัพธ์สุดท้ายของการทำนายเกิดจากผลโหวตที่มากที่สุด (Majority vote) ของ Decision Tree แต่ละต้น ตามภาพประกอบ 13 [28]



ภาพประกอบ 13 หลักการทำงานของแบบจำลอง Random Forest

ที่มา: https://www.tibco.com/reference-center/what-is-a-random-forest

3.5 XGBoost (Extreme Gradient Boosting) คือ แบบจำลองที่นำเอา Decision Tree มา หลายๆ ต้น มาเชื่อมต่อกันเป็นลำดับ (Sequence) โดยที่แต่ละ Decision tree เรียนรู้จาก Error ของ Decision Tree ต้นก่อนหน้า (เรียกเทคนิคนี้ว่า Ensemble model แบบ Boosting) ทำให้ความแม่นยำของในการทำนายผลลัพธ์ มีความแม่นยำมากขึ้นเรื่อยๆ เมื่อมีการเรียนรู้ของ Decision Tree ต่อเนื่องกันจนมีความลึกมากพอ และแบบจำลองจะหยุดเรียนรู้เมื่อไม่เหลือรูปแบบของ Error จาก Decision Tree ก่อนหน้าให้เรียนรู้แล้ว ตามภาพประกอบ 14 [29, 30]



ภาพประกอบ 14 หลักการทำงานของแบบจำลอง XGBoost

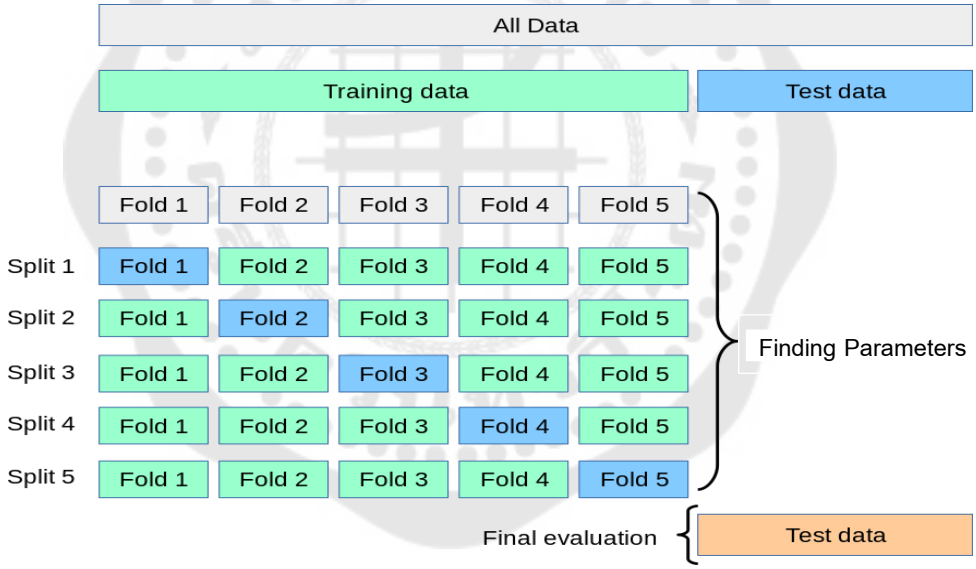
ที่มา: https://blog.bigml.com/2017/03/14/introduction-to-boosted-trees/

4. การแบ่งชุดข้อมูลสำหรับการทดสอบประสิทธิภาพของแบบจำลอง

ในเทคนิคการเรียนรู้ของเครื่องนั้นจำเป็นต้องมีการแบ่งชุดข้อมูลออกเป็นส่วนๆ โดยเทคนิคการแบ่งชุดข้อมูลมี 2 ประเภทหลัก คือ [31-33]

4.1 Train/Test Split เป็นการแบ่งชุดข้อมูลทั้งหมดเป็น 2 ส่วน คือ ชุดข้อมูลสำหรับสร้างแบบจำลอง (Training Set) และชุดข้อมูลสำหรับทดสอบแบบจำลอง (Test Set) ซึ่งโดยทั่วไปอัตราส่วนของชุดข้อมูลสำหรับสร้างแบบจำลองต่อชุดข้อมูลสำหรับทดสอบแบบจำลอง นั้นเป็น 80:20, 70:30, 60:40 หรือ 50:50 [34]

4.2 Cross-validation เป็นการแบ่งข้อมูลสำหรับใช้ในการสร้างและทดสอบประสิทธิภาพแบบจำลองที่ได้จากการเรียนรู้ของเครื่อง โดยมีการแบ่งส่วนของ Training set เป็นชุดย่อยๆ คือ แบ่ง training set ออกเป็น 5 ส่วน โดยใช้ 4 ส่วนในการสร้างแบบจำลอง และอีก 1 ส่วนใช้ในการทดสอบแบบจำลอง และวนใช้ข้อมูลจนครบทั้งหมด 5 ครั้ง ตามภาพประกอบ 15



ภาพประกอบ 15 การแบ่งชุดข้อมูลสำหรับการทดสอบประสิทธิภาพของแบบจำลอง

ที่มา: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation

5. การปรับช่วงขอบเขตของฟีเจอร์ (Feature scaling)

การปรับช่วงขอบเขตของฟีเจอร์เป็นขั้นตอนที่มีความสำคัญขั้นตอนหนึ่งในการจัดเตรียมข้อมูลก่อนสร้างแบบจำลอง โดยช่วยให้ฟีเจอร์ที่เป็นตัวเลขที่มีช่วงขอบเขตที่แตกต่างกัน ถูกแปลงให้มาอยู่ภายในช่วงขอบเขตเดียวกัน หรือเรียกว่าเป็นการทำ Normalization โดยเทคนิคที่ใช้ในการปรับช่วงขอบเขตมีหลายเทคนิค ได้แก่ [35]

5.1 Min Max Scaler เป็นเทคนิคที่ทำให้ช่วงขอบเขตของค่าภายในฟีเจอร์ถูกแปลงให้อยู่ในช่วง 0 ถึง 1 โดยข้อเสียคือมีความไวต่อจุดข้อมูลที่เป็นค่าผิดปกติ (outliers) โดยสูตรคำนวณของเทคนิคนี้เป็นตามสมการที่ 6

$$\text{Xnew} = \frac{X - Xmin}{Xmax - Xmin} \quad (6)$$

Xmin คือ ค่าที่น้อยที่สุดของฟีเจอร์นั้นๆ
Xmax คือ ค่าที่มากที่สุดของฟีเจอร์นั้นๆ

5.2 Standard Scaler เป็นเทคนิคที่ทำให้ช่วงขอบเขตของค่าภายในฟีเจอร์ถูกแปลงให้อยู่ในช่วง -1 ถึง 1 โดยเหมาะกับข้อมูลฟีเจอร์ที่มีการกระจายตัวแบบปกติ (Normal distribution) เท่านั้น ข้อเสียคือมีความไวต่อจุดข้อมูลที่เป็นค่าผิดปกติ (outliers) โดยสูตรคำนวณของเทคนิคนี้เป็นตามสมการที่ 7

$$\text{Xnew} = \frac{X - \mu}{\sigma} \quad (7)$$

μ คือ ค่าเฉลี่ยของฟีเจอร์นั้นๆ
σ คือ ค่าเบี่ยงเบนมาตรฐานของฟีเจอร์นั้นๆ

5.3 Robust Scaler เป็นเทคนิคที่เหมาะสมกับฟีเจอร์ที่มีค่าผิดปกติ (outliers) โดยหลังจากปรับช่วงขอบเขตแล้ว จุดที่เป็น outlier ก็ยังคงเป็น outlier โดยสูตรคำนวณของเทคนิคนี้เป็นตามสมการที่ 8 [36]

$$\text{Xnew} = \frac{X - median}{p75 - p25} \quad (8)$$

median คือ ค่ามัธยฐานของฟีเจอร์นั้นๆ
p75 คือ ค่า percentile ที่ 75 ของฟีเจอร์นั้นๆ
p25 คือ ค่า percentile ที่ 25 ของฟีเจอร์นั้นๆ

6. การปรับค่าพารามิเตอร์ (Hyperparameter Tuning)

Hyperparameters คือ พารามิเตอร์ต่าง ๆ ที่ผู้ใช้สามารถกำหนดเองได้ก่อนที่แบบจำลองสร้างการเรียนรู้ โดยการทำ Hyperparameter Tuning (หรือเรียกว่า Hyperparameter Optimization) นั้นมีวัตถุประสงค์เพื่อหาว่า Set ของ Hyperparameters ใดที่เหมาะสมสำหรับแบบจำลองประเภทนั้น ๆ ที่ทำให้แบบจำลองมีความถูกต้องและความแม่นยำ ที่สูงที่สุดหรือมีค่า Loss ที่ต่ำที่สุด [37, 38]

โดยการเทคนิคปรับค่าพารามิเตอร์พื้นฐานมี 3 เทคนิค [37, 38] ได้แก่

4.1 Manual Search

วิธีนี้เป็นการเลือกค่า Hyperparameter ของแบบจำลองจากประสบการณ์และความคิดเห็นส่วนบุคคล โดยทำการสร้างแบบจำลองขึ้นมาจากค่าที่เลือกและวัดประสิทธิภาพของแบบจำลองไปเรื่อย ๆ จนกว่าพบค่าประสิทธิภาพที่ดี

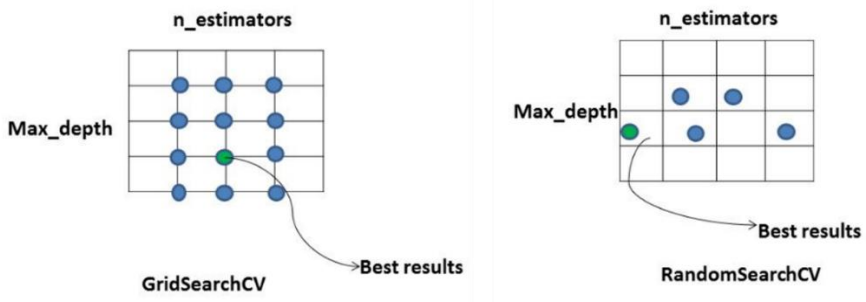
4.2 Grid search

วิธีนี้เป็นการแสกนหาค่า Hyperparameter โดยเริ่มต้นจากการที่กำหนดค่า Hyperparameters แต่ละชนิด ไว้ล่วงหน้าหลายๆค่า ซึ่งค่าทั้งหมดอยู่ในรูปแบบเมทริกซ์ (matrix) แล้วจึงนำ Hyperparameter ทุกตัวที่กำหนดเอาไว้ไปทดสอบกับแบบจำลองที่จำเพาะกับ Hyperparameter ชุดเหล่านั้น จนได้ประสิทธิภาพที่ดีที่สุด จึงมีการสรุปค่า Hyperparameter ออกมา ซึ่งการทำ Grid search มักใช้งานร่วมกับการทำ Cross-Validation ด้วย เรียกว่า GridSearchCV [22, 37, 38]

4.3 Random search

วิธีนี้เป็นการสุ่มแสกนหาค่า Hyperparameter ที่มีการกำหนดค่าไว้ล่วงหน้า โดยการสุ่มนั้นไม่มีการเรียงลำดับ แต่สามารถสลับไปมาขึ้นกับค่าที่สุ่มได้ ซึ่งการทำ Random search มักใช้งานร่วมกับการทำ Cross-Validation ด้วย เรียกว่า RandomizedSearchCV โดยวิธีนี้ใช้เวลาที่น้อยกว่า GridSearchCV แต่ประสิทธิภาพของแบบจำลองที่ได้ก็อาจต่ำกว่า GridSearchCV เล็กน้อย [22, 37-39]

โดยตัวอย่างของการทำ GridSearchCV และ RandomizedSearchCV ของแบบจำลอง Random Forest เป็นไปตามภาพประกอบ 16



ภาพประกอบ 16 รูปแบบของหลักการระหว่าง GridSearchCV และ RandomizedSearchCV
ที่มา: https://www.naukri.com/learning/articles/hyperparameter-tuning-beginners-tutorial/#hyper

7. การคัดเลือกฟีเจอร์และการสกัดฟีเจอร์ (Feature Selection and Feature Extraction)

7.1 การคัดเลือกฟีเจอร์ (Feature Selection) [40-43] มีความสำคัญต่อการสร้างแบบจำลองการเรียนรู้ของเครื่อง โดยช่วยเลือกฟีเจอร์ที่มีความสำคัญต่อการทำนายผลลัพธ์ ส่งผลให้ลดเวลาในการเรียนรู้ของแบบจำลอง และสามารถป้องกันการได้แบบจำลองที่มีความซับซ้อนจนมากเกินไป หรือที่เรียกว่า เกิดการ Overfitting ได้ โดยเครื่องมือและวิธีการคัดเลือกฟีเจอร์สามารถแบ่งเป็น 3 ประเภท คือ

7.1.1 Filter methods เป็นวิธีที่ช่วยคัดเลือกฟีเจอร์ก่อนนำเข้าสู่แบบจำลองเพื่อสร้างการเรียนรู้ โดยมีการพิจารณาความสัมพันธ์ระหว่างฟีเจอร์และผลลัพธ์ โดยฟีเจอร์ใดที่มีความสัมพันธ์กับผลลัพธ์น้อย จะถูกคัดออกก่อนเข้าสู่แบบจำลอง ข้อดีของการใช้เทคนิคนี้คือ การคัดเลือกฟีเจอร์จำนวนมากๆ นั้นใช้เวลาไม่นาน แต่อย่างไรก็ตามข้อเสียคือ ฟีเจอร์แต่ละตัวนั้น ถูกวัดคะแนนความสัมพันธ์แยกกันกับการสร้างแบบจำลอง ดังนั้นฟีเจอร์ที่ได้นั้นอาจไม่เหมาะกับการนำไปใช้ในการสร้างแบบจำลอง ตัวอย่างของ Filter methods เช่น การคำนวณ Pearson's Correlation, Chi-square, Mutual Information, ANOVA เป็นต้น

7.1.2 Wrapper methods เป็นการคัดเลือกฟีเจอร์โดยคำนวณจากการเรียนรู้และทดสอบในการสร้างแบบจำลองที่เลือกไว้ จึงมีข้อดีคือเป็นการหาฟีเจอร์ที่เหมาะสมกับแบบจำลองที่ต้องการ แต่อย่างไรก็ตามมีความเสี่ยงที่อาจทำให้เกิดภาวะ overfiting มากกว่าวิธี Filter methods และเนื่องจากมีการคำนวณหลายครั้ง จึงใช้เวลาในการคำนวณมากกว่า Filter techniques ตัวอย่างของ Wrapper method คือ Recursive Feature Elimination (RFE)

7.1.3 Embedded methods เป็นเทคนิคการคัดเลือกฟีเจอร์ของแบบจำลอง ซึ่งเกิดขึ้นในระหว่างการสร้างการเรียนรู้ของแบบจำลอง ได้แก่ การสุ่มเลือกฟีเจอร์ในกระบวนการสร้าง

แบบจำลอง Random Forest, ค่า Weight ของแบบจำลอง logistic regression หลังจากเสร็จสิ้นกระบวนการเรียนรู้

7.2 การสกัดฟีเจอร์ (Feature Extraction) [41, 42]

เป็นเทคนิคที่ใช้ในการลดมิติของฟีเจอร์ลง โดยมีการแปลงฟีเจอร์เดิมให้เป็นฟีเจอร์ใหม่ที่มีมิติลดลง ช่วยให้การจำแนกข้อมูลมีประสิทธิภาพมากขึ้น โดยใช้ทรัพยากรของคอมพิวเตอร์น้อย แต่อย่างไรก็ตามการสกัดฟีเจอร์นี้ไม่ได้คำนวณความสัมพันธ์ระหว่างฟีเจอร์กับผลลัพธ์ที่ต้องการทำนาย ซึ่งส่งผลให้ข้อมูลฟีเจอร์ที่สำคัญต่อการทำนายผลลัพธ์หายไปในกระบวนการสกัดได้ ตัวอย่างเทคนิคที่ใช้ ได้แก่ เทคนิคการวิเคราะห์องค์ประกอบหลัก (Principal Component Analysis: PCA)

8. งานวิจัยที่เกี่ยวข้อง

การทบทวนวรรณกรรมของงานวิจัยนี้ได้ทำการศึกษาค้นคว้างานวิจัยที่เกี่ยวข้องกับการจำแนกประเภทเห็ดระหว่างเห็ดพิษและเห็ดที่รับประทานได้ โดยงานวิจัยที่เกี่ยวข้อง สามารถแบ่งกลุ่มได้ 4 กลุ่ม โดยมีรายละเอียดดังนี้ และสามารถสรุปได้ดังตาราง 2

กลุ่มที่ 1 งานที่ไม่มีการทำความสะอาดข้อมูล และไม่มีการคัดเลือกฟีเจอร์หรือการสกัดฟีเจอร์

8.1 บทความวิจัยเรื่อง Prediction of Whether Mushroom is Edible or Poisonous Using Back-propagation Neural Network. โด Eyad Sameh Alkronz, Khaled A. Moghayer, Mohamad Meimeh, Mohannad Gazzaz, Bassem S. Abu-Nasser และ Samy S. Abu-Naser [44]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 1 แบบจำลองคือ Artificial Neural Network (ANN) ซึ่งได้ปรับปรุง neurons ใหม่ โดยมีการตั้งชื่อว่า Just Neural Network (JNN) ซึ่งประกอบไปด้วยชั้นแรกสุด (input layer) จำนวน 22 neurons, ชั้นตรงกลาง (hidden layers) จำนวน 3 hidden layers ที่มีขนาด 2x1x3 neurons, และชั้นแสดงผลลัพธ์จากการทำนาย (output layer) จำนวน 1 neuron โดยขั้นตอนในงานวิจัยนี้มี 4 ขั้นตอนด้วยกัน คือ

ขั้นตอน 1 การเลือกตัวแปรต้นและผลลัพธ์ (ตัวแปรตาม) สำหรับตัวแปรตามได้ใช้ประเภทของเห็ด ส่วนตัวแปรต้นใช้ฟีเจอร์ทุกอย่างที่เหลือจำนวน 22 ฟีเจอร์โดยไม่มีการตัดทิ้ง

ขั้นตอน 2 การแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม คือเป็นตัวเลข

ขั้นตอน 3 การแบ่งข้อมูล ในที่นี้มีการแบ่งออกเป็นชุดข้อมูลสำหรับการสร้างแบบจำลอง (training samples) จำนวน 5,724 ข้อมูล และชุดข้อมูลสำหรับการตรวจสอบความถูกต้อง (validation smaples) จำนวน 2,400 ข้อมูล

ขั้นตอน 4 การสร้างการเรียนรู้ให้กับ Just Neural Network (JNN) ซึ่งทำทั้งหมด 161,501 epochs โดยในขั้นตอนนี้มีการแสดงค่าความสำคัญของแต่ละฟีเจอร์ ด้วยค่า Relative Importance ซึ่งพบว่า gill-size (ขนาดของครีบเห็ด) มีค่ามากที่สุด รองลงมาเป็น ring-number (จำนวนวงแหวน) และน้อยที่สุดคือ cap-surface (ลักษณะของพื้นผิวหมวกเห็ด)

ขั้นตอน 5 การประเมินผลแบบจำลอง ด้วยค่า accuracy เพียงอย่างเดียว โดยผลลัพธ์ที่ได้คือ accuracy เท่ากับ 99.25 %

8.2 บทความวิจัยเรื่อง Accuracy of classification poisonous or edible of mushroom using naïve bayes and K-nearest neighbors โดย Roni Hamonangan, Meidika Bagus Saputro และ Cecep Bagus Surya Dinata Karta Atmaja [45]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง 2 แบบจำลอง คือ naïve bayes และ K-nearest neighbors โดยไม่มีการวัดค่า feature importance ใดๆ และการวัดประสิทธิภาพของแบบจำลองโดยใช้เพียงแค่ Accuracy ค่าเดียวเท่านั้น ผลการทดลองคือแบบจำลอง K-nearest neighbors เป็นแบบจำลองที่ให้ Accuracy เท่ากับ 100% และแบบจำลอง naïve bayes ให้ Accuracy เท่ากับ 90.2%

8.3 บทความวิจัยเรื่อง Edible and Poisonous Mushrooms Classification by Machine Learning Algorithms โดย Kemal Tutuncu, Ilkay Cinar, Ramazan Kursun และ Murat Koklu [46]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 4 แบบจำลอง คือ Decision Tree, Naïve Bayes, AdaBoost และ Support Vector Machine โดยไม่มีการวัดค่า feature importance ใดๆ และการทดสอบประสิทธิภาพของแบบจำลองนั้นเป็นการใช้ทั้งชุดข้อมูล และนำไปทำด้วยวิธี Cross validation โดยใช้จำนวน 10 Fold โดยในแต่ละรอบมี 1 Fold เป็น test set ซึ่งประสิทธิภาพที่วัดได้เป็นค่า Cross validation score เท่านั้นซึ่งประกอบไปด้วยค่า Accuracy, Recall, Precision และ F1 score

สำหรับผลการศึกษาพบว่าแบบจำลอง AdaBoost ให้ค่าประสิทธิภาพทั้ง Accuracy, Recall, Precision และ F-1 score ที่สูงที่สุดคือ 100% ส่วนแบบจำลอง Decision Tree, Naïve Bayes และ Support Vector Machine ให้ค่า Accuracy เท่ากับ 98.82%, 90.99% และ 99.98% ตามลำดับ

กลุ่มที่ 2 งานที่มีการทำความสะอาดข้อมูล แต่ไม่มีการคัดเลือกฟีเจอร์หรือการสกัดฟีเจอร์

8.4 บทความวิจัยเรื่อง Mushroom classification using machine-learning techniques โดย Omar Tarawneh, Monther Tarawneh, Yousef Sharrab และ Moath husni [47]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 6 แบบจำลอง คือ Logistic Regression, Decision tree, K-Nearest Neighbors, Support Vector Machines, Naïve Bayes, Artificial Neural Network และแบบจำลองที่เกิดจากการรวมกันของ K-Nearest Neighbors, Artificial Neural Network กับ Support Vector Machines โดยไม่มีการวัดค่า feature importance ใดๆ แต่ได้มีการทำความสะอาดข้อมูลคือ มีการตัดฟีเจอร์ที่ชื่อว่า “stalk-root” (ลักษณะรากที่อยู่ปลายก้านดอก) และ “veil-type” (ชนิดของเยื่อหุ้มดอกเห็ด) ออก เนื่องจาก ในฟีเจอร์ “stalk-root” มีค่าว่าง (missing) จำนวนมาก ส่วน “veil-type” นั้นมีค่าของตัวแปรเพียงค่าเดียวจึงไม่มีความสำคัญต่องานจำแนกประเภทนี้

สำหรับผลการวัดประสิทธิภาพของแบบจำลองต่างๆ ด้วยค่า Accuracy นั้นพบว่าแบบจำลอง K-Nearest Neighbors นั้น มีค่า accuracy สูงที่สุดอยู่ที่ 94% ซึ่งเท่ากันกับแบบจำลอง Decision tree ส่วนแบบจำลองที่มีค่า accuracy รองลงมา ได้แก่ Artificial Neural Network มีค่า accuracy เท่ากับ 93% และแบบจำลอง Support Vector Machines มีค่า accuracy เท่ากับ 91% และเมื่อสร้างแบบจำลองที่เกิดจากการรวมกันของ K-Nearest Neighbors, Artificial Neural Network กับ Support Vector Machines นั้น พบว่าให้ค่า accuracy ดีขึ้นเล็กน้อย (ประมาณ 94%)

กลุ่มที่ 3 งานที่ไม่มีการทำความสะอาดข้อมูล แต่มีการคัดเลือกฟีเจอร์หรือการสกัดฟีเจอร์

8.5 บทความวิจัยเรื่อง Behavioural Features for Mushroom Classification โดย Shuhaida Ismail, Amy Rosshaida Zainal และ Aida Mustapha [48]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 1 แบบจำลองคือ Decision tree (J48 algorithm) โดยใช้

เครื่องมือ WEKA (Waikato Environment for Knowledge Analysis) โดยขั้นตอนในงานวิจัยนี้มี 4 ขั้นตอนด้วยกัน (โดยไม่มีการแบ่งชุดข้อมูล) คือ

ขั้นตอน 1 การแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม คือเป็นตัวเลข

ขั้นตอน 2 การสกัดฟีเจอร์ใช้เทคนิค Principal Component Analysis (PCA) เพื่อลดมิติของข้อมูลลง โดยไม่แสดงว่าใช้จำนวน component เท่าไร

ขั้นตอน 3 การสร้างการเรียนรู้ให้กับ Decision tree (J48 algorithm) ซึ่งจากแบบจำลองนี้ได้สุ่มเลือกฟีเจอร์มาได้ 5 ฟีเจอร์ คือ odor (กลิ่นของเห็ด), spore_print_color (สีของรอยพิมพ์สปอร์), gill-size (ขนาดของครีบเห็ด), stalk_root (ลักษณะรากที่อยู่ปลายก้านดอก), Bruise (รอยช้ำบนเห็ด) อีกทั้งมีการแสดงค่าความสำคัญของแต่ละฟีเจอร์ โดย odor (กลิ่นของเห็ด) มีค่าความสำคัญที่มากที่สุด และ veil-type (ชนิดของเยื่อหุ้มดอกเห็ด) ไม่มีความสำคัญเลย

ขั้นตอน 4 การประเมินผลแบบจำลอง มีการแสดงเป็น classification report และ confusion matrix โดยผลลัพธ์ก็คือ จากจำนวนข้อมูลทั้งหมด 8,124 ข้อมูลนั้น ไม่มีการจำแนกประเภทผิดเลย (accuracy, precision, F-measure = 100%)

8.6 บทความวิจัยเรื่อง Mushroom Classification Using ANN and ANFIS Algorithm โดย S.K. Verma และ M. Dutta [49]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 3 แบบจำลองคือ Naïve Bayes, ANN (Artificial Neural Network) และ ANFIS (Adaptive Neuro Fuzzy inference System) โดยขั้นตอนในงานวิจัยนี้มี 4 ขั้นตอนด้วยกัน คือ

ขั้นตอน 1 การแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม ด้วยการลดข้อมูลรบกวน และ ทำ Normalization

ขั้นตอน 2 การสกัดฟีเจอร์เพื่อลดมิติของข้อมูลลง โดยไม่ระบุเทคนิคที่ใช้

ขั้นตอน 3 การสร้างการเรียนรู้ให้กับแบบจำลองทั้งสามแบบจำลอง นี้ โดยในแต่ละแบบจำลองที่ใช้นั้นมีการทดลองโดยใช้ขนาดของ training set จำนวน 5 ขนาด คือ 40%, 50%, 60%, 70% และ 80%

ขั้นตอน 4 การประเมินผลแบบจำลอง ด้วยค่า Accuracy, Mean absolute error และ Kappa statistic โดยผลลัพธ์คือ แบบจำลอง Naïve Bayes และ ANN ที่มีขนาด training set เท่ากับ 70% นั้นให้ประสิทธิภาพดีกว่าขนาด training set อื่น คือ accuracy เท่ากับ 96.8173%, Mean Absolute Error เท่ากับ 0.033 และ Kappa Statistic เท่ากับ 0.9316 โดยทั้งสอง

แบบจำลองนี้มีประสิทธิภาพในทุกๆ training set เท่ากัน; และแบบจำลอง ANFIS ให้ค่าประสิทธิภาพที่ดีที่สุดกว่าทั้ง 2 แบบจำลองก่อนหน้า โดยที่ training set เท่ากับ 80% นั้นมีประสิทธิภาพดีกว่า training set อื่น คือ accuracy เท่ากับ 99.8763%, Mean Absolute Error เท่ากับ 0.0008 และ Kappa Statistic เท่ากับ 0.9980 และเมื่อขนาดของ training set เพิ่มขึ้นประสิทธิภาพก็เพิ่มขึ้น

8.7 บทความวิจัยเรื่อง A Comparative Study on Mushroom Classification using Supervised Machine Learning Algorithms โดย Kanchi Tank [50]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 6 แบบจำลอง คือ Logistic Regression, Decision tree, K-Nearest Neighbors, Support Vector Machines, Naïve Bayes และ Random Forest

โดยมีกระบวนการทั้งหมด 5 ขั้นตอนหลัก คือ

ขั้นตอน 1 การจัดรูปแบบข้อมูลให้เหมาะสมและการแสดงข้อมูล (Data Pre-processing and Exploratory Data Analysis) โดยมีการแปลงข้อมูลเป็นตัวเลขด้วย Label encoder และนำไปแสดงเป็น violin plot เพื่อตรวจสอบการกระจายตัวของข้อมูล จากนั้นจึงใช้เทคนิค one-hot encoder เพื่อให้เหมาะสมกับข้อมูลที่เป็นรูปแบบ categorical data แล้วตรวจสอบ correlation ของแต่ละฟีเจอร์

ขั้นตอน 2 การแบ่งชุดข้อมูลเป็น training set และ test set โดยใช้สัดส่วน 70:30

ขั้นตอน 3 การปรับช่วงขอบเขตของข้อมูล (Feature scaling) ด้วยเทคนิคที่เรียกว่า standard scaler

ขั้นตอน 4 การสกัดและคัดเลือกฟีเจอร์ของเห็ดโดยการทำ Principal Component Analysis (PCA) โดยกำหนดให้ใช้จำนวน component เท่ากับ 2 เท่านั้น

ขั้นตอน 5 การสร้างการเรียนรู้ของแบบจำลอง และการประเมินผล (ด้วยการวัดค่า Accuracy (เป็นหลัก), Precision, Recall, F1 Score, True Negative Rate (TNR), False Positive Rate (FPR), False Negative Rate (FNR) และ AUC score จาก ROC Curve)

ผลการทดสอบประสิทธิภาพ คือ แบบจำลอง Random Forest และ K-Nearest Neighbor ให้ค่า accuracy สำหรับ test set ที่สูงที่สุด คือ 92.21% และ 92.04% ตามลำดับ ส่วน Support Vector Machine ให้ค่า accuracy เท่ากับ 91.92%, Logistic Regression ให้ค่า accuracy เท่ากับ 90.28%, Naïve Bayes ให้ค่า accuracy เท่ากับ 89.75% และ Decision Tree ให้ค่า accuracy เท่ากับ 89.54%

8.8 บทความวิจัยเรื่อง Edible Mushroom Identification Using Machine Learning โดย K. Kousalya, B. Krishnakumar, S. Boomika, N. Dharati, and N. Hemavathy [3]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษ โดยใช้เทคนิคการเรียนรู้ของเครื่อง ด้วยเครื่องมือที่ชื่อว่า WEKA (Waikato Environment for Knowledge Analysis) โดยมีกระบวนการทั้งหมด 5 ขั้นตอนหลัก คือ การคัดเลือกชุดข้อมูล, การจัดรูปแบบข้อมูลให้เหมาะสม (pre-processing), การคัดเลือกฟีเจอร์ของเห็ด (Feature selection), การสร้างการเรียนรู้ของแบบจำลอง และการประเมินผล (ด้วยการวัดค่า Accuracy, Precision และ Recall) โดยผลการทดลองพบว่า จากทั้ง 22 คุณลักษณะ (Features) นั้น คุณลักษณะที่ให้ค่า feature importance ที่มากที่สุดคือ Gill-color (สีของครีบเห็ด)
และจากทั้ง 4 แบบจำลองคือ Logistic regression, Naive bayes, Decision tree (C4.5) และ Support Vector Machine นั้น พบว่า Decision tree (C4.5) เป็นแบบจำลองที่ให้ค่า Accuracy ที่มากที่สุด คือเท่ากับ 93.34%

กลุ่มที่ 4 งานที่มีการทำความสะอาดข้อมูล และมีการคัดเลือกฟีเจอร์หรือการสกัดฟีเจอร์

8.9 บทความวิจัยเรื่อง Classification of Mushrooms to Detect their Edibility Based on Key Attributes โดย V. Vanitha, M.N. Ahil และ N. Rajathi [51]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษ ด้วยเทคนิคการเรียนรู้ของเครื่อง ด้วยเครื่องมือที่ชื่อว่า WEKA (Waikato Environment for Knowledge Analysis) ด้วยภาษา python โดยมีกระบวนการทั้งหมด 3 ขั้นตอนหลัก คือ

ขั้นตอน 1 การเตรียมข้อมูล (Pre-Processing) โดยมีการแปลงข้อมูลในฟีเจอร์ต่างๆ ให้เป็นตัวเลข และมีการตัดฟีเจอร์ชื่อ “Stalk_root” (ลักษณะรากที่อยู่ปลายก้านดอก) ทิ้งไป เนื่องจากพบค่าว่าง (missing) จำนวนมาก

ขั้นตอน 2 การคัดเลือกฟีเจอร์ ใช้ 2 วิธีแล้วนำมาสรุปฟีเจอร์ที่นำไปใช้

ซึ่งวิธีที่ 1 คือ Wrapper method เป็นการคัดเลือกฟีเจอร์โดยคำนวณจากการเรียนรู้และทดสอบในการสร้างแบบจำลองที่เลือกไว้ โดยใช้เทคนิคการเลือกฟีเจอร์คือ BestFirst ของแบบจำลองคือ Naïve Bayes, Decision Tree (วิธี J48) และ Bagging

สำหรับวิธีที่ 2 Filter method ใช้เทคนิค information gain attribute evaluator ซึ่งต้องมีการใช้การทำ Rank search ร่วมด้วย สุดท้ายแล้วพบว่าฟีเจอร์ที่พบบ่อยที่สุดในวิธีทั้งสองวิธีที่กล่าวมานั้นมี จำนวน 2 ฟีเจอร์ คือ odor (กลิ่นของเห็ด) และ spore_print_color (สีของรอยพิมพ์สปอร์)

ขั้นตอน 3 สร้างแบบจำลองและประเมินประสิทธิภาพ ในขั้นตอนนี้ใช้แบบจำลองเพียงแค่ 1 แบบจำลอง คือ J48 Algorithm (Decision tree) และมีการวัดประสิทธิภาพของการทำ Cross validation จำนวน 10 Fold ด้วยค่า Precision, Recall และ F-Measure

โดยผลลัพธ์ที่ได้คือแบบจำลอง J48 Algorithm (Decision tree) มีประสิทธิภาพดี โดยมีค่า Precision, Recall และ F-Measure ประมาณ 99%

8.10 บทความวิจัยเรื่อง Performance Comparison of Mushroom Types Classification Using K-Nearest Neighbor Method and Decision Tree Method โดย Nadya Chitayae และ Andi Sunyoto [52]

งานวิจัยนี้ได้ทำการศึกษาการจำแนกประเภทเห็ดระหว่างเห็ดรับประทานได้และเห็ดพิษด้วยเทคนิคการเรียนรู้ของเครื่อง 2 แบบจำลอง คือ Decision tree และ K-Nearest Neighbors โดยมีกระบวนการทั้งหมด 3 ขั้นตอนหลัก คือ

ขั้นตอน 1 การจัดรูปแบบข้อมูลให้เหมาะสมและการแสดงข้อมูล (Data Pre-processing and Exploratory Data Analysis) มีการลบแถวข้อมูลที่มีค่าว่าง (missing) จำนวน 2,480 แถว โดยค่าว่างนั้นอยู่ในฟีเจอร์ stalk root (ลักษณะรากที่อยู่ปลายก้านดอก) ทำให้เหลือข้อมูล 5,644 แถวข้อมูล จากนั้นมีการแปลงข้อมูลเป็นตัวเลข แล้วตรวจสอบ correlation ของแต่ละฟีเจอร์

ขั้นตอน 2 การสกัดและคัดเลือกฟีเจอร์ของเห็ด โดยจากการพิจารณาค่า correlation ของฟีเจอร์กับประเภทของเห็ด จึงเลือกฟีเจอร์จำนวน 8 ฟีเจอร์ คือ habitat, gill-size, population, cap-color, cap-shape, veil-color, cap-suface และ ring-number ไปใช้ในขั้นตอนต่อไป

ขั้นตอน 3 การสร้างการเรียนรู้ของแบบจำลอง และการประเมินผล โดยในขั้นตอนนี้มีการทำ Cross validation โดยใช้จำนวน 10 Fold โดยในแต่ละรอบมี 1 Fold เป็น test set ซึ่งประสิทธิภาพที่วัดได้เป็นค่า Cross validation score เท่านั้นซึ่งประกอบไปด้วยค่า Accuracy, Recall, Precision และ F1 score สำหรับแบบจำลอง K-Nearest Neighbors นั้น มีการปรับค่า k_neighbor value ในช่วง 1 ถึง 25 โดย k_neighbor value ที่ดีที่สุดเท่ากับ 11 (accuracy เท่ากับ 89.61%, precision เท่ากับ 89.83%, Recall เท่ากับ 89.61% และ F1 score เท่ากับ 89.72%) ส่วนแบบจำลอง Decision tree (CART algorithm) ไม่มีการระบุการปรับค่าพารามิเตอร์ใดๆ โดยมีประสิทธิภาพจากค่า cross validation score ดีกว่าแบบจำลอง K-Nearest Neighbors คือ accuracy เท่ากับ 91.93%, precision เท่ากับ 92.27%, Recall เท่ากับ 91.93% และ F1 score เท่ากับ 92.10%

ตาราง 2 สรุปงานวิจัยที่เกี่ยวข้อง

No.	Title	Year	Clean data	Feature selection/ Extraction	Training size	scaler	Model	Feature Importance	CV	Max performance
กลุ่มที่ 1 : งานที่ไม่มีทั้งการ clean data และ Feature selection / Extraction										
1	Prediction of Whether Mushroom is Edible or Poisonous Using Back-propagation Neural Network	2019	X	X	70%	X	JNN : 161,501 Epoch	MAX Relative importance = “gill-size”	N/A	Accuracy = 99.25%
2	Accuracy of classification poisonous or edible of mushroom using naïve bayes and K-nearest neighbors	2021	X	X	N/A	X	NB, KNN	X	N/A	Accuracy = 100% (KNN)
3	Edible and Poisonous Mushrooms Classification by Machine Learning Algorithms	2021	X	X	N/A	X	DT, NB, SVM, AdaBoost	X	10 Fold	CV score (Accuracy) = 100% (AdaBoost)
กลุ่มที่ 2 : งานที่มีการ clean data แต่ไม่มี Feature selection / Extraction										
4	Mushroom classification using machine-learning techniques	2022	Drop “stalk-root”, “veil-type”	X	N/A	X	LR, DT, KNN, SVM, NB, ANN, COMBINE the decision of KNN, ANN กับ SVM	X	N/A	Accuracy = 94% (KNN, DT, COMBINE)

ตาราง 2 (ต่อ)

No.	Title	Year	Clean data	Feature selection/ Extraction	Training size	scaler	Model	Feature Importance	CV	Max performance
กลุ่ม 3 : งานที่ไม่มีการ clean data แต่มี Feature selection / Extraction										
5	Behavioural Features for Mushroom Classification	2018	X	PCA (ไม่ระบุ n_component)	N/A	X	DT (J48)	Feature Importance => odor (Max)	N/A	accuracy, precision, F-measure = 100%
6	Mushroom Classification Using ANN and ANFIS Algorithm	2018	X	Feature Extraction (ไม่ระบุเทคนิค)	Vary: 40, 50, 60, 70, 80%	X	NB, ANN, ANFIS	X	N/A	ANFIS with train size = 80% Accuracy = 99.8763%
7	A Comparative Study on Mushroom Classification using Supervised Machine Learning Algorithms	2021	X	PCA (n_component = 2)	70%	standard	LR, NB, DT, SVM, KNN, Random Forest	X	N/A	Accuracy = 92.21% (Random Forest)
8	Edible Mushroom Identification Using Machine Learning	2022	X	Importance = 12 features (Gill-color)	N/A	X	LR, NB, DT(C4.5), SVM	X	N/A	Accuracy = 93.34% (DT(C4.5))

ตาราง 2 (ต่อ)

No.	Title	Year	Clean data	Feature selection/ Extraction	Training size	scaler	Model	Feature Importance	CV	Max performance
กลุ่ม 4 : งานที่มีทั้งการ clean data และ Feature selection / Extraction										
9	Classification of Mushrooms to Detect their Edibility Based on Key Attributes	2020	Drop “stalk-root”	1.1 Wrapper (BestFIrst) : NB, DT(J48), Bagging 1.2 Filter (Information gain attribute evaluator + Rank search) --> Odor, Spore_print_color	N/A	X	DT(J48)	X	N/A	F-Measure = 99%
10	Performance Comparison of Mushroom Types Classification Using K-Nearest Neighbor Method and Decision Tree Method	2020	dropna 2,480 แถว (stalk-root)	หา correlation ของฟีเจอร์ กับ class-> 8 Features	N/A	X	DT(CART) : ไม่ปรับ parameter. KNN : ปรับ k_neighbor value =11	X	10 Fold	CV score (Accuracy) = 91.93% (DT (CART))

บทที่ 3
วิธีดำเนินการวิจัย

ในการวิจัยครั้งนี้ ผู้วิจัยได้ดำเนินการตามขั้นตอนดังนี้

1. ชุดข้อมูลและการแบ่งชุดข้มูล
2. การออกแบบวิธีการดำเนินงานวิจัย
3. การประมวลผลและการวิเคราะห์ข้อมูล
4. การสร้างแบบจำลอง และประเมินประสิทธิภาพแบบจำลอง
5. การพัฒนาแบบจำลองเป็นเว็บแอปพลิเคชัน

1. ชุดข้อมูลและการแบ่งชุดข้อมูล

1.1 ชุดข้อมูล

งานวิจัยนี้ใช้ชุดข้อมูลสาธารณะของเห็ดครีบจำนวน 23 สายพันธุ์ในตระกูล Agaricus และ Lepiota จำนวน 8,124 แถว ซึ่งมีการระบุว่าเป็นเห็ดพิษ หรือเห็ดที่รับประทานได้ ซึ่งได้รับมาจากเว็บไซต์ kaggle [16] โดยเป็นข้อมูลที่ Jeff Schlimmer ซึ่งเป็นผู้บริจาคให้กับ UCI Machine Learning Repository ได้ดึงมาจากข้อมูลในหนังสือ The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf [17, 18]

ชุดข้อมูลประกอบด้วยข้อมูลตัวแปรจำนวน 23 คอลัมน์ ได้แก่ สีของหมวกเห็ด, ลักษณะพื้นผิวบนหมวกเห็ด, รูปทรงของหมวกเห็ด, สีของครีบเห็ด, ความแนบชิดของครีบกับก้านดอกเห็ด, ระยะห่างของครีบเห็ดแต่ละครีบ, ขนาดของครีบเห็ด, รูปทรงของก้านดอก, ลักษณะรากที่อยู่ปลายก้านดอก, สีและลักษณะของพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน และด้านล่างของวงแหวน, ประเภทของเยื่อที่หุ้มดอกเห็ดที่พบ, สีของเยื่อที่หุ้มดอกเห็ด, จำนวนวงแหวนของเห็ด, รูปร่างของวงแหวน, รอยช้ำบนเห็ด, กลิ่นของเห็ด, สีของรอยพิมพ์สปอร์, ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด, บริเวณที่พบเจอเห็ดเจริญเติบโต และประเภทของเห็ด โดยมีจำนวนทั้งหมด 8,124 แถว ดังตาราง 3

ตาราง 3 รายละเอียดของชุดข้อมูลเห็ดครีบ

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
1	สีของหมวกเห็ด (cap-color)	n คือ brown (น้ำตาลเข้ม) b คือ buff (สีน้ำตาลอมเหลือง) c คือ cinnamon (สีน้ำตาลอบเชย) g คือ gray (เทา) r คือ green (เขียว) p คือ pink (ชมพู) u คือ purple (ม่วง) e คือ red (แดง) w คือ white (ขาว) y คือ yellow (เหลือง)
2	ลักษณะพื้นผิวบนหมวกเห็ด (cap-surface)	f คือ fibrous (มีขน) g คือ grooves (เป็นร่อง) y คือ scaly (เป็นเกล็ด) s คือ smooth (เรียบ)
3	รูปทรงของหมวกเห็ด (cap-shape)	b คือ bell (ทรงระฆังคว่ำ) c คือ conical (ทรงกรวยคว่ำ) x คือ convex (นูนโค้ง) f คือ flat (แบนราบ) k คือ knobbed (หรือ umbonate คือมีลักษณะเป็นร่มที่มีจุกอยู่ตรงกลางด้านบนของหมวก) s คือ sunken (ทรงกรวยหงาย)

ตาราง 3 (ต่อ)

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
4	สีของครีบเห็ด (gill-color)	k คือ black (ดำ) n คือ brown (น้ำตาลเข้ม) b คือ buff (น้ำตาลอมเหลือง) h คือ chocolate (ชอคโกแลต) g คือ gray (เทา) r คือ green (เขียว) o คือ orange (ส้ม) p คือ pink (ชมพู) u คือ purple (ม่วง) e คือ red (แดง) w คือ white (ขาว) y คือ yellow (เหลือง)
5	ความแนบชิดของครีบกับก้านดอกเห็ด (gill-attachment)	f คือ free (ครีบไม่แนบชิดกับก้านดอกเห็ด) a คือ attached (ครีบแนบชิดกับก้านดอกเห็ดในระดับเสมอกันตลอด) d คือ descending (ครีบแนบชิดกับก้านดอกเห็ดโดยมีการแผ่ขยายออกลงมาจนเกือบถึงกลางก้านดอกเห็ด) n คือ notched (ครีบแนบชิดกับก้านดอกเห็ด ในระดับเสมอกัน แต่มีรอยบากเล็กน้อย
6	ระยะห่างของครีบเห็ดแต่ละครีบ (gill-spacing)	c คือ close (อยู่ติดกัน) w คือ crowded (อยู่ติดกันอย่างหนาแน่น โดยมองไม่เห็นว่าเป็นเส้นๆ) d คือ distant (อยู่ห่างกันอย่างหลวมๆ)
7	ขนาดของครีบเห็ด (gill-size)	b คือ broad (เส้นหนา) n คือ narrow (เส้นบาง)

ตาราง 3 (ต่อ)

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
8	รูปทรงของก้านดอก (stalk-shape)	e คือ enlarging (บริเวณก้านดอกเห็ดด้านล่างมีขนาดใหญ่กว่าส่วนบนของก้านดอกเห็ด) t คือ tapering (ก้านดอกเห็ดมีลักษณะเรียวเล็กลงไปหาโคน)
9	ลักษณะรากที่อยู่ปลายก้านดอก (stalk-root)	b คือ bulbous (บริเวณโคนก้านดอกเห็ดมีลักษณะป่องเป็นกระเปาะ) c คือ club (บริเวณโคนก้านดอกเห็ดนูนออกเล็กน้อย) u คือ cup (บริเวณโคนก้านดอกเห็ดมี volva หุ้มอยู่เป็นถ้วย) e คือ equal (เรียบเสมอกันตลอดก้านดอกเห็ด) r คือ rooted (ปลายโคนก้านมีลักษณะเป็นรากเรียวยาว แต่ไม่มีการแตกแขนง) ? คือ missing (ไม่มีการระบุลักษณะของรากที่อยู่ปลายก้านดอก) z คือ rhizomorphs (มีรากแตกแขนงออกจากโคนก้าน)
10	สีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน (stalk-color-above-ring)	n คือ brown (น้ำตาลเข้ม) b คือ buff (น้ำตาลอมเหลือง) c คือ cinnamon (น้ำตาลอบเชย) g คือ gray (เทา) o คือ orange (ส้ม) p คือ pink (ชมพู) e คือ red (แดง) w คือ white (ขาว) y คือ yellow (เหลือง)

ตาราง 3 (ต่อ)

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
11	สีของพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน (stalk-color-below-ring)	n คือ brown (น้ำตาลเข้ม) b คือ buff (น้ำตาลอมเหลือง) c คือ cinnamon (น้ำตาลอบเชย) g คือ gray (เทา) o คือ orange (ส้ม) p คือ pink (ชมพู) e คือ red (แดง) w คือ white (ขาว) y คือ yellow (เหลือง)
12	ลักษณะของพื้นผิวก้านดอกบริเวณเหนือวงแหวน (stalk-surface-above-ring)	f คือ fibrous (เป็นเส้นใย / เยื่อ) y คือ scaly (เป็นเกล็ด) k คือ silky (มีขนนิ่มเหมือนสำลี) s คือ smooth (เรียบ)
13	ลักษณะของพื้นผิวก้านดอกบริเวณด้านล่างวงแหวน (stalk-surface-below-ring)	f คือ fibrous (เป็นเส้นใย / เยื่อ) y คือ scaly (เป็นเกล็ด) k คือ silky (มีขนนิ่มเหมือนสำลี) s คือ smooth (เรียบ)
14	ประเภทของเยื่อที่หุ้มดอกเห็ดที่พบ (veil-type)	p คือ partial (คือมีวงแหวนบริเวณก้านดอกเห็ด) u คือ universal (คือมมี volva หุ้มโคนดอกเห็ด ซึ่งมีลักษณะเป็นถ้วย)
15	สีของเยื่อที่หุ้มดอกเห็ด (veil-color)	n คือ brown (น้ำตาลเข้ม) o คือ orange (ส้ม) w คือ white (ขาว) y คือ yellow (เหลือง)

ตาราง 3 (ต่อ)

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
16	จำนวนวงแหวนของเห็ด (ring-number)	n คือ none (ไม่พบวงแหวน) o คือ one (มีวงแหวน 1 วง) t คือ two (มีวงแหวน 2 ชั้นซ้อนกัน)
17	รูปร่างของวงแหวน (ring-type)	c คือ cobwebby (เป็นเส้นใยคล้ายใยแมงมุมที่ยื่นออกไปยังขอบของหมวกเห็ด) e คือ evanescent (มีลักษณะที่เป็นเศษหรือผงที่ติดอยู่บนก้านดอก หรือห้อยลงมาจากขอบหมวก) f คือ flaring (แผ่นคว่ำที่มีลักษณะกางออก) l คือ large (มีขนาดใหญ่ โดยคลุมทั้งก้านดอกเห็ดไว้) n คือ none (ไม่มีวงแหวน) p คือ pendant (เป็นแผ่นห้อยลง) s คือ sheathing (เป็นแผ่นหุ้มขึ้นด้านบน) z คือ zone (เป็นแถบบางๆ โดยมีอยู่รอบๆ ก้านดอก)
18	รอยช้ำบนเห็ด (bruise)	t คือ bruises (มีรอยช้ำ) f คือ no (ไม่มี)
19	กลิ่นของเห็ด (odor)	a คือ almond (กลิ่นอัลมอนด์) l คือ anise (กลิ่นโป๊ยกั๊ก หรือกลิ่นหอมหวานอ่อนๆ) c คือ creosote (กลิ่นน้ำมันถ่านหิน) y คือ fishy (เหม็นคาว) f คือ foul (กลิ่นเหม็นไม่พึงประสงค์ทั่วไป) m คือ musty (กลิ่นเหม็นอับ) n คือ none (ไม่มีกลิ่น) p คือ pungent (กลิ่นสารเคมีที่ฉุนแสบจมูก) s คือ spicy (กลิ่นเผ็ดคล้ายเครื่องเทศที่แสบจมูก)

ตาราง 3 (ต่อ)

ลำดับ	ข้อมูลตัวแปร (Variable)	คำอธิบายข้อมูล (Description)
20	สีของรอยพิมพ์สปอร์ (spore-print-color)	k คือ black (ดำ) n คือ brown (น้ำตาลเข้ม) b คือ buff (น้ำตาลอมเหลือง) h คือ chocolate (ชอคโกแลต) r คือ green (เขียว) o คือ orange (ส้ม) u คือ purple (ม่วง) w คือ white (ขาว) y คือ yellow (เหลือง)
21	ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด (population)	a คือ abundant (อยู่เป็นกลุ่มใหญ่มากๆ ระรานตา) c คือ clustered (เกาะกันเป็นช่อเห็ด และมีหลายช่อในบริเวณเดียวกัน) n คือ numerous (อยู่รวมกันเป็นกลุ่ม) s คือ scattered (อยู่แยกกัน แบบกระจัดกระจาย) v คือ several (อยู่รวมกันเป็นกระจุกเล็กๆ 2 – 4 ดอก) y คือ solitary (อยู่แบบดอกเดี่ยวๆ)
22	บริเวณที่ที่พบเจอเห็ดเจริญเติบโต (habitat)	g คือ grasses (ในกอหญ้า) l คือ leaves (บนใบไม้) m คือ meadows (บริเวณทุ่งหญ้ากว้างใหญ่) p คือ paths (บริเวณทางเดิน / บนพื้นดิน) u คือ urban (ในเมือง) w คือ waste (บริเวณกองขยะ) d คือ woods (บนท่อนไม้)
23	ประเภทของเห็ด (class)	e คือ edible (เห็ดรับประทานได้) p คือ poisonous (เห็ดพิษ)

1.2 การแบ่งชุดข้อมูล

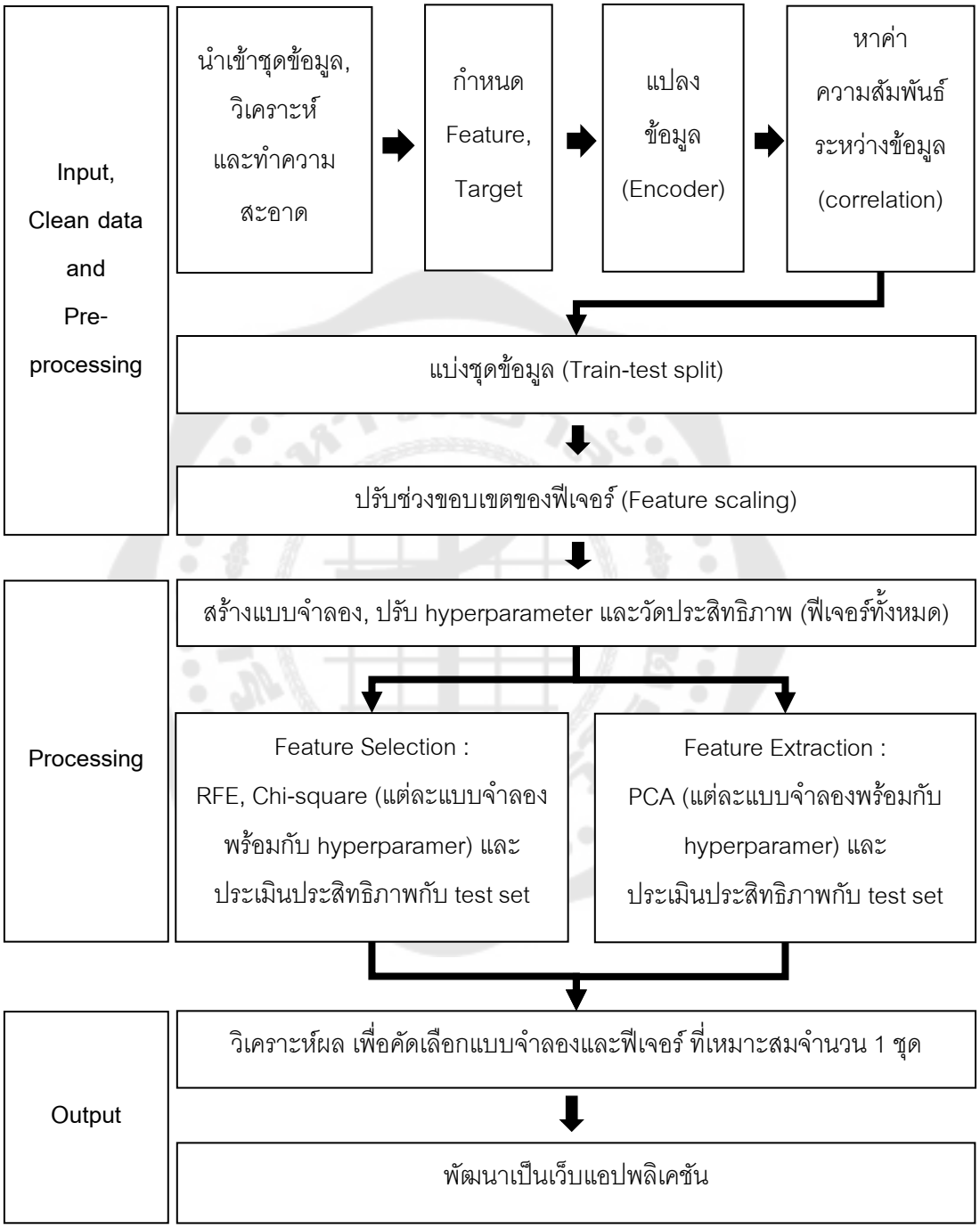
ชุดข้อมูลถูกแบ่งเป็น 2 ชุด คือ ชุดข้อมูลสำหรับสร้างแบบจำลอง (Training Set) และชุดข้อมูลสำหรับทดสอบแบบจำลอง (Test Set) ในอัตราส่วนต่างๆ จำนวน 3 อัตราส่วน (ของข้อมูลทั้งหมดจำนวน 8,124 รายการ

1.2.1 ชุดข้อมูลสำหรับสร้างแบบจำลอง ต่อ ชุดข้อมูลสำหรับทดสอบแบบจำลอง เท่ากับ 50 : 50 ซึ่งอัตราส่วนของข้อมูลตามจำนวนของเห็ดเป็น 4,062 : 4,062 (Training size = 50%)

1.2.2 ชุดข้อมูลสำหรับสร้างแบบจำลอง ต่อ ชุดข้อมูลสำหรับทดสอบแบบจำลอง เท่ากับ 60 : 40 ซึ่งอัตราส่วนของข้อมูลตามจำนวนของเห็ดเป็น 4,874 : 3,250 (Training size = 60%)

1.2.3 ชุดข้อมูลสำหรับสร้างแบบจำลอง ต่อ ชุดข้อมูลสำหรับทดสอบแบบจำลอง เท่ากับ 70 : 30 ซึ่งอัตราส่วนของข้อมูลตามจำนวนของเห็ดเป็น 5,686 : 2,438 (Training size = 70%)

2. การออกแบบวิธีการดำเนินงานวิจัย



ภาพประกอบ 17 ภาพรวมการดำเนินการวิจัย

จากภาพประกอบ 17 ซึ่งแสดงภาพรวมของขั้นตอนการดำเนินการงานวิจัยทั้งหมด โดยสามารถอธิบายรายละเอียดในแต่ละขั้นตอน ในหัวข้อถัดไป

3. การประมวลผลและการวิเคราะห์ข้อมูล

3.1 การนำเข้าชุดข้อมูลและสำรวจข้อมูล

ขั้นตอนการดำเนินงานเริ่มต้นจากการนำเข้าชุดข้อมูลสาธารณะของเห็ดครีบจำนวน 23 สายพันธุ์ในตระกูล Agaricus และ Lepiota มาแสดงรายละเอียด โดยใช้คำสั่ง df = pd.read_csv('file.csv') จากนั้นแสดงรายละเอียดข้อมูล 5 แถวแรก เป็นตาราง (dataframe) โดยใช้คำสั่ง df.head() พร้อมกับแสดงจำนวนแถว, คอลัมน์ทั้งหมด และชนิดของข้อมูลในแต่ละคอลัมน์ โดยใช้คำสั่ง df.info() ดังภาพประกอบ 18 และ 19 ตามลำดับ

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...
0	p	x	s	n	t	p	f	c	n	k	...
1	e	x	s	y	t	a	f	c	b	k	...
2	e	b	s	w	t	l	f	c	b	n	...
3	p	x	y	w	t	p	f	c	n	n	...
4	e	x	s	g	f	n	f	w	b	k	...

stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	population	habitat
s	w	w	p	w	o	p	k	s	u
s	w	w	p	w	o	p	n	n	g
s	w	w	p	w	o	p	n	n	m
s	w	w	p	w	o	p	k	s	u
s	w	w	p	w	o	e	n	a	g

ภาพประกอบ 18 ตัวอย่างข้อมูลในตารางของชุดข้อมูลดั้งเดิมจำนวน 5 แถวแรก ด้วยคำสั่ง df.head()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8124 entries, 0 to 8123
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   class                     8124 non-null   object
 1   cap-shape                 8124 non-null   object
 2   cap-surface               8124 non-null   object
 3   cap-color                 8124 non-null   object
 4   bruises                   8124 non-null   object
 5   odor                      8124 non-null   object
 6   gill-attachment           8124 non-null   object
 7   gill-spacing              8124 non-null   object
 8   gill-size                 8124 non-null   object
 9   gill-color                8124 non-null   object
 10  stalk-shape               8124 non-null   object
 11  stalk-root                8124 non-null   object
 12  stalk-surface-above-ring  8124 non-null   object
 13  stalk-surface-below-ring  8124 non-null   object
 14  stalk-color-above-ring    8124 non-null   object
 15  stalk-color-below-ring    8124 non-null   object
 16  veil-type                 8124 non-null   object
 17  veil-color                8124 non-null   object
 18  ring-number               8124 non-null   object
 19  ring-type                 8124 non-null   object
 20  spore-print-color         8124 non-null   object
 21  population                8124 non-null   object
 22  habitat                   8124 non-null   object
dtypes: object(23)
memory usage: 1.4+ MB
```

ภาพประกอบ 19 รายละเอียดของชุดข้อมูลดั้งเดิม ได้แก่ จำนวนแถว, คอลัมน์ทั้งหมด และชนิดของข้อมูล ด้วยคำสั่ง df.info()

3.2 การทำความสะอาดและวิเคราะห์ข้อมูล

จากนั้นจึงปรับแต่งชุดข้อมูลให้เหมาะสมเพื่อให้สามารถอ่านและทำความเข้าใจได้ง่ายขึ้น ดังนี้

3.2.1 เปลี่ยนชื่อคอลัมน์ต่างๆให้เหมาะสม โดยแทนที่ “-” เป็น “_” เพื่อป้องกันการเข้าใจว่าเป็นเครื่องหมายลบ

3.2.2 เปลี่ยนรูปแบบข้อความของข้อมูลในคอลัมน์ ring_number ให้เป็นตัวเลขแทนเนื่องจากระบุจำนวนของวงแหวนที่พบ ส่วนคอลัมน์อื่นๆที่เหลือ ทำการเปลี่ยนรูปแบบข้อความโดยแทนค่าจากตัวย่อให้เป็นคำเต็ม เช่น ในคอลัมน์ “habitat” ทำการเปลี่ยนจาก ‘g’ เป็น ‘grasses’ เป็นต้น

3.2.3 เปลี่ยนชื่อคอลัมน์ของ class ให้กลายเป็น class_ep เพื่อหลีกเลี่ยงการใช้คำสงวน (reserve word) ของ python

3.2.4 เปลี่ยนชนิดของข้อมูลในคอลัมน์ class_ep จาก object เป็น category และข้อมูลในคอลัมน์ ring_number ทำการเปลี่ยนจาก object เป็นตัวเลขจำนวนเต็ม (integer)

โดยชุดข้อมูลที่ผ่านการปรับแต่งให้เหมาะสมแล้ว สามารถแสดงได้ โดยใช้คำสั่ง df.head() และ df.info() อีกครั้ง ตามภาพประกอบ 20 และ 21 ตามลำดับ

	class_ep	cap_shape	cap_surface	cap_color	bruises	odor	gill_attachment	gill_spacing
0	poisonous	convex	smooth	brown	bruises	pungent	free	close
1	edible	convex	smooth	yellow	bruises	almond	free	close
2	edible	bell	smooth	white	bruises	anise	free	close
3	poisonous	convex	scaly	white	bruises	pungent	free	close
4	edible	convex	smooth	gray	no	none	free	crowded

gill_size	gill_color	...	stalk_surface_below_ring	stalk_color_above_ring	stalk_color_below_ring
narrow	black	...	smooth	white	white
broad	black	...	smooth	white	white
broad	brown	...	smooth	white	white
narrow	brown	...	smooth	white	white
broad	black	...	smooth	white	white

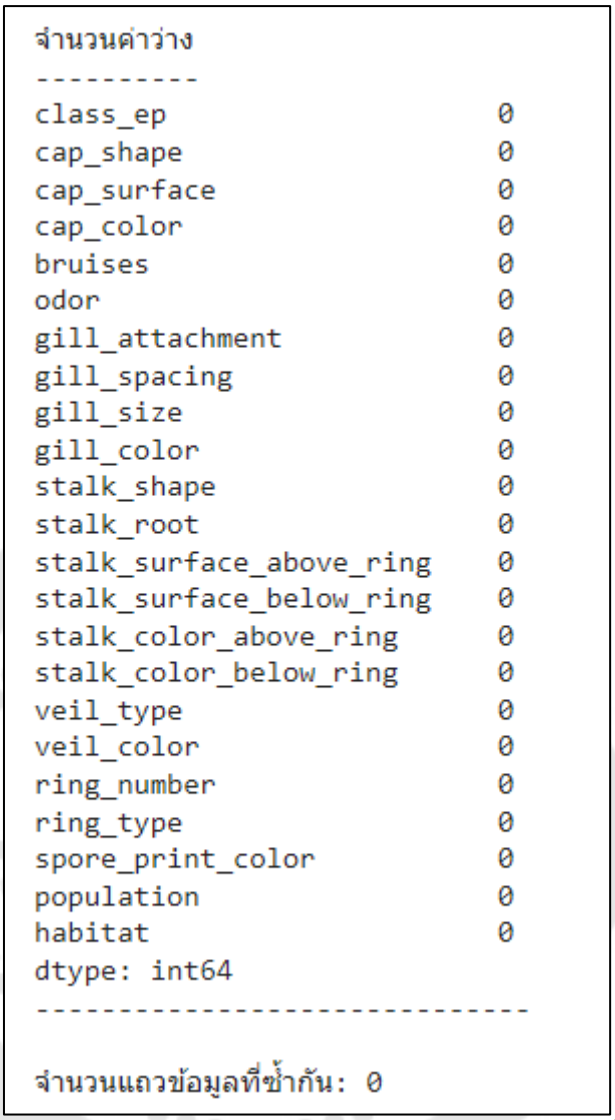
veil_type	veil_color	ring_number	ring_type	spore_print_color	population	habitat
partial	white	1	pendant	black	scattered	urban
partial	white	1	pendant	brown	numerous	grasses
partial	white	1	pendant	brown	numerous	meadows
partial	white	1	pendant	black	scattered	urban
partial	white	1	evanescent	brown	abundant	grasses

ภาพประกอบ 20 ตัวอย่างข้อมูลในตารางของชุดข้อมูลที่ผ่านการปรับแต่งให้เหมาะสม ด้วยคำสั่ง df.head()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8124 entries, 0 to 8123
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   class_ep                  8124 non-null   category
 1   cap_shape                 8124 non-null   object
 2   cap_surface               8124 non-null   object
 3   cap_color                 8124 non-null   object
 4   bruises                   8124 non-null   object
 5   odor                      8124 non-null   object
 6   gill_attachment           8124 non-null   object
 7   gill_spacing              8124 non-null   object
 8   gill_size                 8124 non-null   object
 9   gill_color                8124 non-null   object
 10  stalk_shape               8124 non-null   object
 11  stalk_root                8124 non-null   object
 12  stalk_surface_above_ring  8124 non-null   object
 13  stalk_surface_below_ring  8124 non-null   object
 14  stalk_color_above_ring    8124 non-null   object
 15  stalk_color_below_ring    8124 non-null   object
 16  veil_type                 8124 non-null   object
 17  veil_color                8124 non-null   object
 18  ring_number               8124 non-null   int64
 19  ring_type                 8124 non-null   object
 20  spore_print_color         8124 non-null   object
 21  population                8124 non-null   object
 22  habitat                   8124 non-null   object
dtypes: category(1), int64(1), object(21)
memory usage: 1.4+ MB
```

ภาพประกอบ 21 รายละเอียดของชุดข้อมูลที่ผ่านการปรับแต่งให้เหมาะสม ด้วยคำสั่ง df.info()

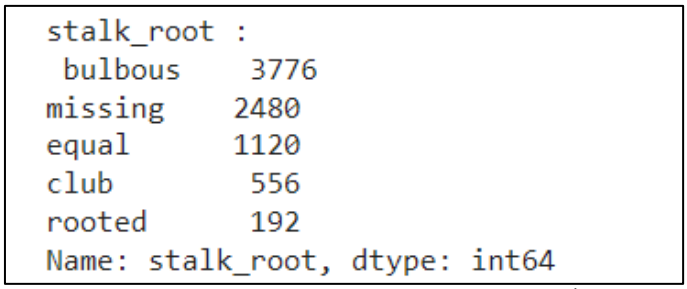
3.2.5 ตรวจสอบค่าว่าง และแถวข้อมูลที่ซ้ำกันว่ามีหรือไม่ ด้วยคำสั่ง df.isnull().sum() และ df.duplicated().sum() โดยสามารถแสดงผลลัพธ์ได้ตามภาพประกอบ 22 และเมื่อพิจารณาข้อมูลในแต่ละตัวแปรพบว่าข้อมูลตัวแปรในคอลัมน์ “stalk_root” (ลักษณะรากที่อยู่ปลายก้านดอก) นั้นมีการเติมค่าว่างด้วยคำว่า “missing” อยู่ ตามภาพประกอบ 23



```
จำนวนค่าว่าง
----------
class_ep                  0
cap_shape                 0
cap_surface               0
cap_color                 0
bruises                   0
odor                      0
gill_attachment           0
gill_spacing              0
gill_size                 0
gill_color                0
stalk_shape               0
stalk_root                0
stalk_surface_above_ring  0
stalk_surface_below_ring  0
stalk_color_above_ring    0
stalk_color_below_ring    0
veil_type                 0
veil_color                0
ring_number               0
ring_type                 0
spore_print_color         0
population                0
habitat                   0
dtype: int64
------------------------------

จำนวนแถวข้อมูลที่ซ้ำกัน: 0
```

ภาพประกอบ 22 ตรวจสอบจำนวนค่าว่าง และแถวข้อมูลที่ซ้ำกัน

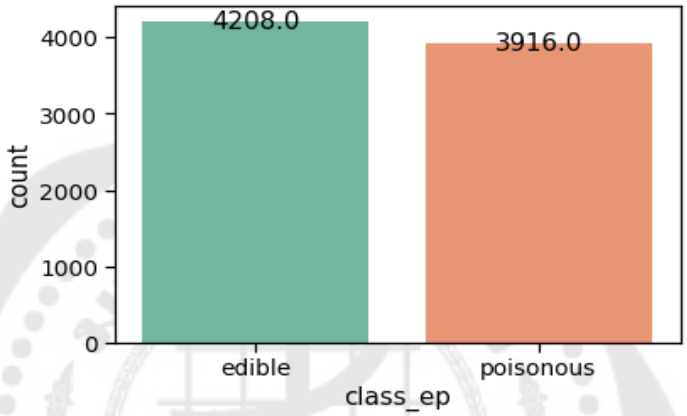


```
stalk_root :
 bulbous    3776
missing     2480
equal       1120
club         556
rooted       192
Name: stalk_root, dtype: int64
```

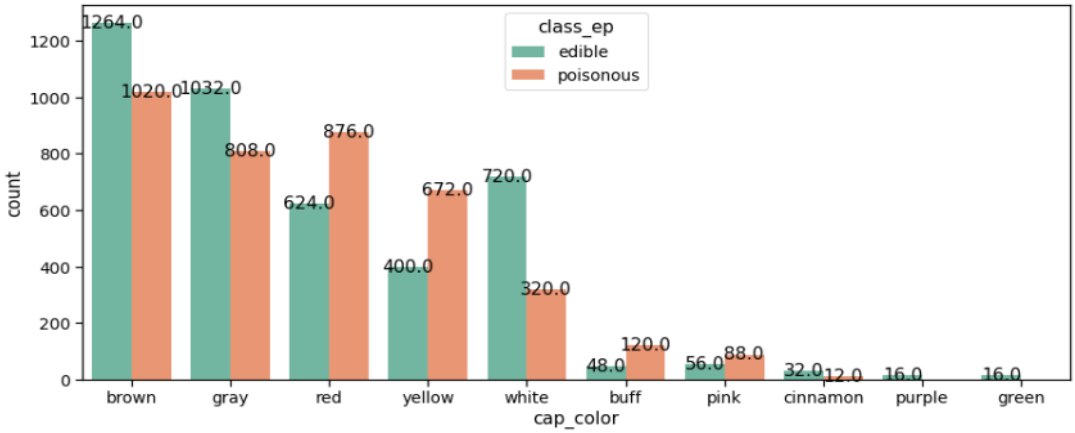
ภาพประกอบ 23 ข้อมูลตัวแปรในคอลัมน์ stalk_root (ลักษณะรากที่อยู่ปลายก้านดอก) ที่มีการเติมค่าว่างด้วยคำว่า missing

3.2.6 การวิเคราะห์เชิงสำรวจ (Exploratory Data Analysis)

ต่อไปจึงทำการวิเคราะห์ข้อมูล ด้วยการหาจำนวนภายในคอลัมน์ต่างๆ ว่ามีค่าตัวแปรแต่ละตัวแปร จำนวนเท่าไร และแบ่งเป็นเห็ดพิษกับเห็ดที่รับประทานได้จำนวนเท่าไร โดยแสดงเป็นกราฟแท่งระหว่างชนิดของตัวแปรกับจำนวนที่นับได้ ดังภาพประกอบ 24 ซึ่งแสดงให้เห็นว่าในชุดข้อมูลนี้มีจำนวนเห็ดที่รับประทานได้ เท่ากับ 4,208 ข้อมูล และจำนวนเห็ดพิษเท่ากับ 3,916 ข้อมูล ซึ่งเป็นสัดส่วนที่เท่าๆกัน

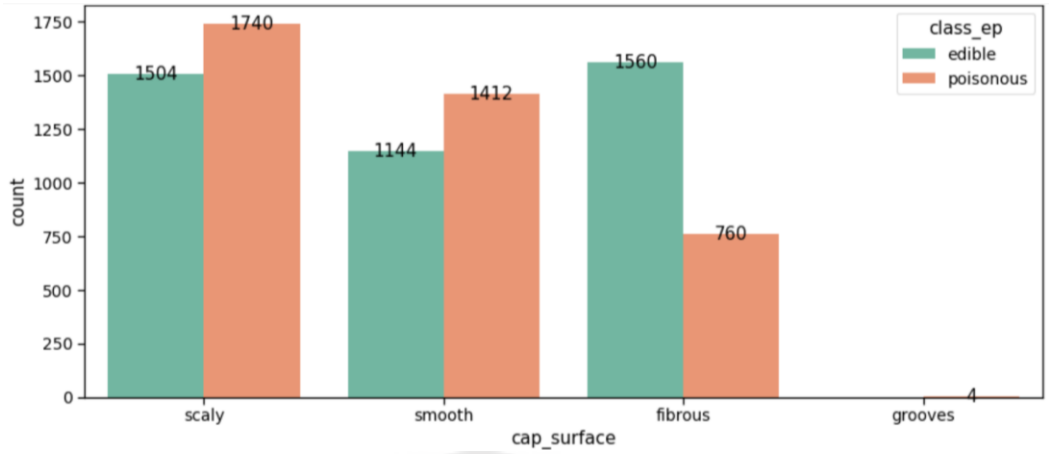


ภาพประกอบ 24 กราฟแท่งแสดงจำนวนข้อมูลเห็ดในแต่ละประเภท



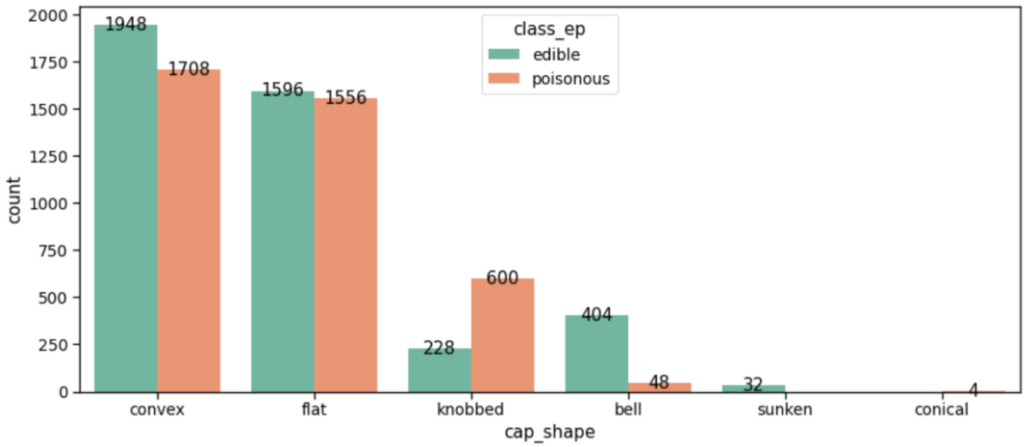
ภาพประกอบ 25 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 25 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีสีของหมวกเห็ดเป็นสีน้ำตาลเข้ม, สีเทา, สีขาว พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่เป็นสีแดง, สีเหลือง พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และเห็ดที่มีหมวกเห็ดสีอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



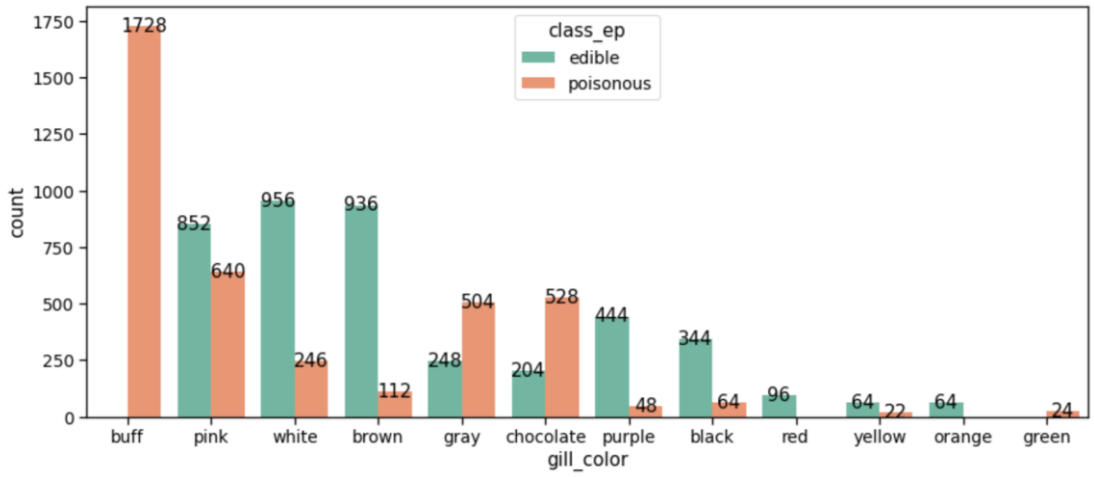
ภาพประกอบ 26 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะพื้นผิวของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 26 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีลักษณะพื้นผิวบนหมวกเห็ดเป็นขน พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนเห็ดที่มีลักษณะพื้นผิวบนหมวกเห็ดเป็นเกล็ด และเรียบ พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และเห็ดที่มีลักษณะพื้นผิวบนหมวกเห็ดเป็นร่องพบว่ามีจำนวนค่อนข้างน้อย



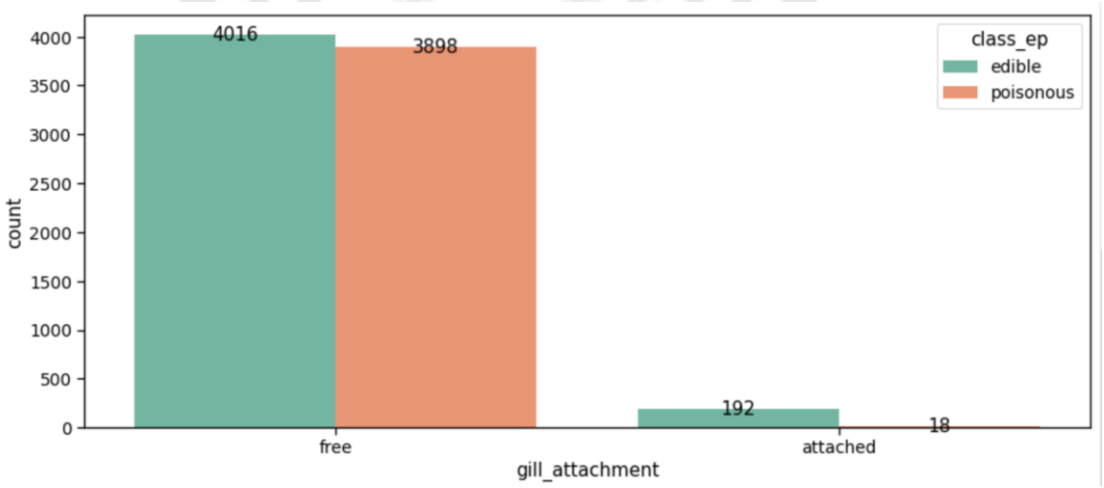
ภาพประกอบ 27 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปทรงของหมวกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 27 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีรูปทรงของหมวกเห็ดเป็นนูนโค้ง และทรงระฆัง พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนเห็ดที่มีรูปทรงของหมวกเห็ดเป็นร่มที่มีจุกอยู่ตรงกลาง พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนเห็ดที่มีรูปทรงของหมวกเห็ดแบนราบพบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่เท่ากันกับเห็ดพิษ และเห็ดที่มีลักษณะพื้นผิวบนหมวกเห็ดแบบอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



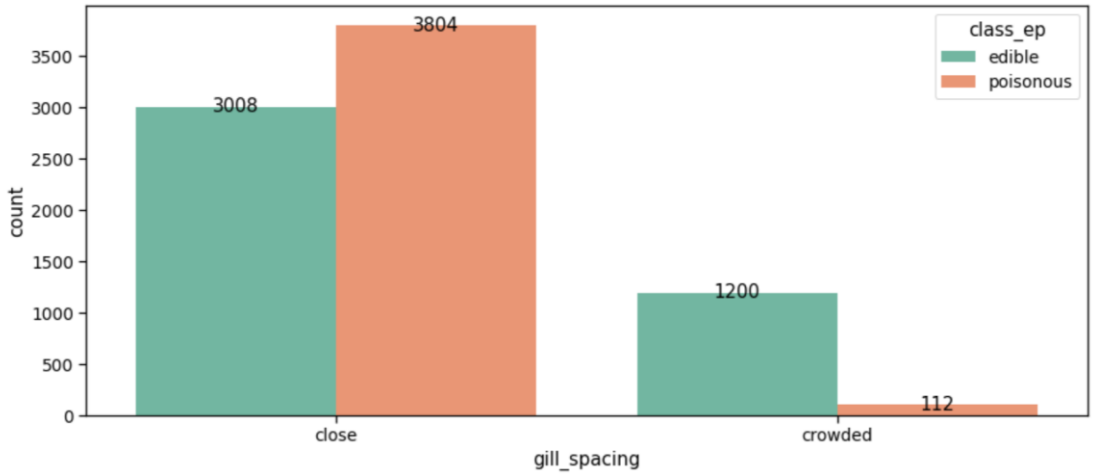
ภาพประกอบ 28 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของครีบเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 28 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีสีของครีบเห็ดเป็นสีชมพู, สีขาว, สีน้ำตาลเข้ม, สีม่วง, สีดำ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนเห็ดที่มีของครีบเห็ดเป็นสีน้ำตาลอมเหลืองพบว่าเป็นเห็ดพิษ ส่วนเห็ดที่มีของครีบเห็ดเป็นสีเทา และสีชอคโกแลต พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และเห็ดที่มีสีของครีบเห็ดเป็นสีอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



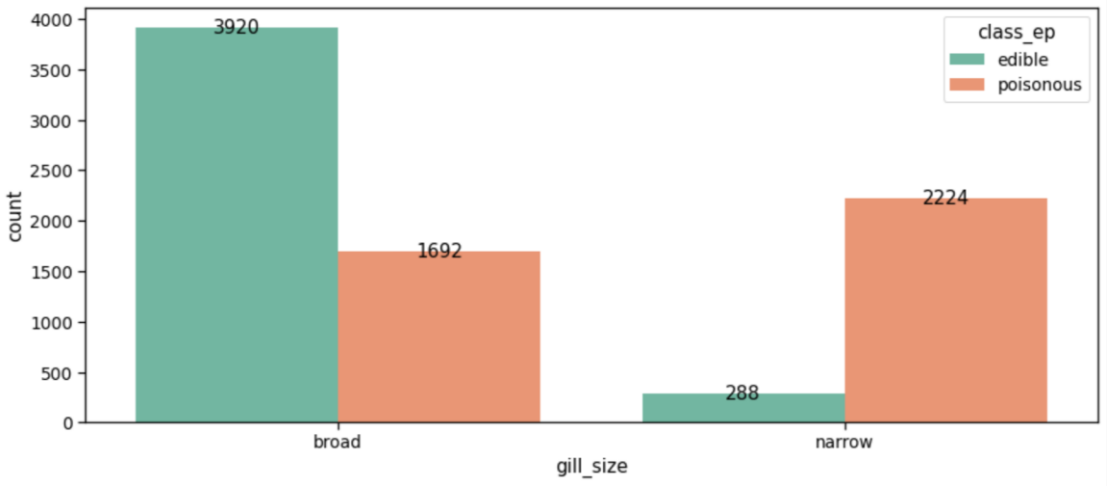
ภาพประกอบ 29 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามความแนบชิดของครีบเห็ดกับก้านดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 29 แสดงให้เห็นว่าเห็ดในชุดข้อมูลนี้ที่ครีบเห็ดกับก้านดอกไม่แนบติดกัน เป็นเห็ดรับประทานได้ในสัดส่วนที่เท่ากันกับเห็ดพิษ ส่วนเห็ดที่ครีบเห็ดกับก้านดอกแนบชิดกันในระดับเสมอกันตลอด พบว่ามีจำนวนค่อนข้างน้อย



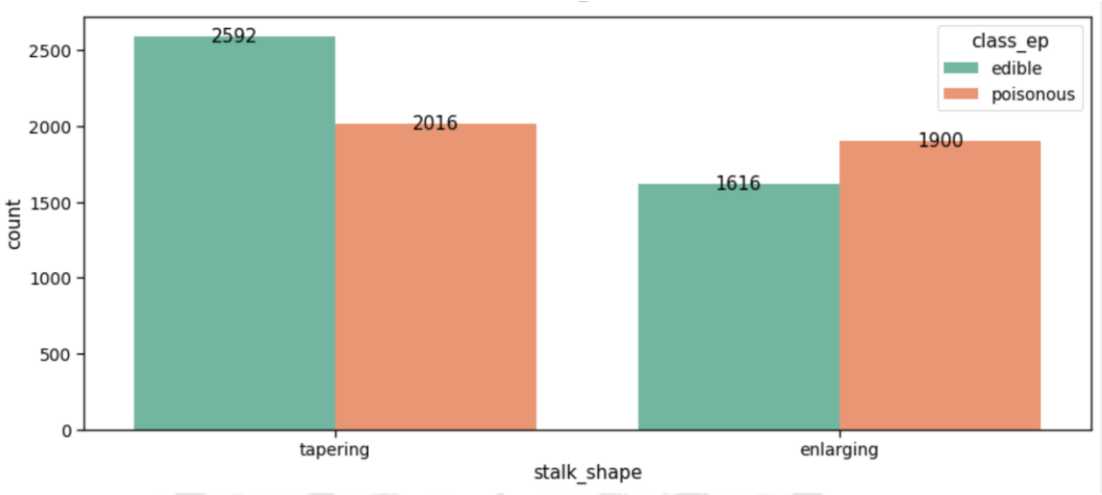
ภาพประกอบ 30 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามระยะห่างของครีบเห็ดแต่ละครีบ โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 30 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่ครีบเห็ดแต่ละครีบนั้นเรียงติดกัน พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนที่เรียงติดกันแบบหนาแน่นโดยไม่เห็นว่าเป็นเส้นๆ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ



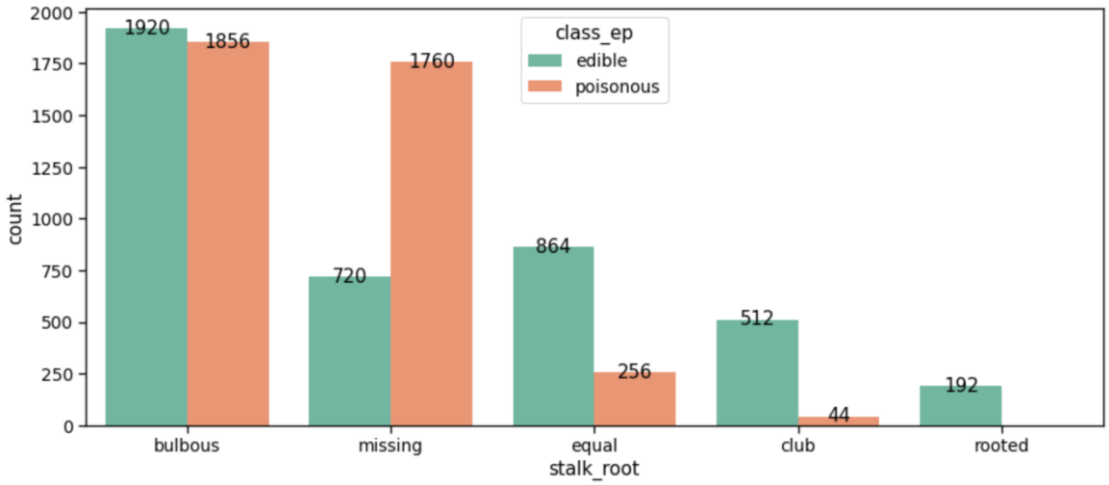
ภาพประกอบ 31 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามขนาดของครีบเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 31 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่ครีบเห็ดเป็นเส้นหนา พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่เป็นเส้นบาง พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้



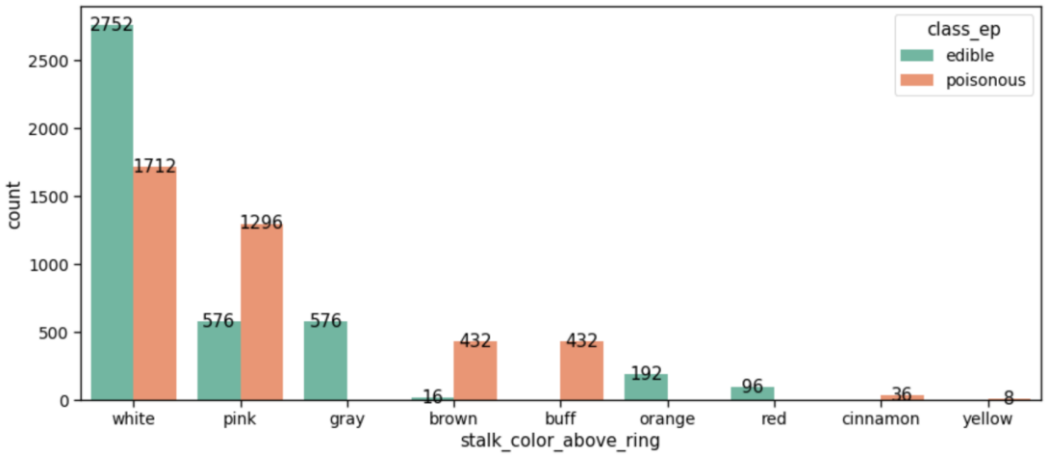
ภาพประกอบ 32 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปทรงของก้านดอก โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 32 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่ก้านดอกเห็ดมีลักษณะเรียวเล็กลงไปหาโคน พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนเห็ดที่บริเวณก้านดอกเห็ดด้านล่างมีขนาดใหญ่กว่าส่วนบนของก้านดอกเห็ด พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้เล็กน้อย



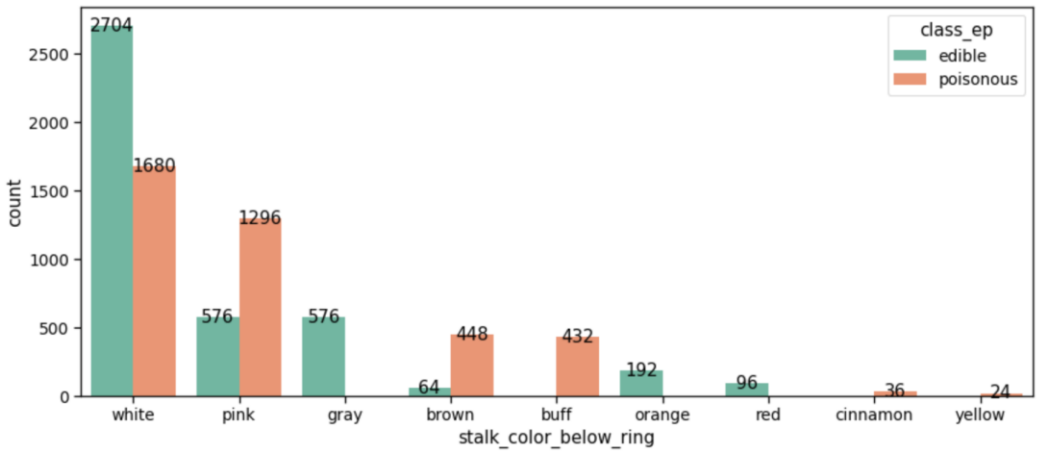
ภาพประกอบ 33 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะรากที่อยู่ปลายก้านดอก โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 33 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่บริเวณโคนก้านดอกเห็ดมีลักษณะป่องเป็นกระเปาะ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่เท่ากันกับเห็ดพิษ ส่วนเห็ดที่ตลอดก้านดอกเห็ดเรียบเสมอกันตลอด หรือบริเวณโคนก้านดอกนูนออกเล็กน้อย หรือปลายโคนก้านเป็นรากเรียวยาวนั้น พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ แต่เห็ดที่ไม่พบข้อมูลของลักษณะรากที่อยู่ปลายก้านดอกนั้น (missing) มีจำนวนค่อนข้างมาก



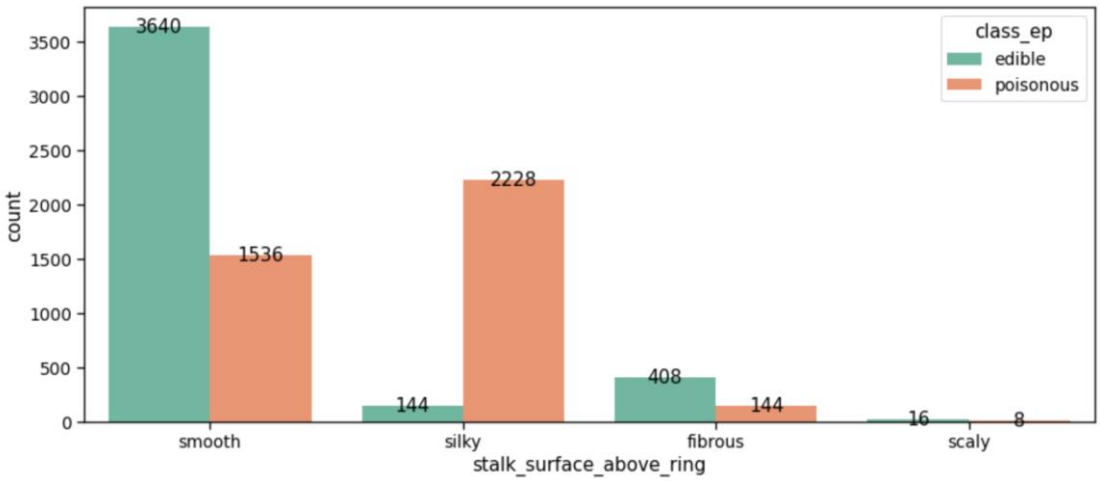
ภาพประกอบ 34 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 34 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีสีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวนเป็นสีขาว พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนสีเทา พบว่าเป็นเห็ดรับประทานได้ ส่วนสีชมพู, สีน้ำตาลเข้ม พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนสีน้ำตาลอมเหลืองพบว่าเป็นเห็ดพิษ และสีอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



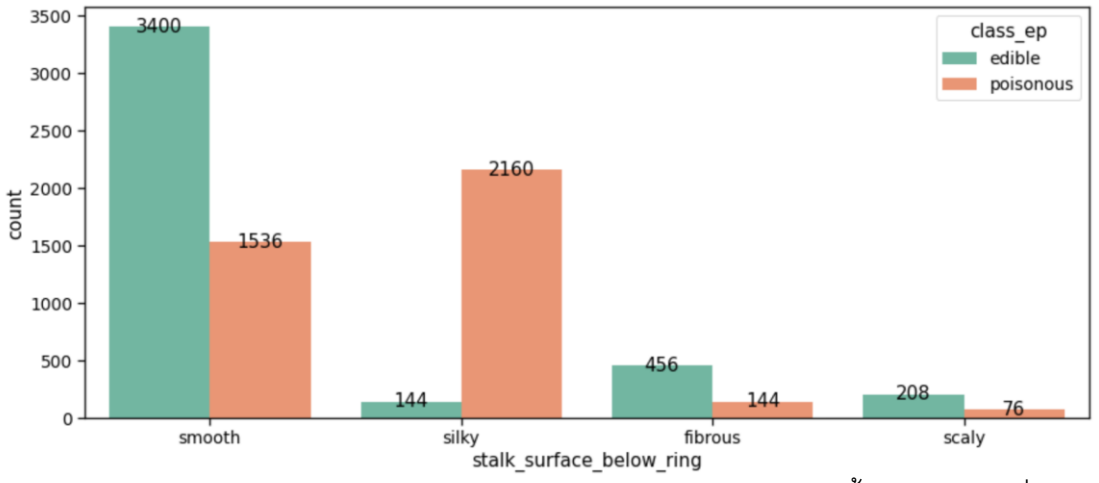
ภาพประกอบ 35 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 35 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีสีพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวนเป็นสีขาว พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนสีเทา พบว่าเป็นเห็ดรับประทานได้ ส่วนสีชมพู, สีน้ำตาลเข้ม พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนสีน้ำตาลอมเหลืองพบว่าเป็นเห็ดพิษ และสีอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



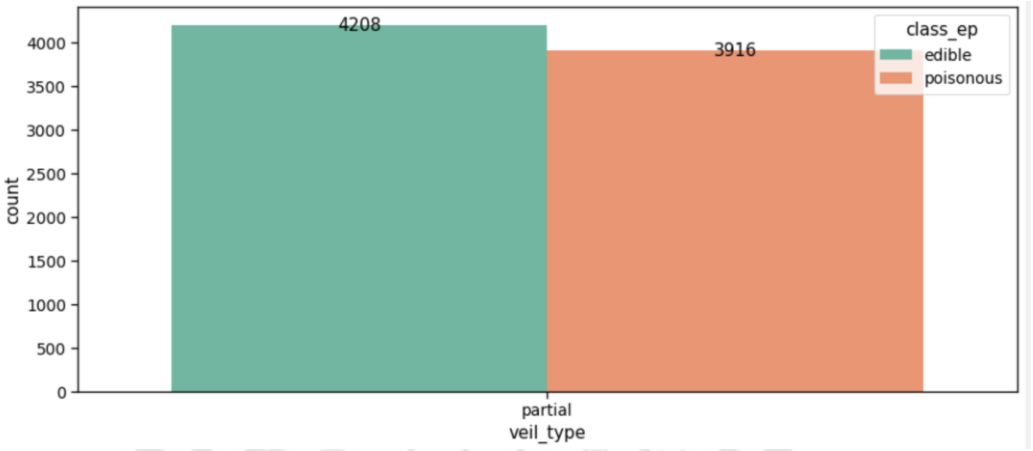
ภาพประกอบ 36 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะของพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 36 แสดงให้เห็นว่า พบว่าเห็ดในชุดข้อมูลนี้ที่พื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวนมีลักษณะเรียบๆ หรือเป็นเส้นใย/เยื่อ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่มีลักษณะเป็นขนนิ่มเหมือนสำลี พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และที่เป็นเกล็ดพบว่ามีจำนวนค่อนข้างน้อย



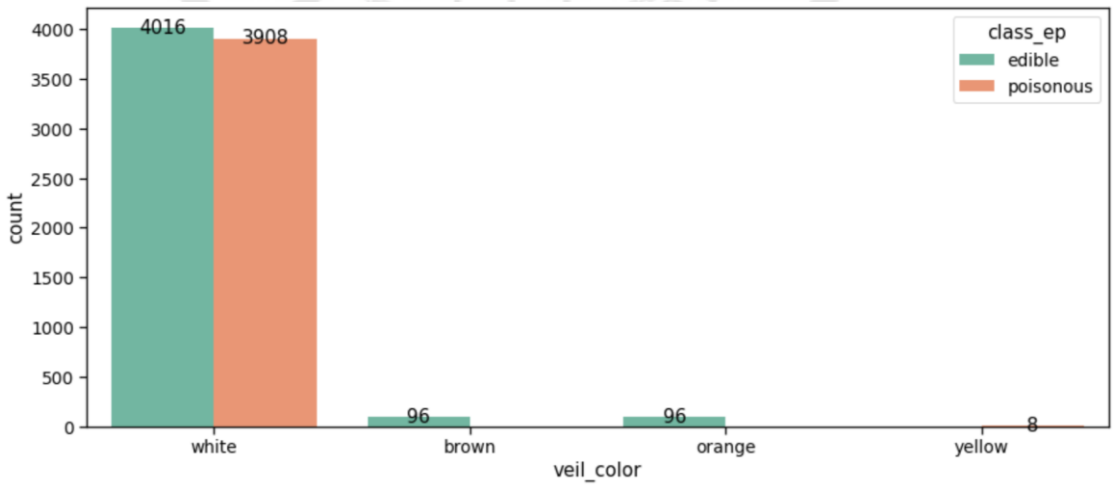
ภาพประกอบ 37 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะของพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 37 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่พื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวนมีลักษณะเรียบๆ หรือเป็นเส้นใย/เยื่อ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่มีลักษณะเป็นขนนิ่มเหมือนสำลี พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และที่เป็นเกล็ดพบว่ามีจำนวนค่อนข้างน้อย



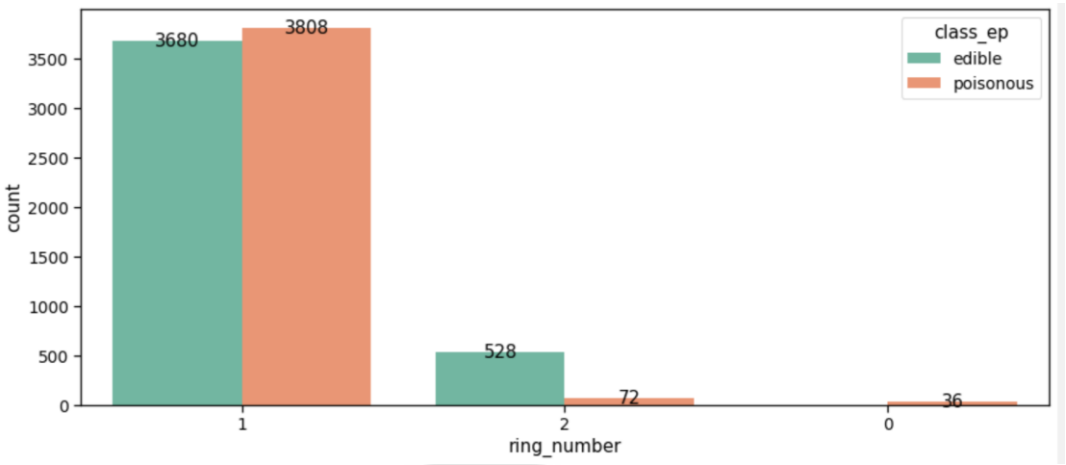
ภาพประกอบ 38 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามประเภทของเยื่อที่หุ้มดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 38 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีเยื่อที่หุ้มดอกเห็ดเพียงประเภทเดียวเท่านั้น คือประเภทที่พบวงแหวนบริเวณก้านดอกเห็ด



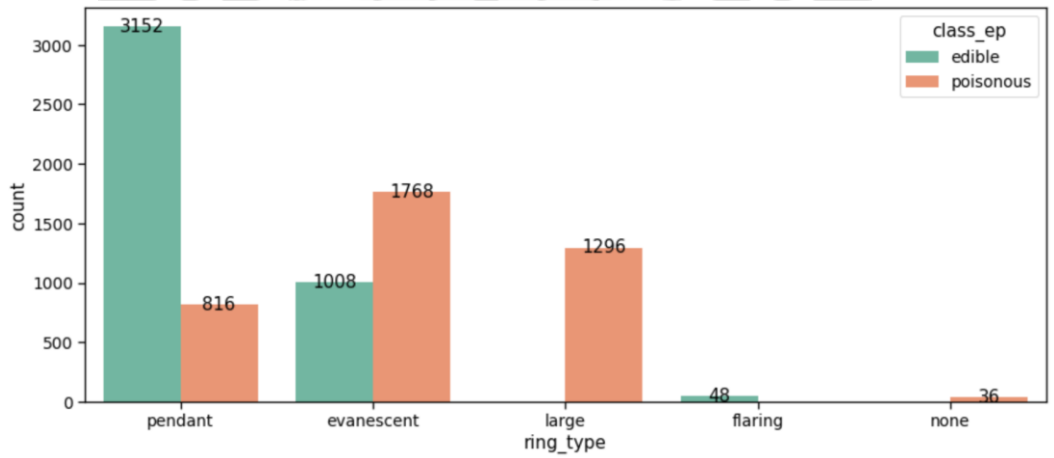
ภาพประกอบ 39 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของเยื่อที่หุ้มดอกเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 39 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่สีของเยื่อที่หุ้มดอกเห็ดเป็นสีขาว พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่เท่ากันกับเห็ดพิษ และสีอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



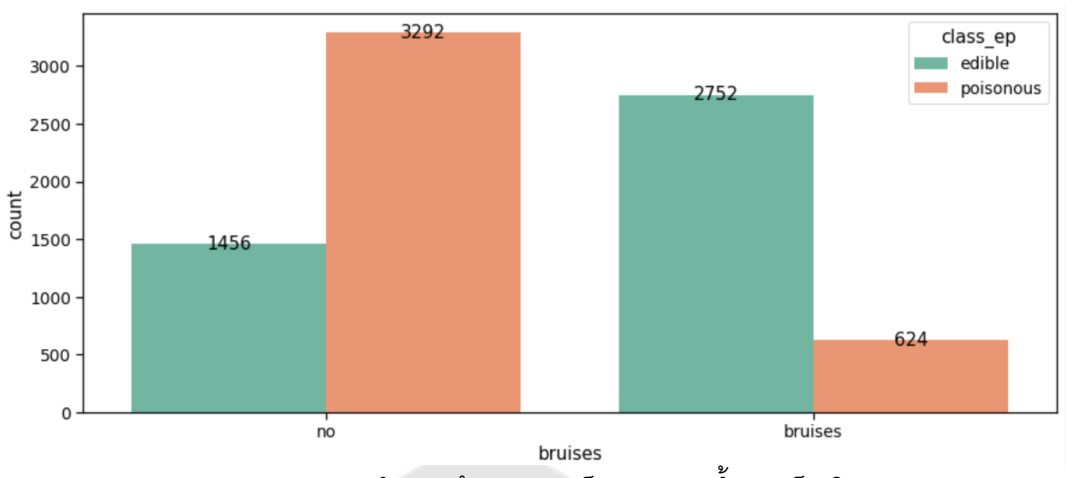
ภาพประกอบ 40 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามจำนวนวงแหวนของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 40 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีจำนวนวงแหวนของเห็ด 1 ชั้น พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่เท่ากันกับเห็ดพิษ ส่วนที่มีวงแหวนสองชั้นซ้อนกันพบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่ไม่พบวงแหวนพบว่ามีจำนวนค่อนข้างน้อย



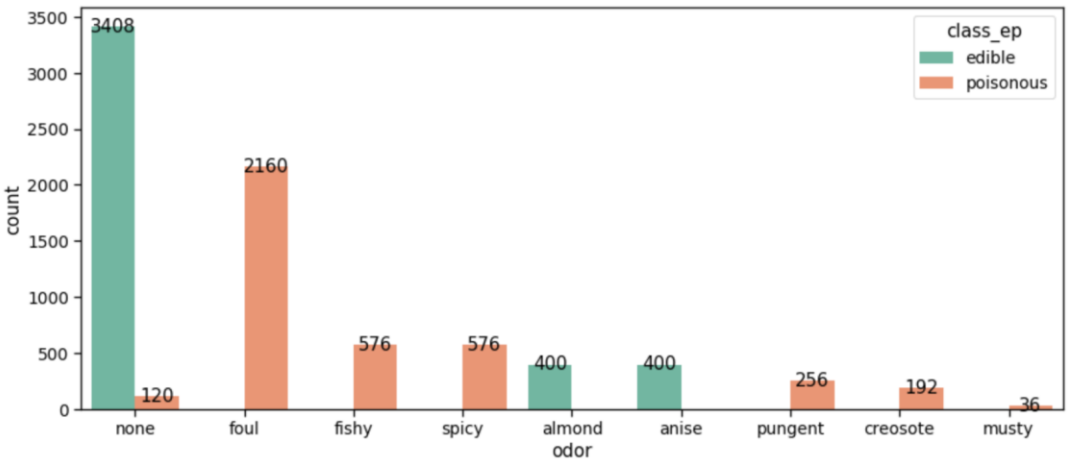
ภาพประกอบ 41 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรูปร่างของวงแหวน โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 41 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่รูปร่างของวงแหวน มีลักษณะเป็นแผ่นห้อยลง พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่มีลักษณะที่เป็นเศษหรือผงที่ติดอยู่บนก้านดอก หรือห้อยลงมาจากขอบหมวก พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนที่มีลักษณะใหญ่จนคลุมทั้งดอกเห็ดเอาไว้ พบว่าเป็นเห็ดพิษ และที่มีลักษณะอื่นๆ พบว่ามีจำนวนค่อนข้างน้อย



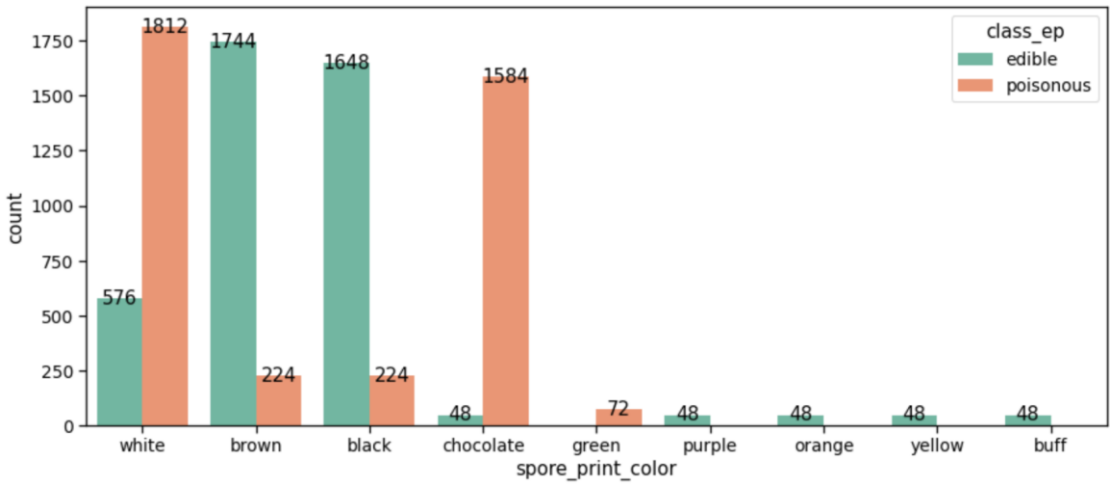
ภาพประกอบ 42 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามรอยช้ำบนเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 42 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่ไม่มีรอยช้ำ พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนที่พบรอยช้ำพบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ



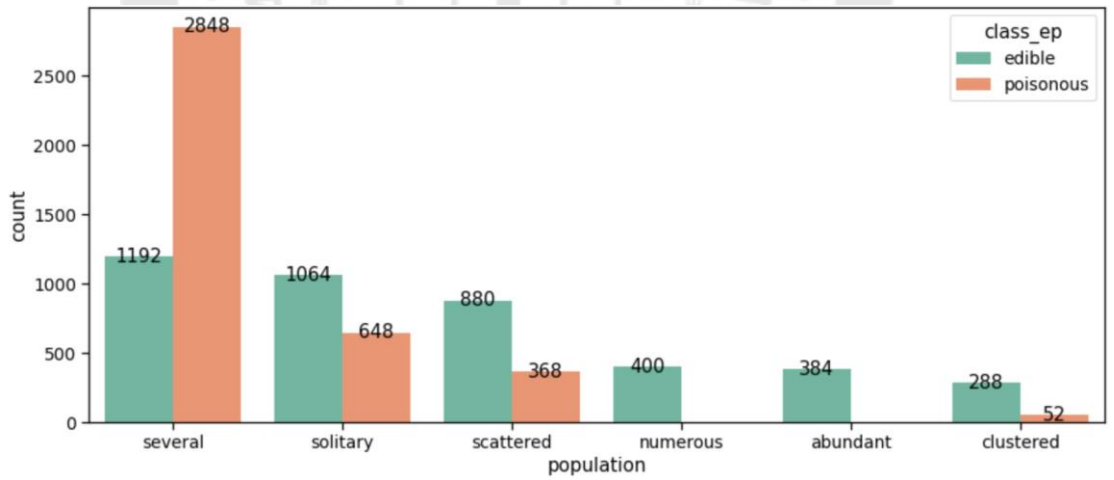
ภาพประกอบ 43 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามกลิ่นของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 43 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่มีกลิ่นแปลกๆ เช่น กลิ่นเหม็นไม่พึงประสงค์ทั่วไป, กลิ่นเหม็นคาว, กลิ่นเผ็ดคล้ายเครื่องเทศที่แสบจมูก, กลิ่นสารเคมีที่ฉุนแสบจมูก , กลิ่นน้ำมันถ่านหิน พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ ส่วนที่ไม่มีกลิ่นพบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ และที่มีกลิ่นอัลมอนด์, กลิ่นโป๊ยกั๊ก หรือกลิ่นหอมหวานอ่อนๆ พบว่าเป็นเห็ดรับประทานได้



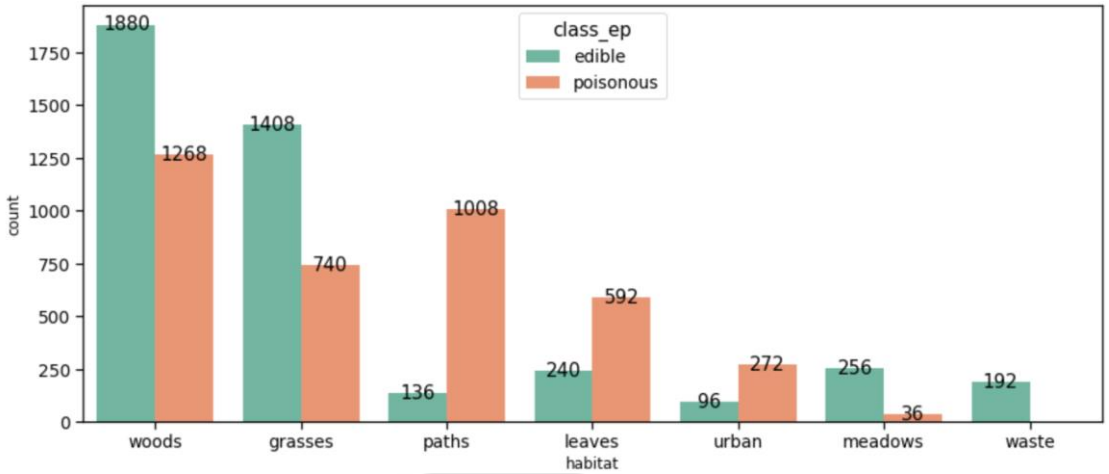
ภาพประกอบ 44 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามสีของรอยพิมพ์สปอร์ โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 44 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่รอยพิมพ์สปอร์มีสีน้ำตาลเข้ม, สีดำ พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่เป็นสีขาวหรือ สีชอคโกแลต พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และที่เป็นสีอื่นๆ มีจำนวนค่อนข้างน้อย



ภาพประกอบ 45 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามลักษณะการกระจายตัว/การเกาะกลุ่มกันของเห็ด โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 45 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่เจริญเติบโตอยู่เป็นดอกเดี่ยว หรืออยู่กระจัดกระจายกัน หรือเป็นช่อเห็ด โดยพบหลายช่อในบริเวณเดียวกัน พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่อยู่กันเป็นกลุ่มเล็กๆ 2-4 ดอก พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และที่อยู่รวมกันเป็นกลุ่ม จนถึงกลุ่มใหญ่ๆ พบว่าเป็นเห็ดที่รับประทานได้



ภาพประกอบ 46 กราฟแท่งแสดงจำนวนข้อมูลของเห็ดตามบริเวณที่พบเจอเห็ดเจริญเติบโต โดยแยกตามประเภทเห็ดพิษ และเห็ดรับประทานได้

จากภาพประกอบ 46 แสดงให้เห็นว่า เห็ดในชุดข้อมูลนี้ที่เจริญเติบโตอยู่บนท่อนไม้ หรือในกอหญ้า, บริเวณทุ่งหญ้า พบว่าเป็นเห็ดรับประทานได้ในสัดส่วนที่มากกว่าเห็ดพิษ ส่วนที่มีเจริญเติบโตอยู่บริเวณทางเดิน/บนพื้นดิน, บนใบไม้, ในเมือง พบว่าเป็นเห็ดพิษในสัดส่วนที่มากกว่าเห็ดรับประทานได้ และที่เจริญเติบโตอยู่บริเวณกองขยะพบว่าเป็นเห็ดรับประทานได้

จากข้อมูลที่กล่าวมาแล้วพบว่าคอลัมน์ที่ชื่อ “veil_type” (ประเภทของเยื่อที่หุ้มดอกเห็ดที่พบ) นั้นพบค่าตัวแปรแค่ชนิดเดียว คือ partial (เยื่อหุ้มดอกเห็ดแบบวงแหวน) และในคอลัมน์ชื่อ “stalk_root” (ลักษณะรากที่อยู่ปลายก้านดอก) พบว่ามีค่าว่างจำนวนมาก (30% ของข้อมูลทั้งหมด) ดังนั้นจึงลบคอลัมน์ 2 คอลัมน์นี้ไปก่อนนำไปใช้ฝึกสอนเทคนิคการเรียนรู้ของเครื่อง

3.3 การกำหนดตัวแปรต้น (feature) และตัวแปรตาม (Target)

กำหนดว่าคอลัมน์ใดเป็นฟีเจอร์ (X) และคอลัมน์ใดเป็นผลลัพธ์จากการทำนายของแบบจำลอง โดยในงานวิจัยนี้ได้กำหนดให้คอลัมน์ชื่อ “class_ep” เป็นผลลัพธ์จากการทำนาย ส่วนคอลัมน์ที่เหลือหลังจากทำความสะอาดชุดข้อมูลโดยลบคอลัมน์ที่ชื่อ “veil_type” และ “stalk_root” ไปแล้ว (เหลือ 20 คอลัมน์) ได้ถูกกำหนดให้เป็นฟีเจอร์ตามคำสั่งในภาพประกอบ 47

```
X_forOrdinal = df[['cap_shape', 'cap_surface', 'cap_color', 'bruises', 'odor',  'gill_attachment',
    'gill_spacing', 'gill_size', 'gill_color', 'stalk_shape', 'stalk_surface_above_ring',
     'stalk_surface_below_ring', 'stalk_color_above_ring', 'stalk_color_below_ring', 'veil_color',
     'ring_type', 'ring_number','spore_print_color', 'population', 'habitat']]
y = data_new2['class_ep']
```

ภาพประกอบ 47 คำสั่งสำหรับการกำหนดตัวแปรต้น (feature) และตัวแปรตาม (Target)

3.4 การแปลงข้อมูลให้เป็นตัวเลข และหาค่าความสัมพันธ์ระหว่างข้อมูล (correlation)

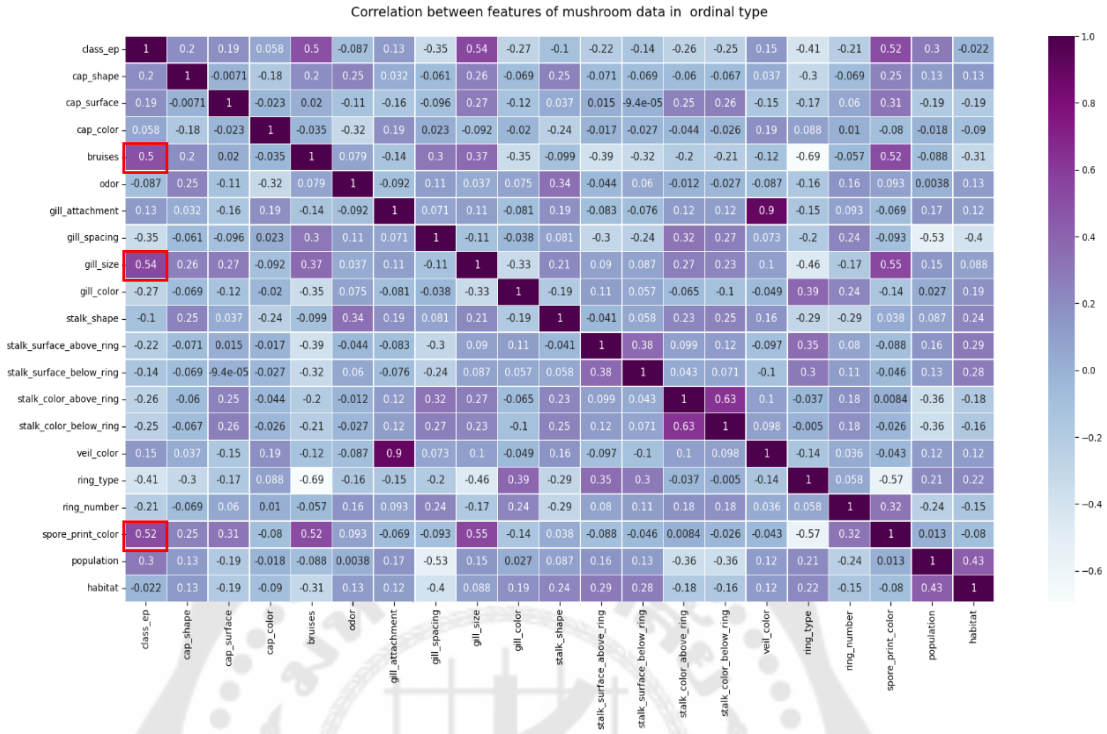
จากนั้นจึงนำฟีเจอร์ไปแปลงข้อมูลให้เป็นตัวเลขโดยใช้เครื่องมือที่ชื่อว่า Ordinal encoder ส่วน ประเภทของเห็ด (“class_ep”) ทำการแปลงข้อมูลให้เป็นตัวเลขโดยใช้เครื่องมือที่ชื่อว่า Label encoder ตามคำสั่งในภาพประกอบ 48 ซึ่งเห็ดพิษ (poisonous) ถูกแปลงเป็นค่า 1 ส่วนเห็ดที่รับประทานได้ (Edible) ถูกแปลงเป็นค่า 0

```
ordinal = OrdinalEncoder()
X_Ordinal = ordinal.fit_transform(X_forOrdinal)
le = LabelEncoder()
y = np.array(le.fit_transform(y))
```

ภาพประกอบ 48 ตัวอย่างคำสั่งการแปลงข้อมูลให้เป็นตัวเลข โดยใช้ Ordinal encoder และ Label Encoder

เมื่อข้อมูลถูกแปลงเป็นตัวเลขแล้วจึงสามารถหาค่าความสัมพันธ์ระหว่างข้อมูล (correlation) ได้ เพื่อตรวจสอบว่าฟีเจอร์ใดมีความสัมพันธ์กับประเภทของเห็ดมากที่สุด โดยสามารถแสดงค่าความสัมพันธ์ระหว่างข้อมูลได้ตามภาพประกอบ 49 ซึ่งสามารถอธิบายได้ดังนี้

ฟีเจอร์ที่ชื่อว่า “gill_size” (ขนาดของครีบเห็ด) มีความสัมพันธ์มากที่สุดกับประเภทของเห็ด โดยค่า correlation เท่ากับ 0.54 และฟีเจอร์ที่มีความสัมพันธ์กับประเภทของเห็ดรองลงมาคือ “spore_print_color” (สีของรอยพิมพ์สปอร์) และ “bruise” (รอยช้ำบนเห็ด) โดยมีค่า correlation เท่ากับ 0.52 และ 0.50 ตามลำดับ ซึ่งมีความสอดคล้องกับงานวิจัย 3 เรื่อง ได้แก่บทความของ V. Vanitha, M.N. Ahil และ N. Rajathi [51], บทความของ Eyad Sameh Alkronz, Khaled A. Moghayer, Mohamad Meimeh, Mohannad Gazzaz, Bassem S. Abu-Nasser และ Samy S. Abu-Naser [44] และบทความของ Shuhaida Ismail, Amy Rosshaida Zainal และ Aida Mustapha [48]



ภาพประกอบ 49 ค่าความสัมพันธ์ระหว่างข้อมูล (correlation)

3.4 การแบ่งชุดข้อมูลและปรับช่วงขอบเขตของฟีเจอร์ (Feature scaling)

จากนั้นจึงนำชุดข้อมูลไปแบ่งเป็น 2 ส่วน คือ ชุดข้อมูลสำหรับสร้างแบบจำลอง (Training Set) และชุดข้อมูลสำหรับทดสอบแบบจำลอง (Test Set) ด้วยเทคนิค Train-test split ในอัตราส่วนจำนวน 3 อัตราส่วน คือ 50 : 50, 60 : 40 และ 70 : 30 โดยในแต่ละอัตราส่วนนั้นให้มีเห็ดทั้ง 2 ประเภทในทั้ง 2 ส่วน เป็นจำนวนเท่าๆ กัน ตามตัวอย่างคำสั่งในภาพประกอบ 50

```
#แบ่งข้อมูลสำหรับ Train และ Test model โดยแบ่งชุดข้อมูลให้คือ test_size = 0.50
from sklearn.model_selection import train_test_split
X_Ord_train, X_Ord_test , y_train, y_test = train_test_split(X_Ordinal,y, stratify=y,
                                                    test_size = 0.50, random_state= 1)
```

ภาพประกอบ 50 ตัวอย่างคำสั่งการแบ่งชุดข้อมูลในอัตราส่วน 50 : 50

แล้วนำไปปรับช่วงขอบเขตของฟีเจอร์ โดยในงานวิจัยนี้ ใช้เทคนิคที่ชื่อว่า MinMaxscaler ซึ่งใช้คำสั่ง fit_transform กับ ฟีเจอร์ในชุดข้อมูลสำหรับสร้างแบบจำลองก่อน แล้วจึงใช้คำสั่ง transform กับ ฟีเจอร์ในชุดข้อมูลสำหรับทดสอบแบบจำลอง ตามตัวอย่างคำสั่งในภาพประกอบ 51

```
scaler = MinMaxScaler()
X_Ord_train_scale = scaler.fit_transform(X_Ord_train)
X_Ord_test_scale = scaler.transform(X_Ord_test)
```

ภาพประกอบ 51 ตัวอย่างคำสั่งการปรับช่วงขอบเขตของฟีเจอร์

4. การสร้างแบบจำลอง และประเมินประสิทธิภาพแบบจำลอง

4.1 การสร้างแบบจำลองด้วยฟีเจอร์ทั้งหมด

แบบจำลองที่ใช้ในงานวิจัยนี้มี 5 แบบจำลอง คือ Logistic regression, Support Vector Machine, Decision tree, Random Forest และ XGBoost โดยในการสร้างแบบจำลองทั้ง 5 แบบจำลอง ได้ปรับจูนหาค่า Hyperparameter ที่ทำให้แบบจำลองมีค่าความถูกต้องมากที่สุด ด้วยเทคนิคที่ชื่อว่า RandomizedSearchCV ซึ่งเป็นเทคนิคที่ใช้เวลาในการประมวลผลไม่นาน โดยตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแต่ละแบบจำลอง เป็นไปตามภาพประกอบ 52 – 56

```
LR = make_pipeline(LogisticRegression(max_iter = 10000))
param_rand = {"logisticregression__C":np.linspace(0.01,100),
                "logisticregression__penalty": ['l2','l1', 'elasticnet', 'none'],
                "logisticregression__solver": ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']}
log_reg = RandomizedSearchCV(LR, param_distributions=param_rand, cv=5, random_state=1)
log_reg.fit(X_Ord_train_scale, y_train)
```

ภาพประกอบ 52 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Logistic regression

```
SVM_pipe = make_pipeline(SVC())
param_dist = {'svc__kernel': ['linear', 'rbf', 'poly'],
              'svc__degree': np.linspace(1, 5, 4),
               'svc__C': np.linspace(0.01,100,20),
               'svc__gamma': np.logspace(-3, 2, 6) / X_Ord_train_scale.shape[1]}
svm = RandomizedSearchCV(SVM_pipe, param_distributions=param_dist, cv = 5, random_state=1)
svm.fit(X_Ord_train_scale, y_train)
```

ภาพประกอบ 53 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Support Vector Machine

```
dt = DecisionTreeClassifier()
dt_param = {'max_depth': [int(x) for x in np.linspace(0, 50)],
            'max_leaf_nodes': [int(x) for x in np.linspace(0, 50)],
            'min_samples_split':  [int(x) for x in np.linspace(0,50)],
            'max_features': ['sqrt', 'log2']}
dt_search = RandomizedSearchCV(dt, dt_param,  cv=5, random_state=0)
dt_search.fit(X_Ord_train_scale, y_train)
```

ภาพประกอบ 54 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Decision tree

```
rf_model = RandomForestClassifier(n_estimators= 300, n_jobs=-1)
rf_param = {'max_depth': [int(x) for x in np.linspace(10, 100, 20)],
            'max_leaf_nodes': [int(x) for x in np.linspace(10, 100, 20)],
            'min_samples_split':  [int(x) for x in np.linspace(10, 100, 20)],
            'max_features': ['sqrt', 'log2']}
rf_search = RandomizedSearchCV(rf_model, rf_param,  cv=5, random_state=1)
rf_search.fit(X_Ord_train_scale, y_train)
```

ภาพประกอบ 55 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง Random Forest

```
param = {"subsample":[0.5, 0.75, 1],
         "colsample_bytree":[0.5, 0.75, 1],
         "max_depth":[int(x) for x in np.linspace(3, 10, 5)],
         "min_child_weight":[1,5,10,15],
         "learning_rate":np.linspace(0.01, 0.3, 5)}
xg = xgb.XGBClassifier(n_estimators=300, n_jobs=-1)
model_xg= RandomizedSearchCV(xg,param,cv=5,random_state=1)
model_xg.fit(X_Ord_train_scale,y_train)
```

ภาพประกอบ 56 ตัวอย่างคำสั่งของการสร้างแบบจำลองและการปรับจูนหาค่า Hyperparameter ของแบบจำลอง XGBoost

เมื่อได้ค่า Hyperparameter ที่เหมาะสมของแต่ละแบบจำลองแล้ว จึงวัดประสิทธิภาพแบบจำลองเบื้องต้น ด้วยการสร้าง confusion matrix, วัดค่า accuracy, F1 score และ AUC (Area Under the Curve) score ก่อนนำไปใช้ในขั้นตอนต่อไป

4.2 การคัดเลือกฟีเจอร์ (Feature Selection)

ในงานวิจัยนี้ใช้เทคนิคการคัดเลือกฟีเจอร์จำนวน 2 เทคนิคต่อไปนี้ ร่วมกับทั้ง 5 แบบจำลอง ซึ่งได้นำค่า Hyperparameter ที่เหมาะสมของแต่ละแบบจำลองมาใช้ในขั้นตอนการคัดเลือกฟีเจอร์

4.2.1 Recursive feature elimination (RFE)

ในงานวิจัยนี้ได้ตรวจสอบค่า cross validation score (accuracy) (โดยกำหนด cross validation = 5 folds) เมื่อใช้จำนวนฟีเจอร์ (n_features_to_select) ตั้งแต่ 1 - 20 โดยตัวอย่างคำสั่งในการตรวจสอบค่า cross validation score (accuracy) ที่จำนวนฟีเจอร์ตั้งแต่ 1 - 20 ซึ่งเป็นไปตามภาพประกอบ 57 - 60 ซึ่งคำสั่งที่ใช้ในแต่ละแบบจำลองนั้นแตกต่างกันอย่างชัดเจนในส่วนที่เป็น estimator

โดยหลังจากตรวจสอบค่า cross validation score (accuracy) ในแต่ละจำนวน n_features_to_select แล้ว ได้นำไปสร้างเป็นกราฟ โดยใช้เครื่องมือที่ชื่อว่า “Microsoft Office Excel 365” หรือสามารถเขียนคำสั่งภาษา Python เพื่อแสดงเป็นกราฟได้ตามตัวอย่างที่แสดงในภาพประกอบ 61

หลังจากนั้นจึงเลือกจำนวนฟีเจอร์ที่เหมาะสมที่เทคนิค RFE ของแต่ละแบบจำลองประมวลผลได้ โดยกำหนดให้ใช้จำนวนฟีเจอร์ในช่วงที่ค่า cross validation score (accuracy) เริ่มมีความคงที่ หรือเริ่มมากกว่า 0.9000 เป็นต้นไป โดยจำนวนฟีเจอร์ที่ศึกษานั้นต้องไม่มากเกินไป เนื่องจากหากใช้จำนวนฟีเจอร์ที่มากเกินไปอาจส่งผลให้แบบจำลองที่ได้มีโอกาสเกิดภาวะ overfitting ได้ จากนั้นจึงตรวจสอบประสิทธิภาพ

ซึ่งเทคนิค RFE นี้มีข้อจำกัดกับแบบจำลองที่ไม่มีฟังก์ชั่นการหา Feature importance หรือค่า coefficient อย่าง Support Vector Machine ที่เมื่อปรับจูน hyperparameter แล้วพบว่า kernel ไม่เป็น “linear”

```
n_features_RFE = []
accuracy_LR_RFE = []
for n_feature in range(1,21):
    LR = LogisticRegression(max_iter = 10000, random_state = 1, solver ='newton-cg',
                            penalty = 'none', C = 22.456734693877554)
   LRrfe = RFE(estimator= LR, n_features_to_select=n_feature, step=1, verbose=0,
                importance_getter='auto')
   accu_score = cross_val_score(LRrfe, X_Ord_train_scale, y_train, cv = 5, scoring="accuracy")
   accLR_RFE = np.mean(accu_score)
   accuracy_LR_RFE.append(accLR_RFE)
   n_features_RFE.append(n_feature)
print(n_features_RFE)
print(accuracy_LR_RFE)
```

ภาพประกอบ 57 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Logistic regression

```
n_features_RFE = []
accuracy_DT_RFE = []
for n_feature in range(1,21):
    dt = DecisionTreeClassifier(min_samples_split = 9, max_leaf_nodes = 35,
                                max_features = 'log2', max_depth =21)
   DTrfe = RFE(estimator= dt, n_features_to_select=n_feature, step=1, verbose=0,
         importance_getter='auto')
   accu_score = cross_val_score(DTrfe, X_Ord_train_scale, y_train, cv = 5, scoring="accuracy")
   accDT_RFE = np.mean(accu_score)
   accuracy_DT_RFE.append(accDT_RFE)
   n_features_RFE.append(n_feature)
print(n_features_RFE)
print(accuracy_DT_RFE)
```

ภาพประกอบ 58 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Decision tree

```
n_features_RFE = []
accuracy_RF_RFE = []
for n_feature in range(1,21):
    rf_model = RandomForestClassifier(n_estimators= 300, n_jobs=-1, min_samples_split = 52,
                                      max_leaf_nodes = 43, max_features = 'log2', max_depth =85)
    RFrfe = RFE(estimator= rf_model,  n_features_to_select=n_feature, step=1, verbose=0,
                importance_getter='auto')
    accu_score = cross_val_score(RFrfe, X_Ord_train_scale, y_train, cv = 5, scoring="accuracy")
    accRF_RFE = np.mean(accu_score)
    accuracy_RF_RFE.append(accRF_RFE)
    n_features_RFE.append(n_feature)
print(n_features_RFE)
print(accuracy_RF_RFE)
```

ภาพประกอบ 59 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ Random Forest

```
n_features_RFE = []
accuracy_XG_RFE = []
for n_feature in range(1,21):
    xg = xgb.XGBClassifier(n_estimators=300, subsample = 1, min_child_weight= 10,
                           max_depth= 4, learning_rate= 0.3, colsample_bytree= 1)
    XGrfe = RFE(estimator= xg, n_features_to_select=n_feature, step=1, verbose=0,
                importance_getter='auto')
    accu_score = cross_val_score(XGrfe, X_Ord_train_scale, y_train,
                                 cv = 5, scoring="accuracy")
    accXG_RFE = np.mean(accu_score)
    accuracy_XG_RFE.append(accXG_RFE)
    n_features_RFE.append(n_feature)
print(n_features_RFE)
print(accuracy_XG_RFE)
```

ภาพประกอบ 60 คำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features_to_select ตั้งแต่ 1 – 20 ของการใช้เทคนิค RFE ร่วมกับ XGBoost

```
RFE_LR = pd.DataFrame({"Number of selected Features": n_features_RFE})
RFE_LR['accuracy score'] = accuracy_LR_RFE
RFE_LR.sort_values(by = "Number of selected Features", ascending = 1)
No_selectedFeat = RFE_LR["Number of selected Features"]
ac_score = RFE_LR['accuracy score']
plt.figure(figsize=(8,4))
ax = plt.gca()
max_index = np.where(ac_score == max(ac_score))
ac_score_max = ac_score[max_index[0][0]]
No_selectedFeat_MAX = No_selectedFeat[max_index[0][0]]
maxvalue = '\n\n features ='+str(No_selectedFeat_MAX)+ ' , ' + '\n Accuracy =' + str(ac_score_max)
plt.annotate(maxvalue, xy = (No_selectedFeat_MAX, ac_score_max))
ax = plt.gca()
plt.title('Feature Selection (RFE) \n\n\n')
plt.ylabel('Cross validation score (Accuracy)')
RFE_LR.plot(kind='line', x = "Number of selected Features", y = 'accuracy score', ax = ax)
plt.show()
```

ภาพประกอบ 61 ตัวอย่างคำสั่งสร้างกราฟแสดงค่า cross validation score ในแต่ละ n_features_to_select

4.2.2 Chi-square

ในงานวิจัยนี้ได้แสดงคะแนน chi-square test ของแต่ละฟีเจอร์ โดยแสดงค่าแยกกันในแต่ละ training size

จากนั้นทำการตรวจสอบค่า cross validation score (accuracy) (โดยกำหนด cross validation = 5 folds) เมื่อใช้จำนวนฟีเจอร์ (k) ตั้งแต่ 1 – 20 โดยตัวอย่างคำสั่งของการตรวจสอบค่า cross validation score (accuracy) ที่จำนวนฟีเจอร์ตั้งแต่ 1-20 เป็นไปตามภาพประกอบ 62 – 66 ซึ่งคำสั่งที่ใช้ในแต่ละแบบจำลองนั้น แตกต่างกันอย่างชัดเจนในส่วนที่เป็น estimator ของฟังก์ชัน “cross_val_score()”

โดยหลังจากตรวจสอบค่า cross validation score (accuracy) ในแต่ละจำนวนฟีเจอร์แล้ว จึงนำค่าที่ได้ไปสร้างเป็นกราฟ โดยใช้เครื่องมือที่ชื่อว่า “Microsoft Office Excel 365”

หลังจากนั้นจึงเลือกจำนวนฟีเจอร์ที่เหมาะสม โดยกำหนดให้ศึกษาจำนวนฟีเจอร์ในช่วงที่ค่า cross validation score (accuracy) เริ่มมีความคงที่ หรือเริ่มมากกว่า 0.9000 เป็นต้นไป โดยจำนวนฟีเจอร์ที่ศึกษานั้นต้องไม่มากเกินไป เนื่องจากหากใช้จำนวนฟีเจอร์ที่มากเกินไปอาจส่งผลให้แบบจำลองที่ได้มีโอกาสเกิดภาวะ overfitting ได้ จากนั้นจึงตรวจสอบประสิทธิภาพ

```
n_features = []
accuracy_LR_chi2 = []
for n_feature in range(1,21):
    fs_chi = SelectKBest(chi2, k= n_feature)
    X_Ord_train_scale_Chi2_LR = fs_chi.fit_transform(X_Ord_train_scale, y_train)
     LR = LogisticRegression(max_iter = 10000, random_state = 1, solver ='newton-cg',
                                    penalty = 'none', C = 22.456734693877554)
    accu_score = cross_val_score(LR, X_Ord_train_scale_Chi2_LR, y_train,
                                            cv = 5, scoring="accuracy")
    accLR = np.mean(accu_score)
    accuracy_LR_chi2.append(accLR)
    n_features.append(n_feature)
print(n_features)  ##เพื่อนำค่าไปสร้าง visualization โดยใช้ excel
print(accuracy_LR_chi2)
```

ภาพประกอบ 62 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Logistic regression

```
n_features = []
accuracy_svm_chi2 = []
for n_feature in range(1,21):
    fs_chi = SelectKBest(chi2, k= n_feature)
    X_Chi2_SVM = fs_chi.fit_transform(X_Ord_train_scale, y_train)
     SVM = SVC(kernel= 'poly', gamma = 5.0, degree=3.6666666666666665,
                        C=73.68684210526315, random_state=1, probability=True)
    accu_score = cross_val_score(SVM, X_Chi2_SVM, y_train, cv = 5, scoring="accuracy")
    accSVM = np.mean(accu_score)
    accuracy_svm_chi2.append(accSVM)
    n_features.append(n_feature)
print(n_features)
print(accuracy_svm_chi2)
```

ภาพประกอบ 63 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Support Vector Machine

```
n_features = []
accuracy_DT_chi2 = []
for n_feature in range(1,21):
    fs_chi = SelectKBest(chi2, k= n_feature)
    X_Ord_train_scale_Chi2_DT = fs_chi.fit_transform(X_Ord_train_scale, y_train)
     dt = DecisionTreeClassifier(min_samples_split = 15, max_leaf_nodes = 46,
                                        max_features = 'sqrt', max_depth =30)
    accu_score = cross_val_score(dt, X_Ord_train_scale_Chi2_DT, y_train,
                                            cv = 5, scoring="accuracy")
    accDT = np.mean(accu_score)
    accuracy_DT_chi2.append(accDT)
    n_features.append(n_feature)
print(n_features)
print(accuracy_DT_chi2)
```

ภาพประกอบ 64 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ค่า k ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Decision tree

```
n_features = []
accuracy_RF_chi2 = []
for n_feature in range(1,21):
    fs_chi = SelectKBest(chi2, k= n_feature)
    X_Ord_train_scale_Chi2_RF = fs_chi.fit_transform(X_Ord_train_scale, y_train)
     rf_model = RandomForestClassifier(n_estimators= 300, n_jobs=-1, min_samples_split = 71,
                                       max_leaf_nodes = 57, max_features = log2, max_depth =52)
    accu_score = cross_val_score(rf_model, X_Ord_train_scale_Chi2_RF, y_train,
                                       cv = 5, scoring="accuracy")
    accRF = np.mean(accu_score)
    accuracy_RF_chi2.append(accRF)
    n_features.append(n_feature)
print(n_features)
print(accuracy_RF_chi2)
```

ภาพประกอบ 65 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ Random Forest

```
n_features = []
accuracy_XGB_chi2 = []
for n_feature in range(1,21):
    fs_chi = SelectKBest(chi2, k= n_feature)
    X_Ord_train_scale_Chi2_XGB = fs_chi.fit_transform(X_Ord_train_scale, y_train)
     xg =xgb.XGBClassifier(n_estimators=300,subsample = 0.5,min_child_weight= 1,
                           max_depth= 3,learning_rate=0.22749999999999998,colsample_bytree=1)
    accu_score = cross_val_score(xg, X_Ord_train_scale_Chi2_XGB, y_train,
                           cv = 5, scoring="accuracy")
    accXGB = np.mean(accu_score)
    accuracy_XGB_chi2.append(accXGB)
    n_features.append(n_feature)
print(n_features)
print(accuracy_XGB_chi2)
```

ภาพประกอบ 66 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) ในการใช้ n_features ตั้งแต่ 1 – 20 ของการใช้เทคนิค chi-square ร่วมกับ XGBoost

4.3 การสกัดฟีเจอร์ (Feature Extraction)

ในงานวิจัยนี้ใช้เทคนิคการสกัดฟีเจอร์คือ Principle Component Analysis (PCA) ร่วมกับทั้ง 5 แบบจำลอง ซึ่งได้นำค่า Hyperparameter ที่เหมาะสมของแต่ละแบบจำลองมาใช้ในขั้นตอนการสกัดฟีเจอร์

โดยได้ตรวจสอบค่า cross validation score (accuracy) (โดยกำหนด cross validation = 5 folds) เมื่อใช้จำนวน n_components ตั้งแต่ 1 - 20 โดยตัวอย่างคำสั่งในการตรวจสอบค่า cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 - 20 ซึ่งเป็นไปตามภาพประกอบ 67 - 71 ซึ่งคำสั่งที่ใช้ในแต่ละแบบจำลองนั้น แตกต่างกันอย่างชัดเจนในส่วนที่เป็น estimator ของฟังก์ชัน “cross_val_score()”

เมื่อตรวจสอบค่า cross validation score (accuracy) ในแต่ละจำนวน n_components แล้ว จึงนำค่าที่ได้ไปสร้างเป็นกราฟ โดยใช้เครื่องมือที่ชื่อว่า “Microsoft Office Excel 365”

จากนั้นจึงเลือกจำนวน n_components ที่เหมาะสม โดยกำหนดให้ศึกษาจำนวน n_components ในช่วงที่ค่า cross validation score (accuracy) เริ่มมีความคงที่ หรือเริ่มมากกว่า 0.9000 เป็นต้นไป โดยจำนวน n_components ที่ศึกษานั้นต้องไม่มากเกินไป เนื่องจากหากใช้จำนวนฟีเจอร์ที่มากเกินไปอาจส่งผลให้แบบจำลองที่ได้มีโอกาสเกิดภาวะ overfitting ได้ จากนั้นจึงแสดง Component loading ของการใช้จำนวน n_components ที่เลือก แล้วจึงตรวจสอบประสิทธิภาพ

```
n_components = []
accuracy_LR_PCA = []
for nc in range(1,21):
    pca = PCA(n_components = nc)
    X_Ord_train_scale_PCA_LR = pca.fit_transform(X_Ord_train_scale, y_train)
     LR = LogisticRegression(max_iter = 10000, random_state = 1, solver = 'newton-cg',
                                penalty = 'none', C = 22.456734693877554)
    acscore = cross_val_score(LR, X_Ord_train_scale_PCA_LR, y_train, cv = 5, scoring="accuracy"
    accLR = np.mean(acscore)
    accuracy_LR_PCA.append(accLR)
    n_components.append(nc)
print(n_components)
print(accuracy_LR_PCA)
```

ภาพประกอบ 67 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้ n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Logistic regression

```
n_components = []
accuracy_SVM_PCA = []
for nc in range(1,21):
    pca = PCA(n_components = nc)
    X_Ord_train_scale_PCA_SVM = pca.fit_transform(X_Ord_train_scale, y_train)
     SVM = SVC(kernel= 'poly', gamma = 5.0, degree=3.6666666666666665,
                        C=73.68684210526315, random_state=1, probability=True)
    acscore = cross_val_score(SVM, X_Ord_train_scale_PCA_SVM, y_train,
                                        cv = 5, scoring="accuracy")
    accSVM = np.mean(acscore)
    accuracy_SVM_PCA.append(accSVM)
    n_components.append(nc)
print(n_components)
print(accuracy_SVM_PCA)
```

ภาพประกอบ 68 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้ n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Support Vector Machine

```
n_components = []
accuracy_dt_PCA = []
for nc in range(1,21):
    pca = PCA(n_components = nc)
    X_Ord_train_scale_PCA_dt = pca.fit_transform(X_Ord_train_scale, y_train)
     dt = DecisionTreeClassifier(min_samples_split = 15, max_leaf_nodes = 46,
                                        max_features = 'sqrt', max_depth =30)
    acscore = cross_val_score(dt, X_Ord_train_scale_PCA_dt, y_train, cv = 5, scoring="accuracy")
    accdt = np.mean(acscore)
    accuracy_dt_PCA.append(accdt)
    n_components.append(nc)
print(n_components)
print(accuracy_dt_PCA)
```

ภาพประกอบ 69 ตัวอย่างคำสั่งการตรวจสอบ cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Decision tree

```
n_components = []
accuracy_RF_PCA = []
for nc in range(1,21):
    pca = PCA(n_components = nc)
    X_Ord_train_scale_PCA_RF = pca.fit_transform(X_Ord_train_scale, y_train)
     rf_model = RandomForestClassifier(n_estimators= 300, n_jobs=-1, min_samples_split = 71,
                                       max_leaf_nodes = 57, max_features = log2, max_depth =52)
    acscore = cross_val_score(rf_model, X_Ord_train_scale_PCA_RF, y_train,
                                       cv = 5, scoring="accuracy")
    accRF = np.mean(acscore)
    accuracy_RF_PCA.append(accRF)
    n_components.append(nc)
print(n_components)
print(accuracy_RF_PCA)
```

ภาพประกอบ 70 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ Random Forest

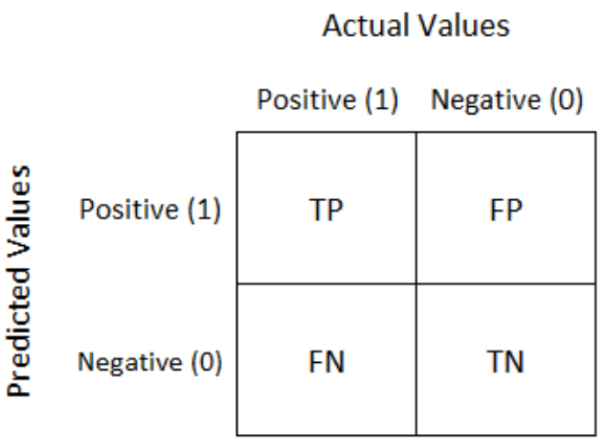
```
n_components = []
accuracy_XG_PCA = []
for nc in range(1,21):
    pca = PCA(n_components = nc)
    X_Ord_train_scale_PCA_XG = pca.fit_transform(X_Ord_train_scale, y_train)
     xg = xgb.XGBClassifier(n_estimators=300, subsample = 0.5, min_child_weight= 1,
                            max_depth= 3, learning_rate= 0.22749999999999998, colsample_bytree= 1)
    acscore = cross_val_score(xg, X_Ord_train_scale_PCA_XG, y_train,
                                       cv = 5, scoring="accuracy")
    accXG = np.mean(acscore)
    accuracy_XG_PCA.append(accXG)
    n_components.append(nc)
print(n_components)
print(accuracy_XG_PCA)
```

ภาพประกอบ 71 ตัวอย่างคำสั่งการตรวจสอบค่า cross validation score (accuracy) เมื่อใช้จำนวน n_components ตั้งแต่ 1 – 20 ของการใช้เทคนิค PCA ร่วมกับ XGBoost

4.4 การประเมินประสิทธิภาพแบบจำลอง

การวัดประสิทธิภาพแบบจำลองในงานวิจัยนี้ได้แสดงเป็น Confusion matrix, ค่า Accuracy, ค่า F1 score และ AUC (Area Under the Curve) score โดยมีรายละเอียดดังต่อไปนี้

4.4.1 Confusion matrix เป็นเครื่องมือสำคัญในการประเมินประสิทธิภาพของแบบจำลองเทคนิคการเรียนรู้ของเครื่อง โดยแสดงให้เห็นว่า สิ่งที่แบบจำลองทำนายผลลัพธ์ได้กับสิ่งที่เกิดขึ้นจริงมีค่าเป็นเท่าไร ดังแสดงในภาพประกอบ 72 [53]



ภาพประกอบ 72 confusion matrix

ที่มา : https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

- True Positive (TP) คือผลลัพธ์จากแบบจำลองคือเห็ดพิษ โดยประเภทของเห็ดคือเห็ดพิษ (แบบจำลองจำแนกประเภทได้ถูกต้อง)
- True Negative (TN) คือผลลัพธ์จากแบบจำลองคือเห็ดรับประทานได้ โดยประเภทของเห็ดคือเห็ดรับประทานได้ (แบบจำลองจำแนกประเภทได้ถูกต้อง)
- False Positive (FP) คือผลลัพธ์จากแบบจำลองคือเห็ดพิษ แต่ประเภทของเห็ดคือเห็ดรับประทานได้ (แบบจำลองจำแนกประเภทผิด)
- False Negative (FN) คือผลลัพธ์จากแบบจำลองคือเห็ดเห็ดรับประทานได้ แต่ประเภทของเห็ดคือเห็ดพิษ (แบบจำลองจำแนกประเภทผิด)

4.4.2 ค่า Accuracy และค่า F1 score: โดยค่า Accuracy เป็นค่าที่บอกประสิทธิภาพของแบบจำลองที่เรียบง่ายที่สุด โดยค่าที่วัดได้นั้นถูกแสดงเป็นสัดส่วนระหว่างจำนวนที่แบบจำลองสามารถทำนายได้ถูกต้องกับจำนวนข้อมูลทั้งหมด โดยหากค่า Accuracy มีค่าสูงเข้าใกล้ 1 แปลว่าแบบจำลองมีประสิทธิภาพดี ส่วนค่า F1 score นั้นเป็นค่าที่เกิดจากนำค่า Precision และ recall มาหาค่าเฉลี่ยแบบ harmonic ซึ่งหากค่า F1 score สูงแปลว่าค่า Precision และ ค่า Recall ก็สูงด้วย โดยหากค่า F1 score มีค่าสูงเข้าใกล้ 1 แปลว่าแบบจำลองมีประสิทธิภาพดี โดย ค่า Accuracy และค่า F1 score มีสูตรคำนวณตามสมการที่ 9 – 10 ตามลำดับ [53]

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{9}$$

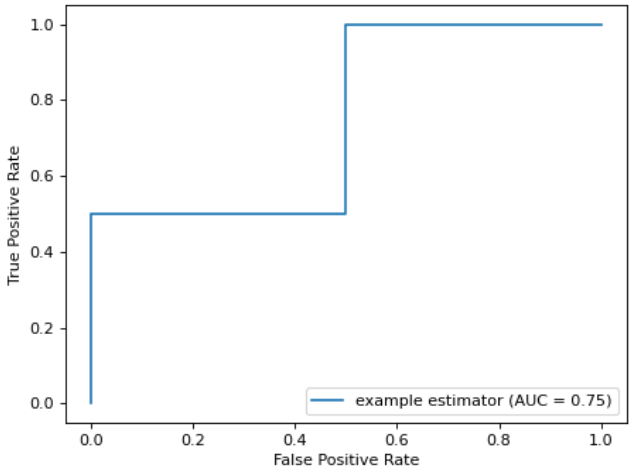
$$\text{F1 score} = 2 \text{ x } \frac{\text{Precision x Recall}}{\text{Precision + Recall}} \tag{10}$$

4.4.3 ROC_AUC curves (กราฟของ True Positive Rate (Recall) และ False Positive Rate) เพื่อดูค่า AUC (Area Under the Curve) score สำหรับตรวจสอบความสามารถของแบบจำลองว่าสามารถจำแนกประเภทของเห็ดทั้งสองประเภทออกจากกันได้ดีเท่าไร โดยหากค่ามากกว่า 0.5 แสดงว่า แบบจำลองมีประสิทธิภาพดีกว่าการสุ่มเดาผลลัพธ์ และถ้ามีค่าสูงเข้าใกล้ 1 แปลว่าแบบจำลองมีประสิทธิภาพดี [53]

สูตรการคำนวณของ Recall (True Positive Rate) และ False Positive Rate เป็นตามสมการที่ 11 และ 12 ตามลำดับ [53, 54] และตัวอย่างกราฟ ROC_AUC curves เป็นตามภาพประกอบ 73 [55]

$$\text{Recall (True Positive Rate)} = \frac{\text{TP}}{\text{TP+FN}} \tag{11}$$

$$\text{False Positive Rate} = \frac{\text{FP}}{\text{TN+FP}} \tag{12}$$



ภาพประกอบ 73 ตัวอย่าง ROC_AUC curves และ AUC score

ที่มา : https://scikit-learn.org/stable/modules/generated/sklearn.metrics.RocCurveDisplay.html

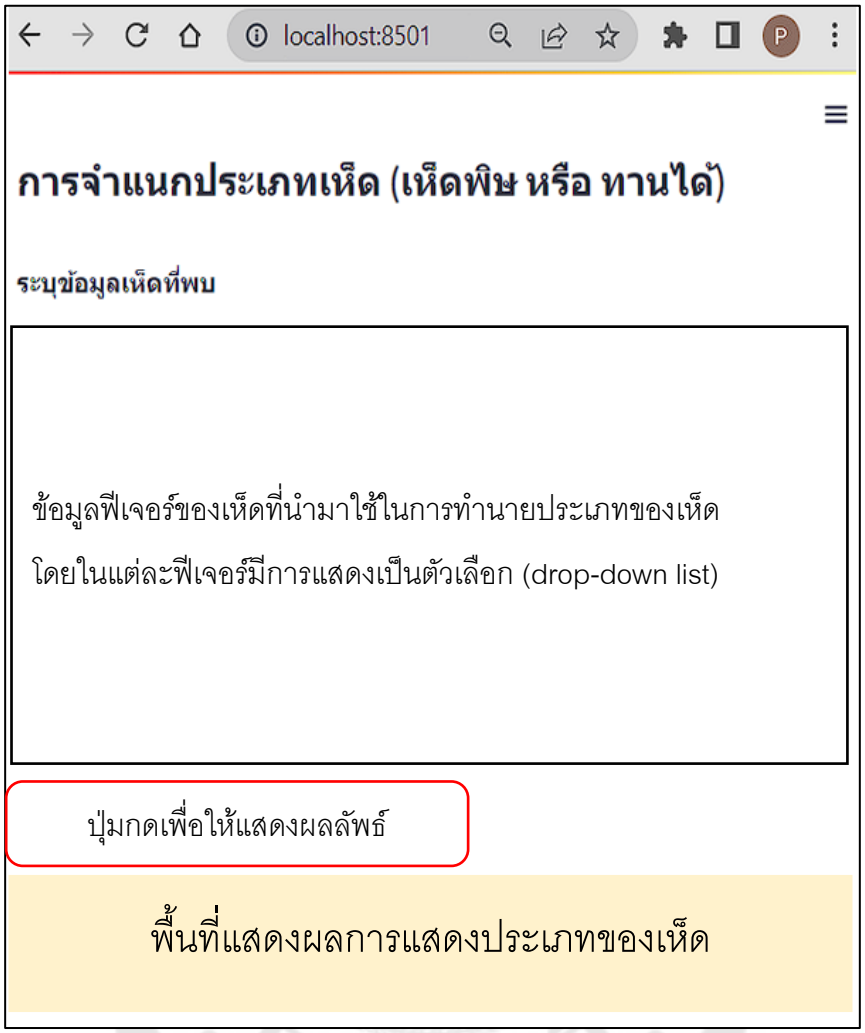
5. การพัฒนาแบบจำลองเป็นเว็บแอปพลิเคชัน

ในงานวิจัยนี้ผู้วิจัยสร้างตัวอย่างหน้าเว็บแอปพลิเคชันโดยใช้เครื่องมือที่ชื่อว่า “Streamlit” ซึ่งเป็น library ของภาษา python เหมาะกับการสร้างหน้าเว็บแอปพลิเคชันสำหรับเทคนิคการเรียนรู้ของเครื่อง [56]

องค์ประกอบหลักบนหน้าเว็บแอปพลิเคชัน ประกอบไปด้วย หัวข้อ, คำชี้แจง, ฟีเจอร์ที่ให้ผู้ใช้งานระบุโดยให้เลือกจากตัวเลือก (drop-down list), ปุ่มกดเพื่อให้แสดงผลลัพธ์เป็นประเภทของเห็ดระหว่างเห็ดกินได้ หรือ เห็ดพิษ และพื้นที่แสดงผลการแสดงประเภทของเห็ด โดยตัวอย่างแผนผังหน้าเว็บแอปพลิเคชันสามารถแสดงได้ตามภาพประกอบ 74

แบบจำลองที่เลือกนำมาใช้ในการจำแนกประเภทของเห็ดบนหน้าเว็บแอปพลิเคชันนั้น ได้จากการพิจารณาจากผลการศึกษาในการคัดเลือกฟีเจอร์และการสกัดฟีเจอร์ ที่ให้ประสิทธิภาพดี, เป็นเทคนิคที่มีความเหมาะสม รวมถึงฟีเจอร์ที่ใช้มีความเหมาะสม

หลังจากที่นำแบบจำลองไปไว้ที่หน้าเว็บแอปพลิเคชันจึงมีการทดลองการใช้งานเบื้องต้น โดยการทดสอบนั้นเริ่มจากการที่เลือกฟีเจอร์จากตัวเลือกที่ทำให้ได้ผลลัพธ์เป็น เห็ดพิษ กับ เห็ดรับประทานได้อย่างแน่นอน โดยตรวจสอบได้จากผลการวิเคราะห์เชิงสำรวจ



ภาพประกอบ 74 ตัวอย่างแผนผังหน้าเว็บแอปพลิเคชัน

บทที่ 4
ผลการศึกษา

ในการวิจัยเรื่องการจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษและไม่มีพิษโดยใช้เทคนิคการเรียนรู้ของเครื่อง ผู้วิจัยได้ดำเนินการวิจัยโดยการศึกษาตามขั้นตอนต่าง ๆ ตลอดจนการวัดประสิทธิภาพ เพื่อให้บรรลุจุดประสงค์ของการวิจัยที่ได้กำหนดไว้ ได้ดังนี้

1. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ข้อมูลคุณลักษณะทั้งหมดของเห็ดครีบจากชุดข้อมูล
2. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Recursive Feature Elimination
3. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี chi-square
4. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้เทคนิคการสกัดฟีเจอร์ด้วยวิธี Principal Principal Component Analysis (PCA)

1. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ข้อมูลคุณลักษณะทั้งหมดของเห็ดครีบจากชุดข้อมูล

ในการสร้างแบบจำลองจำแนกประเภทเห็ดระหว่างเห็ดมีพิษและไม่มีพิษ โดยกำหนดให้ใช้ฟีเจอร์จำนวน 20 ฟีเจอร์ (ฟีเจอร์ที่เหลือหลังจากทำความสะอาด โดยลบ “veil_type” และ “stalk_root” ไปแล้ว) คือ สีของหมวกเห็ด (cap_color), ลักษณะพื้นผิวบนหมวกเห็ด (cap_surface), รูปทรงของหมวกเห็ด (cap_shape), สีของครีบเห็ด (gill_color), ความแนบชิดของครีบกับก้านดอกเห็ด (gill_attachment), ระยะห่างของครีบเห็ดแต่ละครีบ (gill_spacing), ขนาดของครีบเห็ด (gill_size), รูปทรงของก้านดอก (stalk_shape), สีพื้นผิวก้านดอกที่อยู่บริเวณเหนือวงแหวน (stalk_color_above_ring), สีของพื้นผิวก้านดอกที่อยู่บริเวณด้านล่างวงแหวน (stalk_color_below_ring), ลักษณะของพื้นผิวก้านดอกบริเวณเหนือวงแหวน (stalk_surface_above_ring), ลักษณะของพื้นผิวก้านดอกบริเวณด้านล่างวงแหวน (stalk_surface_below_ring), สีของเยื่อที่หุ้มดอกเห็ด (veil_color), จำนวนวงแหวนของเห็ด (ring_number), รูปร่างของวงแหวน (ring_type), รอยช้ำบนเห็ด (bruise), กลิ่นของเห็ด (odor), สีของรอยพิมพ์สปอร์ (spore_print_color), ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด (population), บริเวณที่ที่พบเจอเห็ดเจริญเติบโต (habitat) โดยกำหนดให้ ประเภทของเห็ด (class_ep) เป็นลาเบล (Label)

โดยในการศึกษานี้มีการแบ่งชุดข้อมูลเป็นชุดสำหรับสร้างแบบจำลอง (Training Set) และชุดข้อมูลสำหรับทดสอบแบบจำลอง (Test Set) ในอัตราส่วนต่างๆ จำนวน 3 อัตราส่วน คือ Training size = 50%, Training size = 60% และ Training size = 70% และในแต่ละอัตราส่วนมีการสร้างแบบจำลองด้วยอัลกอริธึมทั้งหมด 5 อัลกอริธึม ได้แก่ Logistic Regression, Support Vector Machine, Decision Tree, Random Forest และ XGBoost โดยมีการปรับจูนหาค่า hyperparameter ด้วยเครื่องมือ RandomizedSearchCV โดยรายละเอียดของ hyperparameter และประสิทธิภาพของแต่ละแบบจำลองนั้นได้แสดงในตาราง 4 และแสดงเป็น confusion matrix ตามภาพประกอบ 75

ตาราง 4 ประสิทธิภาพของแบบจำลองจำแนกประเภทของเห็ดโดยใช้ฟีเจอร์ทั้งหมด

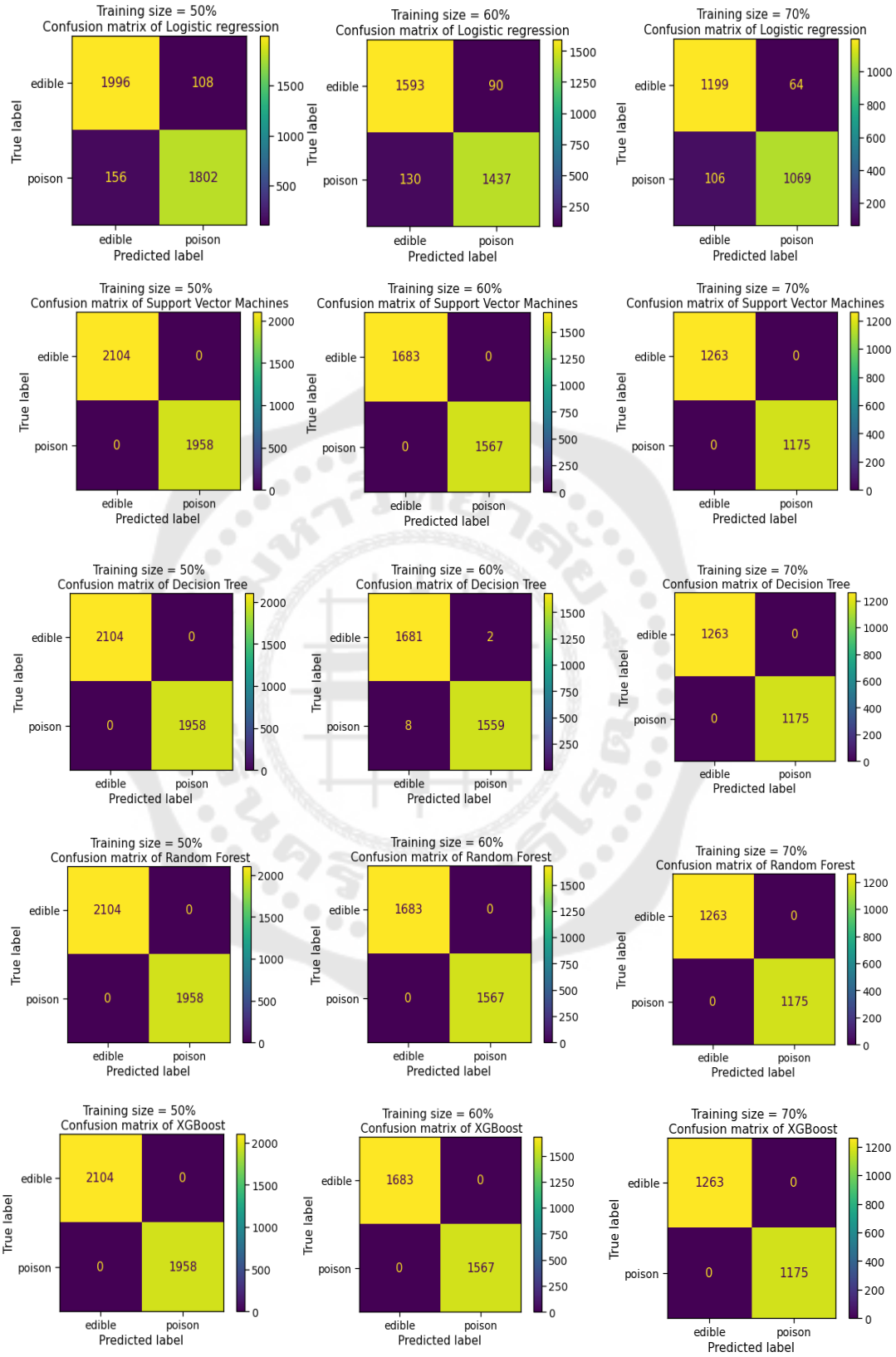
Training size	Algorithm	Hyperparameters	Accuracy (%)	F1 score (%)	AUC score (%)
50%	Logistic Regression	solver = 'newton-cg', penalty = 'none', C = 22.456734693877554	93.50	93.50	97.14
	Support Vector Machine	kernel= 'poly', gamma = 5.0, degree=3.6666666666666665, C=73.68684210526315	100	100	100
	Decision Tree	min_samples_split = 15, max_leaf_nodes = 46, max_features = 'sqrt', max_depth =30	100	100	100
	Random Forest	n_estimators= 300, min_samples_split = 71, max_leaf_nodes = 57, max_features = 'log2', max_depth =52	100	100	100
	XGBoost	n_estimators=300, subsample = 0.5, max_depth =3, min_child_weight= 1, learning_rate = 0.22749999999999998, colsample_bytree = 1	100	100	100

ตาราง 4 (ต่อ)

Training size	Algorithm	Hyperparameters	Accuracy (%)	F1 score (%)	AUC score (%)
60%	Logistic Regression	solver = 'newton-cg', penalty = 'none', C = 22.456734693877554	93.23	93.23	97.27
	Support Vector Machine	kernel = 'poly', gamma = 5.0, degree = 3.6666666666666665, C = 73.68684210526315	100	100	100
	Decision Tree	min_samples_split = 9, max_leaf_nodes = 35, max_features = 'log2', max_depth = 21	99.69	99.69	99.69
	Random Forest	n_estimators = 300, min_samples_split = 52, max_leaf_nodes = 43, max_features = 'log2', max_depth = 85	100	100	100
	XGBoost	n_estimators = 300, subsample = 0.5, min_child_weight = 1, max_depth = 3, learning_rate = 0.22749999999999998, colsample_bytree = 1	100	100	100

ตาราง 4 (ต่อ)

Training size	Algorithm	Hyperparameters	Accuracy (%)	F1 score (%)	AUC score (%)
70%	Logistic Regression	solver = 'newton-cg', penalty = 'none', C = 22.456734693877554	93.03	93.02	97.41
	Support Vector Machine	kernel = 'poly', gamma = 5.0, degree = 3.6666666666666665, C = 73.68684210526315	100	100	100
	Decision Tree	min_samples_split = 9, max_leaf_nodes = 35, max_features = 'log2', max_depth = 21	100	100	100
	Random Forest	n_estimators = 300, min_samples_split = 52, max_leaf_nodes = 43, max_features = 'log2', max_depth = 85	100	100	100
	XGBoost	n_estimator s=300, subsample = 1, min_child_weight = 10, max_depth = 4, learning_rate = 0.3, colsample_bytree = 1	100	100	100



ภาพประกอบ 75 Confusion matrix ของแบบจำลองที่ใช้ฟีเจอร์ทั้งหมด

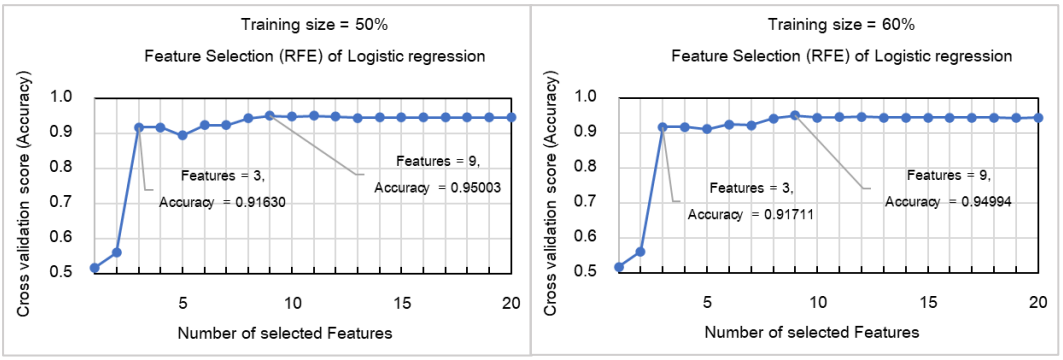
จากตาราง 3 และภาพประกอบ 75 พบว่าเมื่อใช้ฟีเจอร์ทั้งหมด 20 ฟีเจอร์ในการจำแนกประเภทของเห็ด แบบจำลองส่วนใหญ่ยกเว้น Logistic Regression ให้ประสิทธิภาพที่ดีมาก คือ 100% แต่อย่างไรก็ตามหากใช้ฟีเจอร์จำนวนมากเกินไป อาจทำให้แบบจำลองมีโอกาสเกิดภาวะ overfitting ได้ ซึ่งหากนำไปใช้งานจริงอาจไม่เหมาะสมและการจำแนกประเภทเห็ดก็อาจไม่มีความถูกต้องและแม่นยำ ดังนั้นการคัดเลือกและสกัดฟีเจอร์เพื่อหาฟีเจอร์ที่เหมาะสมนั้น สามารถลดโอกาสการเกิดภาวะ overfitting ได้

2. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Recursive Feature Elimination

เมื่อใช้เทคนิค Recursive Feature Elimination (RFE) ในการคัดเลือกฟีเจอร์ในการสร้างแบบจำลองจำนวน 5 แบบจำลอง ผลที่ได้พบว่า

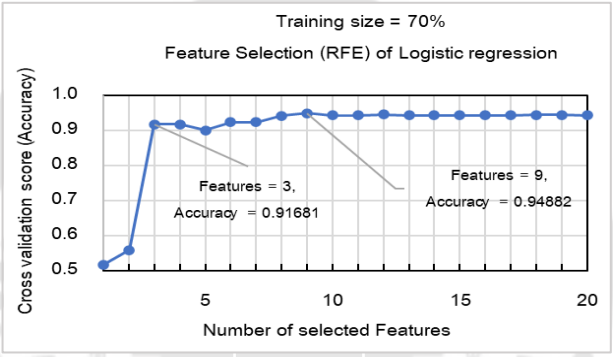
2.1 Logistic Regression จากภาพประกอบ 76 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 3 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดคือที่จำนวนฟีเจอร์ 9 ฟีเจอร์

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9500, 0.9499 และ 0.9488 ตามลำดับ



(ก)

(ข)



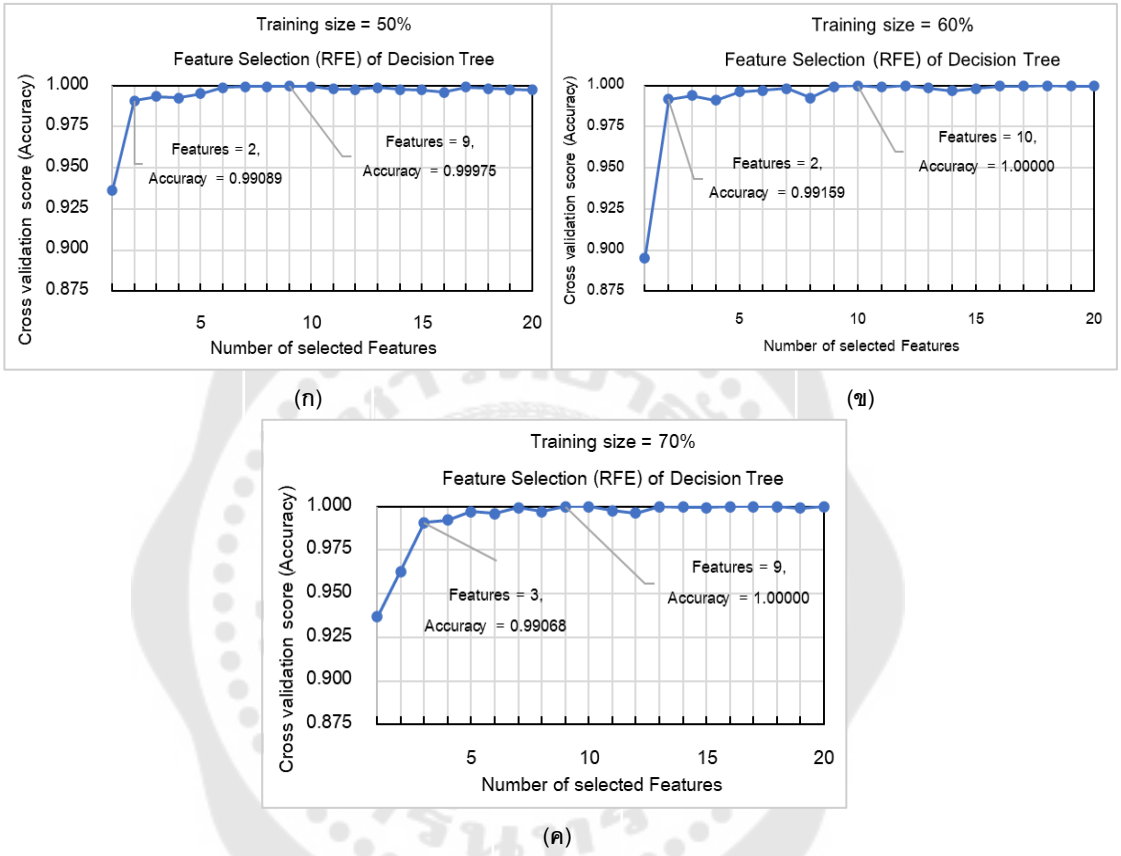
(ค)

ภาพประกอบ 76 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Logistic regression ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

2.2 Support Vector Machine ไม่สามารถใช้เทคนิค RFE ในการคัดเลือกฟีเจอร์ได้ เนื่องจาก Hyperparameter ที่ปรับจูนค่าได้นั้นพบว่า kernel เป็น Polynomial

2.3 Decision Tree จากภาพประกอบ 77 พบว่า การใช้ Training size ที่แตกต่างกัน จำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยที่ training size เท่ากับ 50% และ 60% ค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 2 เป็นต้นไป และสูงที่สุดเมื่อใช้ 9 และ 10 ฟีเจอร์ ตามลำดับ ส่วนที่ training size เท่ากับ 70% ค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 3 เป็นต้นไป และสูงที่สุดเมื่อใช้ 9 ฟีเจอร์

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9997, 1.0000 และ 1.0000 ตามลำดับ



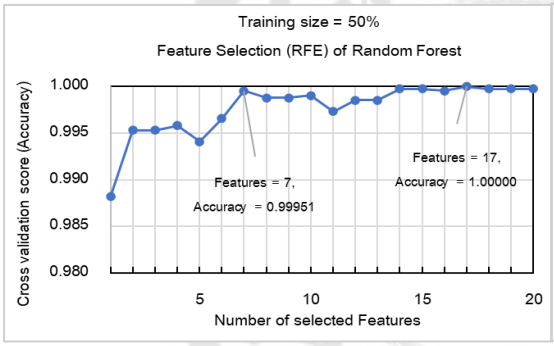
ภาพประกอบ 77 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Decision Tree ที่ training size ต่างๆ

(ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

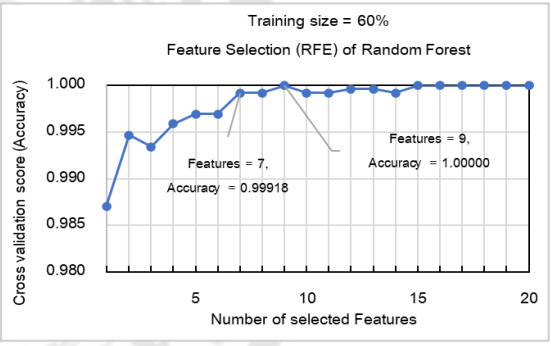
- Random Forest จากภาพประกอบ 78 พบว่า การใช้ Training size ที่แตกต่างกัน จำนวน 3 อัตราส่วน พบว่าที่ training size เท่ากับ 50% และ 60% นั้น เมื่อจำนวนฟีเจอร์ที่ใช้เพิ่มขึ้น ค่า cross validation score (accuracy) มีแนวโน้มขึ้นๆลงๆ ซึ่งที่ training size เท่ากับ 50% นั้น การใช้ฟีเจอร์เท่ากับ 7 ก็ทำให้ค่า cross validation score (accuracy) มีค่าที่ใกล้เคียงกับการใช้ฟีเจอร์เท่ากับ 17 ส่วนที่ training size เท่ากับ 60% นั้นการใช้ฟีเจอร์เท่ากับ 7 ก็ทำให้ค่า cross validation score (accuracy) มีค่าที่ใกล้เคียงกับการใช้ฟีเจอร์เท่ากับ 9

ส่วนที่ training size เท่ากับ 70% เมื่อจำนวนฟีเจอร์ที่ใช้เพิ่มขึ้น ค่า cross validation score (accuracy) มีแนวโน้มเพิ่มขึ้น และค่าถึงจุดสูงที่สุด คือเท่ากับ 1.0000 ที่จำนวนฟีเจอร์เท่ากับ 7

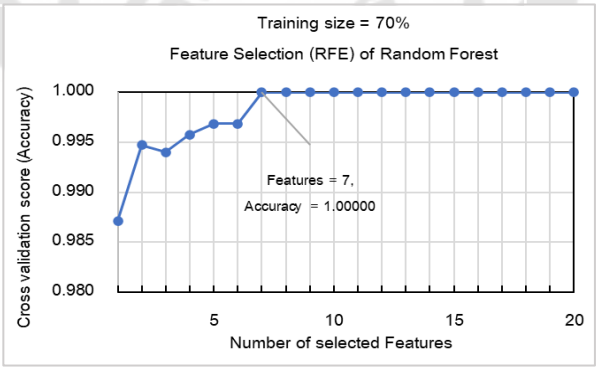
โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้น ให้ค่าเท่ากับ 1.0000



(ก)



(ข)

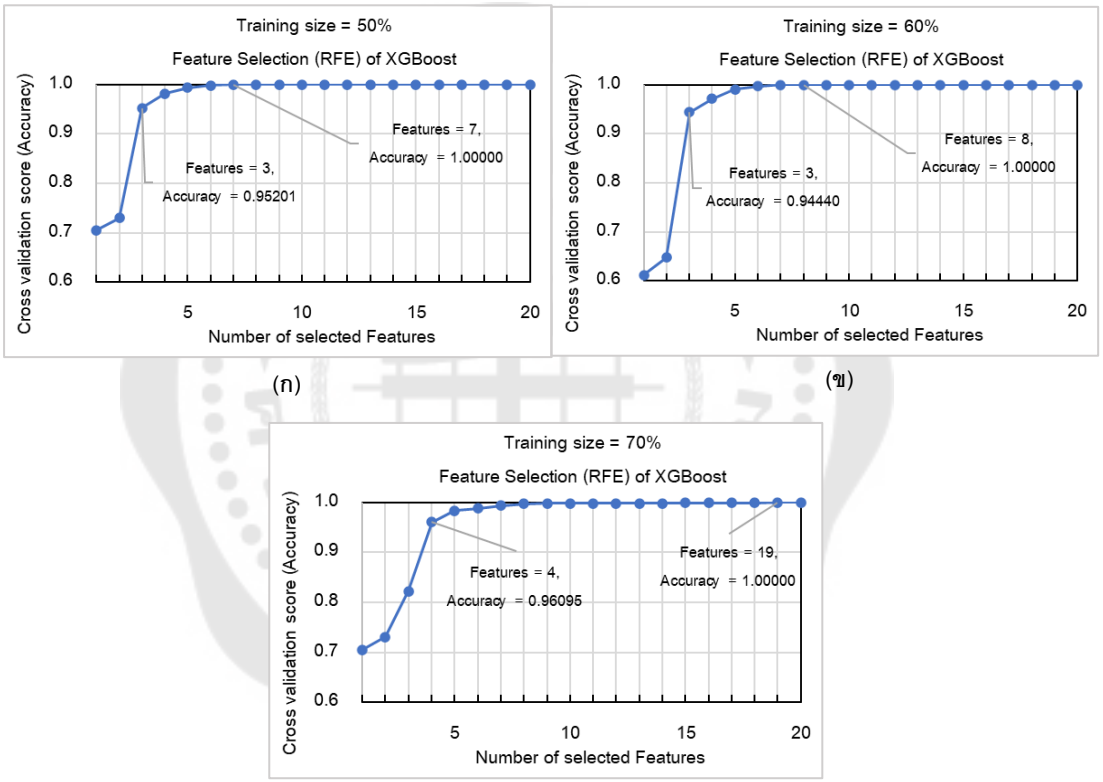


(ค)

ภาพประกอบ 78 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ Random Forest ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

- XGBoost จากภาพประกอบ 79 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยที่ training size เท่ากับ 50% และ 60% ค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 3 เป็นต้นไป และสูงที่สุดเมื่อใช้ 7 และ 8 ฟีเจอร์ ตามลำดับ ส่วนที่ training size เท่ากับ 70% ค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 4 เป็นต้นไป และสูงที่สุดเมื่อใช้ 19 ฟีเจอร์

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size เท่ากับ 1.0000



(ก) (ข)

(ค)

ภาพประกอบ 79 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค RFE ร่วมกับ XGBoost ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

จากนั้นจึงทำการเลือกจำนวนฟีเจอร์ที่นำมาสร้างแบบจำลองร่วมกับเทคนิค RFE และทำการทดสอบวัดประสิทธิภาพกับ Test Set ซึ่งจากกราฟในภาพประกอบ 76 – 79 พบว่าในทุกๆ แบบจำลองนั้นจำนวนฟีเจอร์ที่ทำให้ค่า cross validation score (accuracy) เริ่มมีความคงที่ และมากกว่า 0.90 นั้น ส่วนใหญ่เท่ากับ 2 และ 3 ฟีเจอร์ แต่อย่างไรก็ตามในการใช้แบบจำลอง Logistic regression พบว่าการใช้จำนวนฟีเจอร์เท่ากับ 2 นั้น ค่า cross validation score (accuracy) นั้นยังมีค่าค่อนข้างต่ำ คือน้อยกว่า 0.6000 และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ที่สูงที่สุด ส่วนใหญ่เท่ากับ 9 ฟีเจอร์ ดังนั้นจึงกำหนดขอบเขตจำนวนฟีเจอร์ที่นำไปศึกษาประสิทธิภาพต่อไป ในช่วง 3 – 9 ฟีเจอร์ โดยผู้วิจัยได้เลือกศึกษาทั้งหมด 4 ค่า คือ 3, 5, 7 และ 9 ซึ่งผลการศึกษาแสดงได้ดังตาราง 5 โดยพบว่า

สำหรับแบบจำลอง Logistic Regression การทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 9 และ training size เท่ากับ 70% โดยให้ค่า accuracy และ F1 score เท่ากับ 95.78% นอกจากนี้พบว่าการใช้จำนวนฟีเจอร์เป็น 5 มีประสิทธิภาพที่น้อยที่สุด ซึ่งสอดคล้องกับกราฟแสดงค่า cross validation score (accuracy) ดังภาพประกอบ 76

สำหรับแบบจำลอง Decision Tree เมื่อมีการใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้นเล็กน้อย โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด โดยใช้จำนวนฟีเจอร์น้อยที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 7 และ training size เท่ากับ 60% โดยให้ค่า accuracy และ F1 score เท่ากับ 100%

สำหรับแบบจำลอง Random Forest เมื่อมีการใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้นเล็กน้อย โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด โดยใช้จำนวนฟีเจอร์น้อยที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 7, training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 100%

สำหรับแบบจำลอง XGBoost ภาพรวมนั้นเมื่อมีการใช้จำนวนฟีเจอร์มากขึ้น ประสิทธิภาพมีแนวโน้มมากขึ้นเล็กน้อย แต่ในการทดลองที่ใช้จำนวนฟีเจอร์เท่ากันเมื่อ training size มากขึ้น พบว่าประสิทธิภาพลดลง โดยเฉพาะที่ training size เท่ากับ 70% ของการใช้จำนวนฟีเจอร์เท่ากับ 3 และ 5 พบว่าลดลงอย่างเห็นได้ชัด ซึ่งอาจเกิดจากสาเหตุของการแบ่งชุดข้อมูลที่ทำให้ training set มีข้อมูลฟีเจอร์ที่แตกต่างกัน โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด โดยใช้จำนวนฟีเจอร์น้อยที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 7 และ training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 100%

ตาราง 5 ประสิทธิภาพของแบบจำลอง ที่ใช้เทคนิคคัดเลือกฟีเจอร์ด้วยวิธี RFE

Algorithm	No. of features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Logistic Regression	3	50%	“veil_color“,“ring_number“, “spore_print_color“	92.02	92.00	91.85
		60%		92.00	92.00	91.13
		70%		92.14	92.17	91.34
	5	50%	“bruises“,“gill_spacing“, “veil_color“,“ring_number“, “spore_print_color“	87.79	87.71	93.25
		60%		87.51	87.42	92.93
		70%		87.37	87.27	93.01
	7	50%	“bruises“,“gill_spacing“, “stalk_surface_above_ring“, “veil_color“, “ring_type“, “ring_number“, “spore_print_color“	91.97	91.96	94.86
		60%		91.82	91.80	94.87
		70%		92.29	92.27	94.83
	9	50%	“cap_surface”, “bruises“, “gill_spacing“, “gill_size“, “stalk_surface_above_ring“, “veil_color“, “ring_type“, “ring_number“,“spore_print_color“	93.67	93.67	96.32
		60%		93.17	93.16	96.11
		70%		95.78	95.78	96.16
Support Vector Machine	3	50%	ไม่สามารถหาค่าได้ เนื่องจาก kernel = polynomial			
		60%				
		70%				
	5	50%				
		60%				
		70%				
	7	50%				
		60%				
		70%				
	9	50%				
		60%				
		70%				

ตาราง 5 (ต่อ)

Algorithm	No. of features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Decision Tree	3	50%	“odor“, “gill_size“, “spore_print_color”	99.29	99.29	99.99
		60%		99.32	99.32	99.99
		70%	“odor“, “spore_print_color” “population”	99.26	99.26	99.98
	5	50%	“odor“, “spore_print_color”, “stalk_color_below_ring“, “population“, “habitat“	99.51	99.51	99.99
		60%	“cap_color”, “odor“, “gill_size“, “spore_print_color”, “population”	99.72	99.72	100
		70%	"odor”, “gill_spacing”, “gill_size“, “stalk_color_above_ring”, “habitat“	99.34	99.34	99.99
	7	50%	“bruises”, “odor“, “gill_size“, “stalk_surface_above_ring”, “stalk_surface_below_ring”, “ring_type”, “spore_print_color”	99.63	99.63	100
		60%	“bruises”, “odor“, “gill_size“, “stalk_surface_below_ring”, “stalk_color_below_ring“, “spore_print_color”, “population”	100	100	100
		70%	"odor”, “gill_spacing”, “gill_size“, “stalk_color_above_ring”, “ring_type”, “spore_print_color”, “population”	100	100	100
	9	50%	bruises, “gill_spacing”, “gill_size“, “stalk_shape”, “habitat”, “stalk_surface_below_ring”, “stalk_color_below_ring“, “spore_print_color”, “population”	100	100	100

ตาราง 5 (ต่อ)

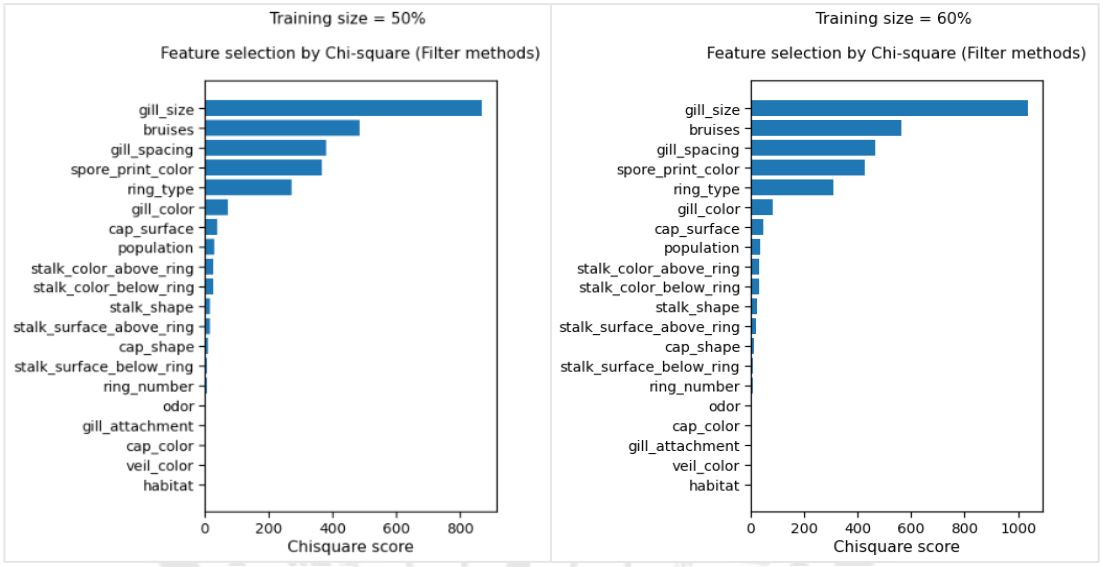
Algorithm	No. of features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Decision Tree	9	60%	“odor”, “stalk_shape”, “habitat”, “stalk_surface_below_ring”, “stalk_color_above_ring”, “ring_type”, “ring_number”, “spore_print_color”, “population”	100	100	100
		70%	“bruises”, “odor“, “gill_spacing”, “gill_size”, “stalk_shape”, habitat”, “ring_type”, “spore_print_color” “stalk_color_above_ring”,	100	100	100
Random Forest	3	50%	“odor“, “gill_size“, “spore_print_color”	99.29	99.29	99.99
		60%		99.32	99.32	99.99
		70%		99.26	99.26	99.99
	5	50%	“odor“, “gill_size“, “ring_type”, “spore_print_color”, “population”	99.51	99.51	99.98
		60%		99.48	99.48	99.99
		70%		99.43	99.43	99.99
	7	50%	“bruises”, “odor“, “gill_spacing”, “gill_size”, “ring_type”, “spore_print_color”, “population”	100	100	100
		60%		100	100	100
		70%		100	100	100
	9	50%	“bruises”, “odor“, “gill_spacing”, “gill_size”, “ring_type”, “stalk_surface_above_ring”, “stalk_surface_below_ring”, “spore_print_color”, “population”	100	100	100
		60%		100	100	100
		70%	“bruises”, “odor“, “gill_spacing”, “gill_size”, “ring_type”, “stalk_surface_below_ring”, “habitat”, “spore_print_color”, “population”	100	100	100

ตาราง 5 (ต่อ)

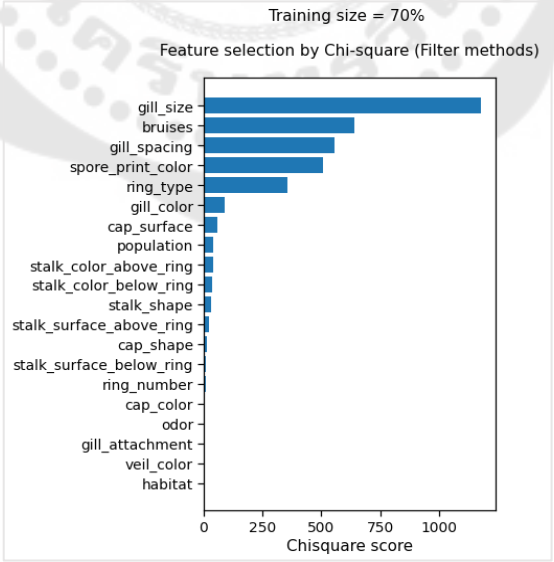
Algorithm	No. of features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
XGBoost	3	50%	“veil_color”, “ring_number”,	93.53	93.51	94.39
		60%	“spore_print_color”	93.60	93.58	94.47
		70%	“stalk_shape”, “veil_color”, “ring_number”	62.18	61.02	65.73
	5	50%	“odor“, “gill_size“, “veil_color”,	99.29	99.29	99.99
		60%	“ring_number”,“spore_print_color”	99.42	99.42	99.99
		70%	“gill_spacing”, “stalk_shape”, “veil_color”, “ring_number”, “spore_print_color”	95.20	95.19	98.75
	7	50%	“odor“, “gill_spacing”, “gill_size”, “veil_color”, “ring_number”, “spore_print_color”, ”population”	100	100	100
		60%	“odor“, “gill_spacing”, “gill_size”, “stalk_surface_above_ring”, “veil_color”, “ring_number”, “spore_print_color”	99.94	99.94	100
		70%	“odor“, “gill_spacing”, “gill_size”, “stalk_shape” , “veil_color”, “ring_number”, “spore_print_color”	99.84	99.84	100
	9	50%	“odor“, “gill_spacing”, ”population”	100	100	100
		60%	“stalk_surface_above_ring”, “veil_color”, “stalk_surface_below_ring”, “gill_size”, “ring_number”, “spore_print_color”	100	100	100
		70%	“cap_surface”, “odor“, “gill_size”, “gill_spacing”, “stalk_shape” “veil_color”, “ring_number”, “spore_print_color”, ”population”	100	100	100

3. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี chi-square

ในการคัดเลือกฟีเจอร์ด้วยวิธี chi-square ผู้วิจัยได้หาค่า chi-square ของแต่ละฟีเจอร์ของการทดลองในแต่ละ training size โดยสามารถแสดงได้เป็นกราฟดังภาพประกอบ 78 โดยพบว่าในทุกๆ training size นั้น ฟีเจอร์ที่มีความสำคัญมากที่สุด คือ “gill_size” และรองลงมาอีก 8 ลำดับ คือ “bruise”, “gill_spacing”, “spore_print_color”, “ring_type”, “gill_color”, “cap_surface”, “population” และ“stalk_color_above_ring” ตามลำดับ



(ก) **(ข)**



(ค)

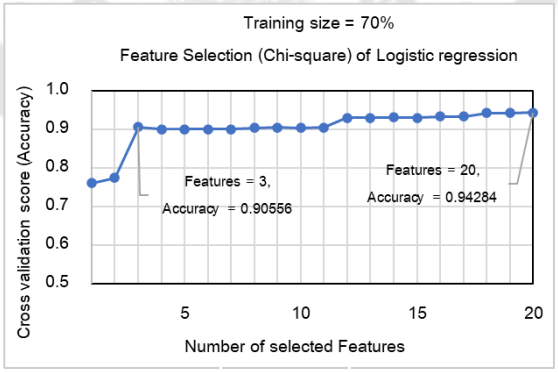
ภาพประกอบ 80 กราฟแสดงค่า chi-square ของฟีเจอร์ของเห็ดในแต่ละ training size (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

เมื่อใช้เทคนิค chi-square ในการคัดเลือกฟีเจอร์ในการสร้างแบบจำลองจำนวน 5 แบบจำลอง ผลที่ได้พบว่า

- Logistic Regression จากภาพประกอบ 81 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 3 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 19, 19 และ 20 ฟีเจอร์ ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9458, 0.9446 และ 0.9428 ตามลำดับ

(ก) (ข)

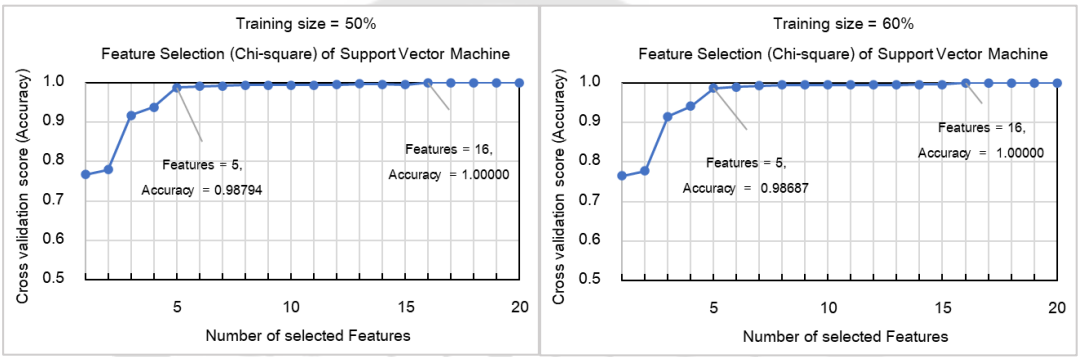


(ค)

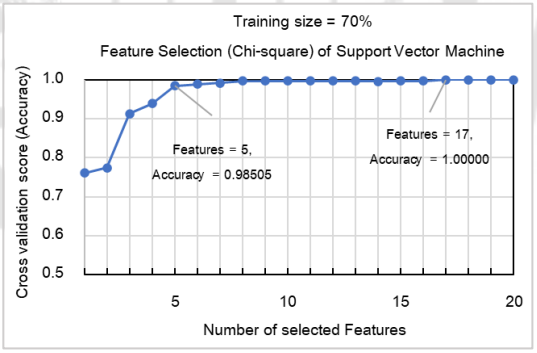
ภาพประกอบ 81 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Logistic regression ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

- Support Vector Machine จากภาพประกอบ 82 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มสูงมากกว่า 0.9000 ที่จำนวนฟีเจอร์เท่ากับ 3 และค่าเริ่มคงที่จำนวนฟีเจอร์เท่ากับ 5 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 16, 16 และ 17 ฟีเจอร์ ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นเท่ากับ 1.0000



(ก) (ข)

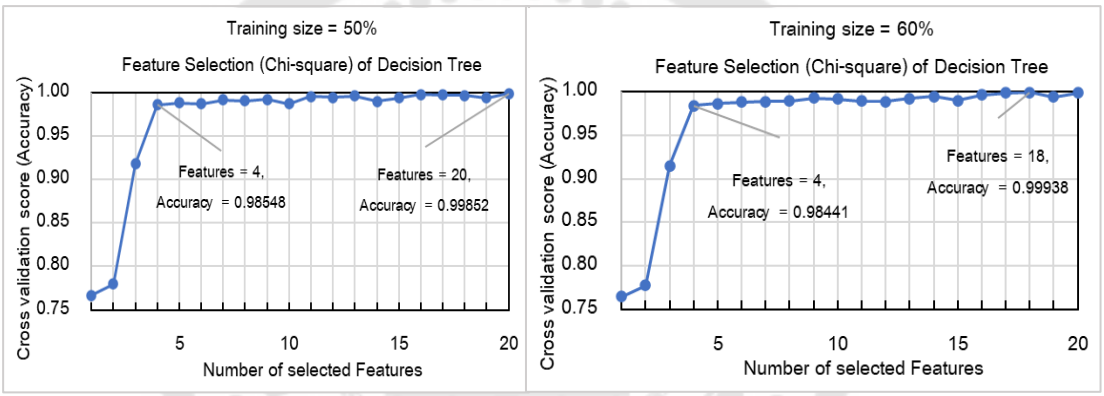


(ค)

ภาพประกอบ 82 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Support Vector Machine ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

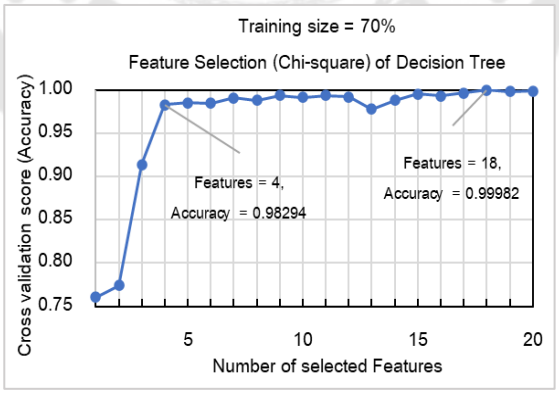
- Decision Tree จากภาพประกอบ 83 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มสูงมากกว่า 0.9000 ที่จำนวนฟีเจอร์เท่ากับ 3 และค่าเริ่มคงที่จำนวนฟีเจอร์เท่ากับ 4 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 20, 18 และ 18 ฟีเจอร์ ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9985, 0.9993 และ 0.9998 ตามลำดับ



(ก)

(ข)

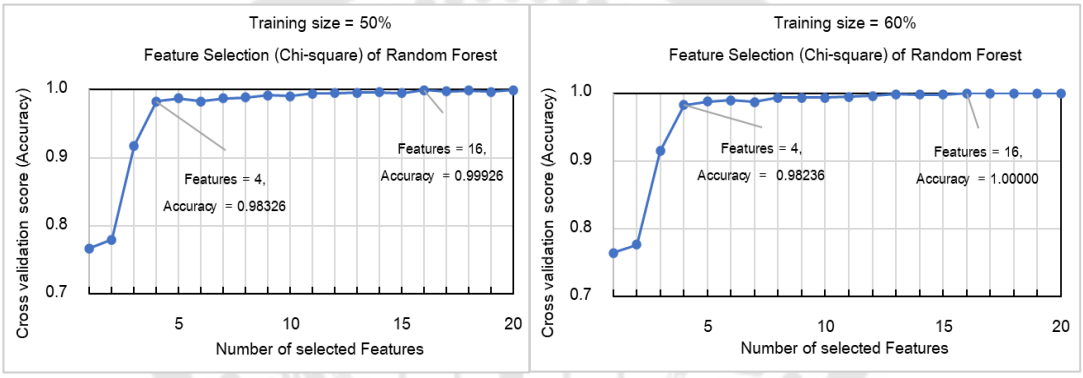


(ค)

ภาพประกอบ 83 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Decision Tree ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

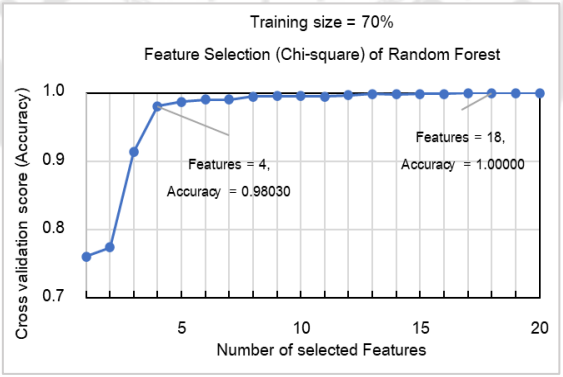
- Random Forest จากภาพประกอบ 84 พบว่า การใช้ Training size ที่แตกต่างกัน จำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มสูงมากกว่า 0.9000 ที่จำนวนฟีเจอร์เท่ากับ 3 และค่าเริ่มคงที่จำนวนฟีเจอร์เท่ากับ 4 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 16, 16 และ 18 ฟีเจอร์ ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9992, 1.0000 และ 1.0000 ตามลำดับ



(ก)

(ข)

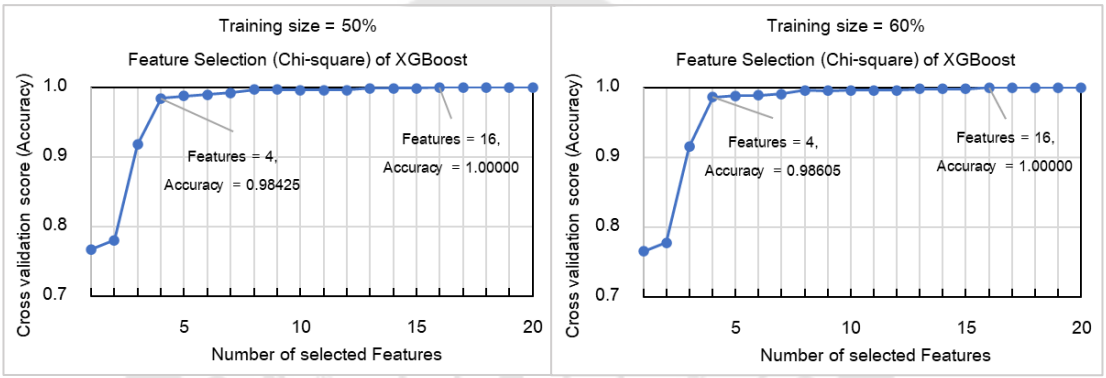


(ค)

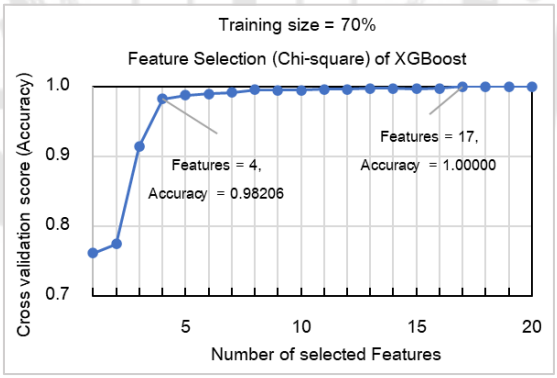
ภาพประกอบ 84 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ Random Forest ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

- XGBoost จากภาพประกอบ 85 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวนฟีเจอร์เพิ่มขึ้น โดยค่าเริ่มสูงมากกว่า 0.9000 ที่จำนวนฟีเจอร์เท่ากับ 3 และค่าเริ่มคงที่จำนวนฟีเจอร์เท่ากับ 4 เป็นต้นไป และจำนวนฟีเจอร์ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 16, 16 และ 17 ฟีเจอร์ ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นเท่ากับ 1.0000



(ก) (ข)



(ค)

ภาพประกอบ 85 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวนฟีเจอร์ ตั้งแต่ 1 - 20 ในการใช้เทคนิค chi-square ร่วมกับ XGBoost ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

จากนั้นจึงทำการเลือกจำนวนฟีเจอร์ที่นำมาสร้างแบบจำลองร่วมกับการใช้เทคนิค chi-square และทำการทดสอบวัดประสิทธิภาพกับ Test Set ซึ่งจากกราฟในภาพประกอบ 80 – 85 พบว่าในทุกๆ แบบจำลองนั้น จำนวนฟีเจอร์ที่ทำให้ค่า cross validation score (accuracy) เริ่มมากกว่า 0.9000 นั้น เท่ากับ 3 ฟีเจอร์ และค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์เท่ากับ 4 ฟีเจอร์ (ยกเว้นแบบจำลอง Logistic Regression เท่ากับ 5 ฟีเจอร์) ซึ่งหลังจากค่าเริ่มคงที่นั้น แม้ว่าจำนวนฟีเจอร์มากขึ้น ค่า cross validation score (accuracy) นั้นก็ไม่ได้มากขึ้นอย่างเห็นได้ชัด และหากให้จำนวนฟีเจอร์ที่มากเกินไปก็อาจทำให้แบบจำลองเกิดภาวะ overfitting ได้มาก ดังนั้นจึงกำหนดขอบเขตจำนวนฟีเจอร์ที่นำไปศึกษาประสิทธิภาพต่อไป ในช่วง 3 – 9 ฟีเจอร์ โดยผู้วิจัยได้เลือกศึกษาทั้งหมด 4 ค่า คือ 3, 5, 7 และ 9 ซึ่งผลการศึกษาแสดงได้ดังตาราง 6 โดยพบว่า

สำหรับแบบจำลอง Logistic Regression พบว่าการใช้จำนวนฟีเจอร์ที่แตกต่างกันนั้นประสิทธิภาพไม่มีความแตกต่างกัน แต่การใช้ training size ที่แตกต่างกันมีผลต่อประสิทธิภาพ คือ ที่ training size เท่ากับ 50% พบว่าให้ประสิทธิภาพที่ดีกว่า 60% และ 70% ตามลำดับ (เมื่อควบคุมจำนวนฟีเจอร์ให้เท่ากัน) ซึ่งอาจเกิดจากสาเหตุของการแบ่งชุดข้อมูลที่ทำให้ training set มีข้อมูลฟีเจอร์ที่แตกต่างกัน โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 3 และ training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 89.34%

สำหรับแบบจำลอง Support Vector Machine เมื่อใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวนฟีเจอร์ตั้งแต่ 5 ฟีเจอร์ขึ้นไป และการใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 9 และ training size เท่ากับ 60% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.79% และ 99.78% ตามลำดับ

สำหรับแบบจำลอง Decision Tree เมื่อใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวนฟีเจอร์ตั้งแต่ 5 ฟีเจอร์ขึ้นไป แต่การใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 9 และ training size เท่ากับ 50%, 70% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.75%

สำหรับแบบจำลอง Random Forest เมื่อใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวนฟีเจอร์ตั้งแต่ 5 ฟีเจอร์ขึ้นไป แต่การใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่

จำนวนฟีเจอร์เท่ากับ 7 และ training size เท่ากับ 70% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.34%

สำหรับแบบจำลอง XGBoost เมื่อมีการใช้จำนวนฟีเจอร์มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวนฟีเจอร์ตั้งแต่ 5 ฟีเจอร์ขึ้นไป แต่การใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 9 และ training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.66%

ตาราง 6 ประสิทธิภาพของแบบจำลอง ที่ใช้เทคนิคคัดเลือกฟีเจอร์ด้วยวิธี Chi-square

Algorithm	No. of Features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Logistic Regression	3	50%	“gill_size”, “bruise”, “gill_spacing”	89.34	89.34	91.39
		60%		88.62	88.60	91.22
		70%		88.23	88.22	90.92
	5	50%	“gill_size”, “bruise”, “gill_spacing”,	88.40	88.40	95.14
		60%	“spore_print_color”, “ring_type”	88.09	88.09	95.17
		70%		87.74	87.73	94.87
	7	50%	“gill_size”, “bruise”, “gill_spacing”,	88.53	88.52	94.52
		60%	“spore_print_color”, “ring_type”,	88.22	88.21	94.40
		70%	“gill_color”, “cap_surface”	87.74	87.73	94.12
	9	50%	“gill_size”, “bruise”, “gill_spacing”,	89.19	89.19	94.31
		60%	“spore_print_color”, “ring_type”,	88.86	88.86	94.28
		70%	“gill_color”, “cap_surface”, “population”, “stalk_color_above_ring”	88.56	88.55	94.35

ตาราง 6 (ต่อ)

Algorithm	No. of Features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Support Vector Machine	3	50%	“gill_size”, “bruise”, “gill_spacing”	89.34	89.34	91.34
		60%		89.11	89.11	89.34
		70%		88.64	88.64	90.85
	5	50%	“gill_size”, “bruise”, “gill_spacing”,	98.55	98.55	99.07
		60%	“spore_print_color”, “ring_type”	98.52	98.52	98.42
		70%		98.56	98.56	98.44
	7	50%	“gill_size”, “bruise”, “gill_spacing”,	99.26	99.26	99.92
		60%	“spore_print_color”, “ring_type”,	99.63	99.63	99.91
		70%	“gill_color”, “cap_surface”	99.67	99.67	99.91
	9	50%	“gill_size”, “bruise”, “gill_spacing”,	99.50	99.50	99.76
		60%	“spore_print_color”, “ring_type”,	99.79	99.78	99.98
		70%	“gill_color”, “cap_surface”, “population”, “stalk_color_above_ring”	99.75	99.75	99.98
Decision Tree	3	50%	“gill_size”, “bruise”, “gill_spacing”	89.34	89.34	92.03
		60%		89.11	89.11	91.86
		70%		88.64	88.64	91.58
	5	50%	“gill_size”, “bruise”, “gill_spacing”,	98.55	98.55	99.96
		60%	“spore_print_color”, “ring_type”	98.52	98.52	99.97
		70%		98.56	98.56	99.97
	7	50%	“gill_size”, “bruise”, “gill_spacing”,	98.79	98.79	99.96
		60%	“spore_print_color”, “ring_type”,	99.63	99.63	99.99
		70%	“gill_color”, “cap_surface”	99.67	99.67	99.99
	9	50%	“gill_size”, “bruise”, “gill_spacing”,	99.75	99.75	99.84
		60%	“spore_print_color”, “ring_type”,	99.69	99.69	100.00
		70%	“gill_color”, “cap_surface”, “population”, “stalk_color_above_ring”	99.75	99.75	99.87

ตาราง 6 (ต่อ)

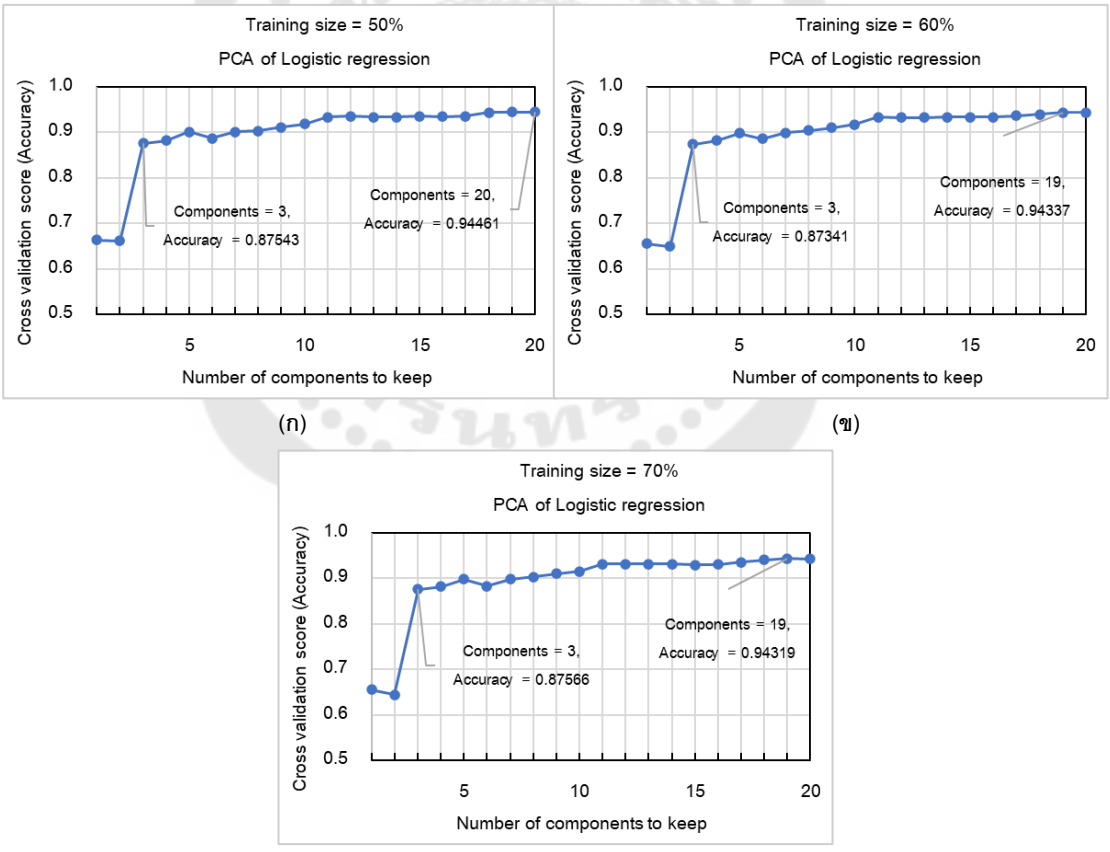
Algorithm	No. of Features	Training size	Features name	Accuracy (%)	F1 score (%)	AUC score (%)
Random Forest	3	50%	“gill_size”, “bruise”, “gill_spacing”	89.34	89.34	92.03
		60%		89.11	89.11	92.03
		70%		88.64	88.64	91.58
	5	50%	“gill_size”, “bruise”, “gill_spacing”,	98.47	98.47	99.95
		60%	“spore_print_color”, “ring_type”	98.47	97.47	99.95
		70%		98.48	98.48	99.97
	7	50%	“gill_size”, “bruise”, “gill_spacing”,	98.52	98.52	99.98
		60%	“spore_print_color”, “ring_type”,	98.52	98.52	99.98
		70%	“gill_color”, “cap_surface”	99.34	99.34	99.98
	9	50%	“gill_size”, “bruise”, “gill_spacing”,	98.82	98.82	99.98
		60%	“spore_print_color”, “ring_type”,	98.82	98.82	99.98
		70%	“gill_color”, “cap_surface”, “population”, “stalk_color_above_ring”	99.02	99.02	100
XGBoost	3	50%	“gill_size”, “bruise”, “gill_spacing”	89.34	89.34	92.03
		60%		89.11	89.11	91.86
		70%		88.64	88.64	91.43
	5	50%	“gill_size”, “bruise”, “gill_spacing”,	98.55	98.55	99.96
		60%	“spore_print_color”, “ring_type”	98.52	98.52	99.97
		70%		98.56	98.56	99.97
	7	50%	“gill_size”, “bruise”, “gill_spacing”,	99.36	99.36	99.99
		60%	“spore_print_color”, “ring_type”,	99.45	99.45	99.99
		70%	“gill_color”, “cap_surface”	99.63	99.63	99.99
	9	50%	“gill_size”, “bruise”, “gill_spacing”,	99.66	99.66	100
		60%	“spore_print_color”, “ring_type”,	99.63	99.63	100
		70%	“gill_color”, “cap_surface”, “population”, “stalk_color_above_ring”	99.55	99.55	100

4. ผลลัพธ์ของการสร้างแบบจำลองโดยใช้เทคนิคการสกัดฟีเจอร์ด้วยวิธี Principal Component Analysis (PCA)

เมื่อใช้วิธี PCA ในการสกัดฟีเจอร์สำหรับการสร้างแบบจำลอง 5 แบบจำลอง พบว่า

- Logistic Regression จากภาพประกอบ 86 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวน components เพิ่มขึ้น โดยค่าเริ่มคงที่เมื่อใช้จำนวนฟีเจอร์ตั้งแต่ 3 เป็นต้นไป และจำนวน components ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือ 20, 19 และ 19 components ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9446, 0.9433 และ 0.9431 ตามลำดับ

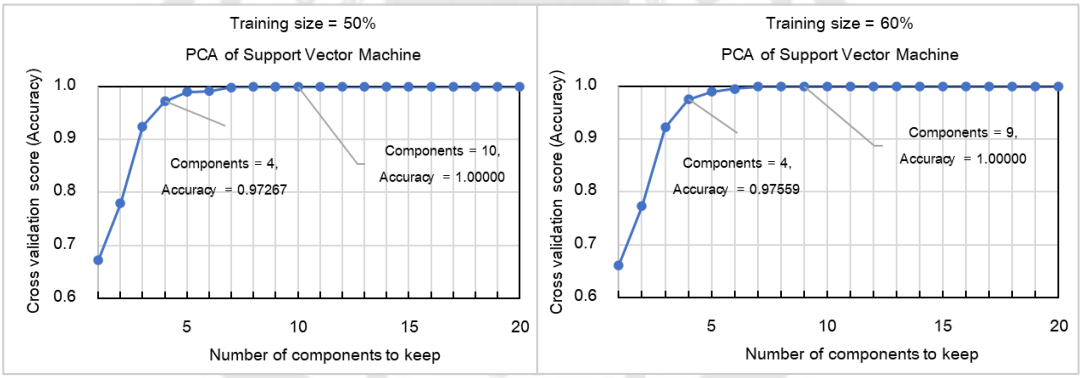


ภาพประกอบ 86 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Logistic regression ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

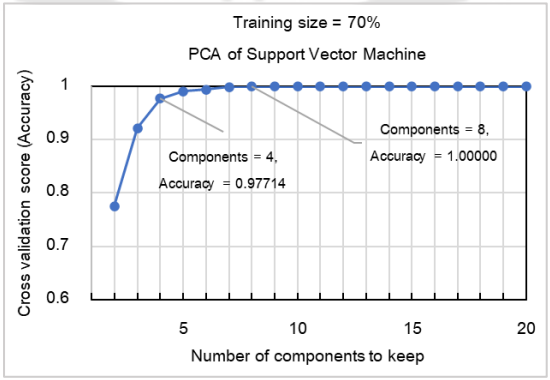
- Support Vector Machine จากภาพประกอบ 87 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวน components เพิ่มขึ้น โดยค่าเริ่มสูงมากกว่า 0.9000 ที่จำนวนฟีเจอร์เท่ากับ 3 และค่าเริ่มคงที่จำนวน components เท่ากับ 4 เป็นต้นไป และจำนวน components ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวนฟีเจอร์ 10, 9 และ 8 components ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นเท่ากับ 1.0000

สำหรับจุดข้อมูลที่จำนวน components เท่ากับ 1 ของ training size เท่ากับ 70% นั้น ไม่สามารถหาค่า cross validation score (accuracy) ได้ เนื่องจาก Google colab ใช้เวลาในการประมวลผลนานกว่า 10 ชั่วโมง แต่อย่างไรก็ตามแนวโน้มของค่า cross validation score (accuracy) นั้นคาดว่าเท่ากับ training size อื่นๆ และไม่ใช่จุดที่สนใจ



(ก) (ข)

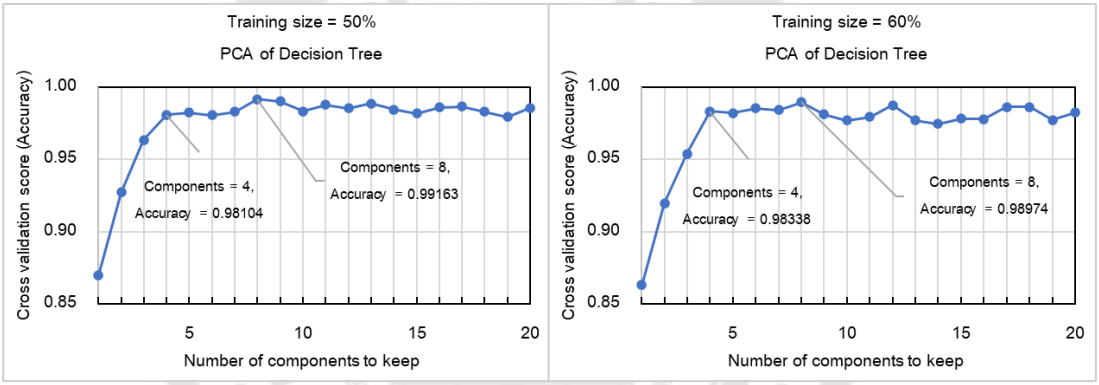


(ค)

ภาพประกอบ 87 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Support Vector Machine ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

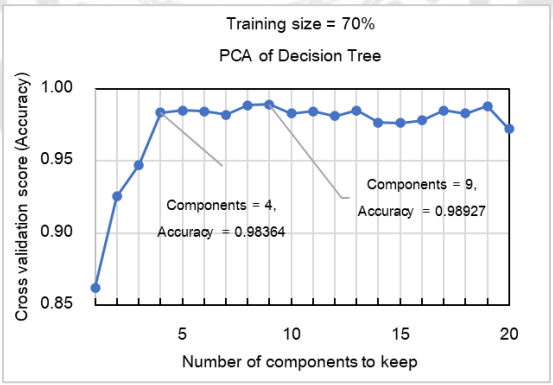
- Decision Tree จากภาพประกอบ 88 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวน components เพิ่มขึ้น โดยค่าเริ่มคงที่จำนวน components เท่ากับ 4 เป็นต้นไป และจำนวน components ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวน components เท่ากับ 8, 8 และ 9 components ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9916, 0.9897 และ 0.9892 ตามลำดับ



(ก)

(ข)

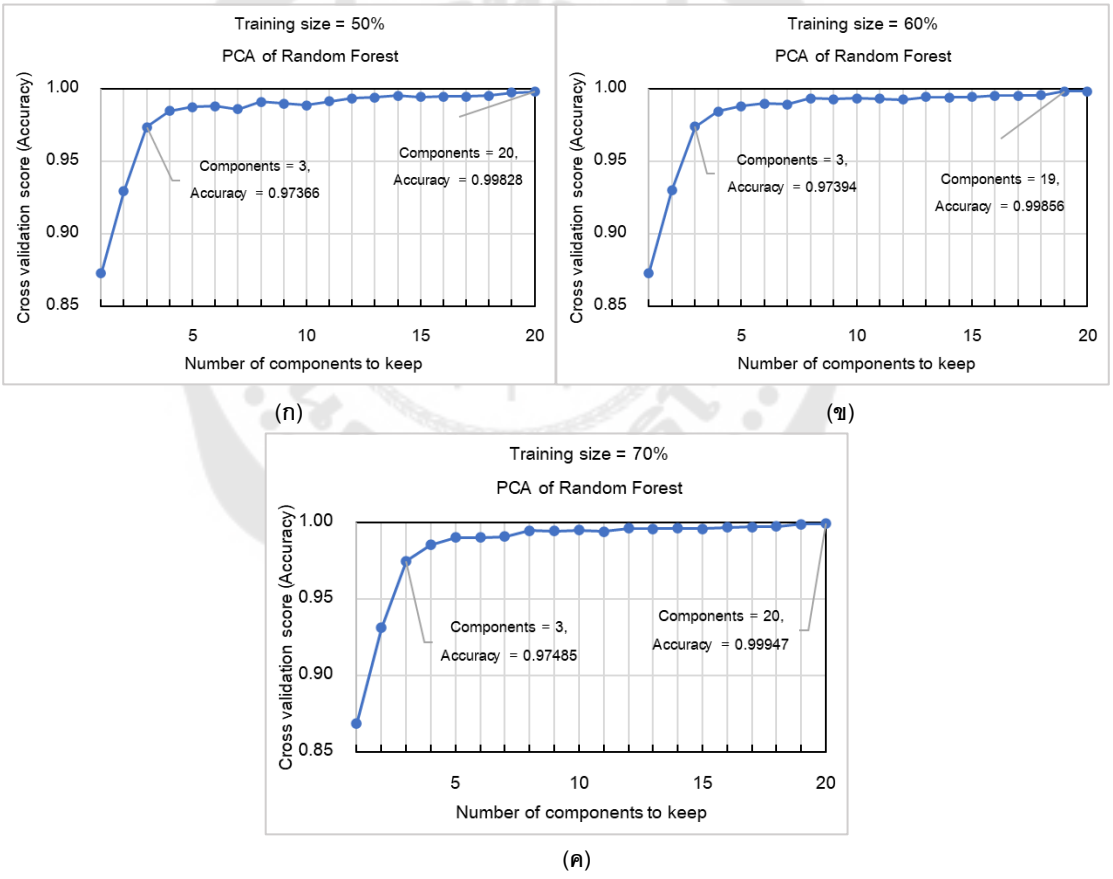


(ค)

ภาพประกอบ 88 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Decision Tree ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

- Random Forest จากภาพประกอบ 89 พบว่า การใช้ Training size ที่แตกต่างกัน จำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวน components เพิ่มขึ้น โดยค่าเริ่มคงที่จำนวน components เท่ากับ 3 เป็นต้นไป และจำนวน components ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวน components เท่ากับ 20, 19 และ 20 components ตามลำดับ

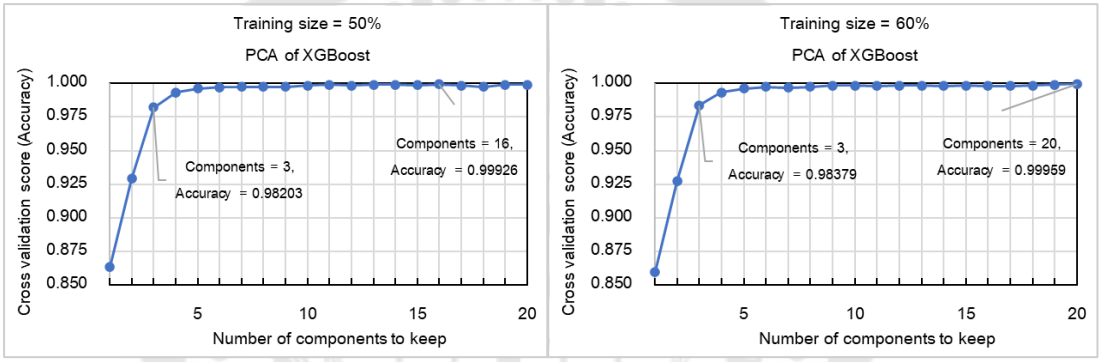
โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9982, 0.9985 และ 0.9994 ตามลำดับ



ภาพประกอบ 89 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ Random Forest ที่ training size ต่างๆ (ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

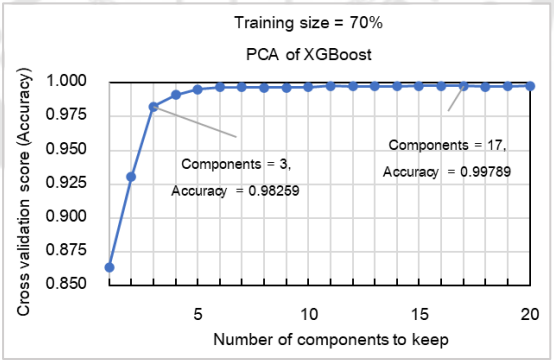
- XGBoost จากภาพประกอบ 90 พบว่า การใช้ Training size ที่แตกต่างกันจำนวน 3 อัตราส่วน พบว่าในทุกๆ training size ค่า cross validation score (accuracy) มีแนวโน้มสูงขึ้นเมื่อจำนวน components เพิ่มขึ้น โดยค่าเริ่มคงที่จำนวน components เท่ากับ 3 เป็นต้นไป และจำนวน components ที่ให้ค่า cross validation score (accuracy) ถึงจุดที่สูงที่สุดของ training size เท่ากับ 50%, 60% และ 70% คือที่จำนวน components เท่ากับ 16, 20 และ 17 components ตามลำดับ

โดยค่า cross validation score (accuracy) ที่สูงที่สุดของทั้ง 3 training size นั้นแตกต่างกันเพียงเล็กน้อย คือที่ training size เท่ากับ 50%, 60% และ 70% ให้ค่าประมาณ 0.9992, 0.9995 และ 0.9978 ตามลำดับ



(ก)

(ข)



(ค)

ภาพประกอบ 90 กราฟแสดง cross validation score (accuracy) เมื่อใช้จำนวน components ตั้งแต่ 1 - 20 ในการใช้วิธี PCA ร่วมกับ XGBoost ที่ training size ต่างๆ

(ก) training size = 50%, (ข) training size = 60% และ (ค) training size = 70%

จากนั้นจึงทำการเลือกจำนวน components ที่นำมาสร้างแบบจำลองร่วมกับการใช้เทคนิค PCA และทำการทดสอบวัดประสิทธิภาพกับ Test Set ซึ่งจากกราฟในภาพประกอบ 86 – 90 พบว่าในทุกๆ แบบจำลองนั้น จำนวน components ที่ทำให้ค่า cross validation score (accuracy) เริ่มคงที่เมื่อใช้จำนวน components ในช่วง 3 - 4 components ซึ่งหลังจากค่าเริ่มคงที่นั้น แม้ว่าจำนวน components มากขึ้น ค่า cross validation score (accuracy) นั้นก็ไม่ได้มากขึ้นอย่างเห็นได้ชัด และหากให้จำนวน components ที่มากเกินไปก็อาจทำให้แบบจำลองเกิดภาวะ overfitting ได้มาก ดังนั้นจึงกำหนดขอบเขตจำนวน components ที่นำไปศึกษาประสิทธิภาพต่อไป ในช่วง 3 – 9 components โดยผู้วิจัยได้เลือกศึกษาทั้งหมด 4 ค่า คือ 3, 5, 7 และ 9 ซึ่งผู้วิจัยได้หาองค์ประกอบของแต่ละ components (component loading) เมื่อกำหนดให้ใช้จำนวน components = 3, 5, 7 และ 9 ของการทดลองในแต่ละ training size โดยสามารถแสดงได้ ดังภาพประกอบที่ 91 – 93 โดยสามารถสรุปได้ว่า

การใช้จำนวน components = 3 พบว่าทั้ง 3 training size มีความเหมือนกันคือ

- principal component ที่ 1: ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “bruises” รองลงมาจะเป็น “gill_size” กับฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบมากที่สุด คือ “ring_type”
- principal component ที่ 2: ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “bruises” กับฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบมากที่สุด คือ “stalk_shape”
- principal component ที่ 3: ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “gill_spacing” และ “stalk_shape”

การใช้จำนวน components = 5 พบว่าทั้ง 3 training size มีความเหมือนกันคือ

- principal component ที่ 1 ถึง 3 เหมือนกับการใช้จำนวน components = 3
- principal component ที่ 4: ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “cap_surface”
- principal component ที่ 5 : ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “cap_surface” กับฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบมากที่สุด คือ “cap_color”

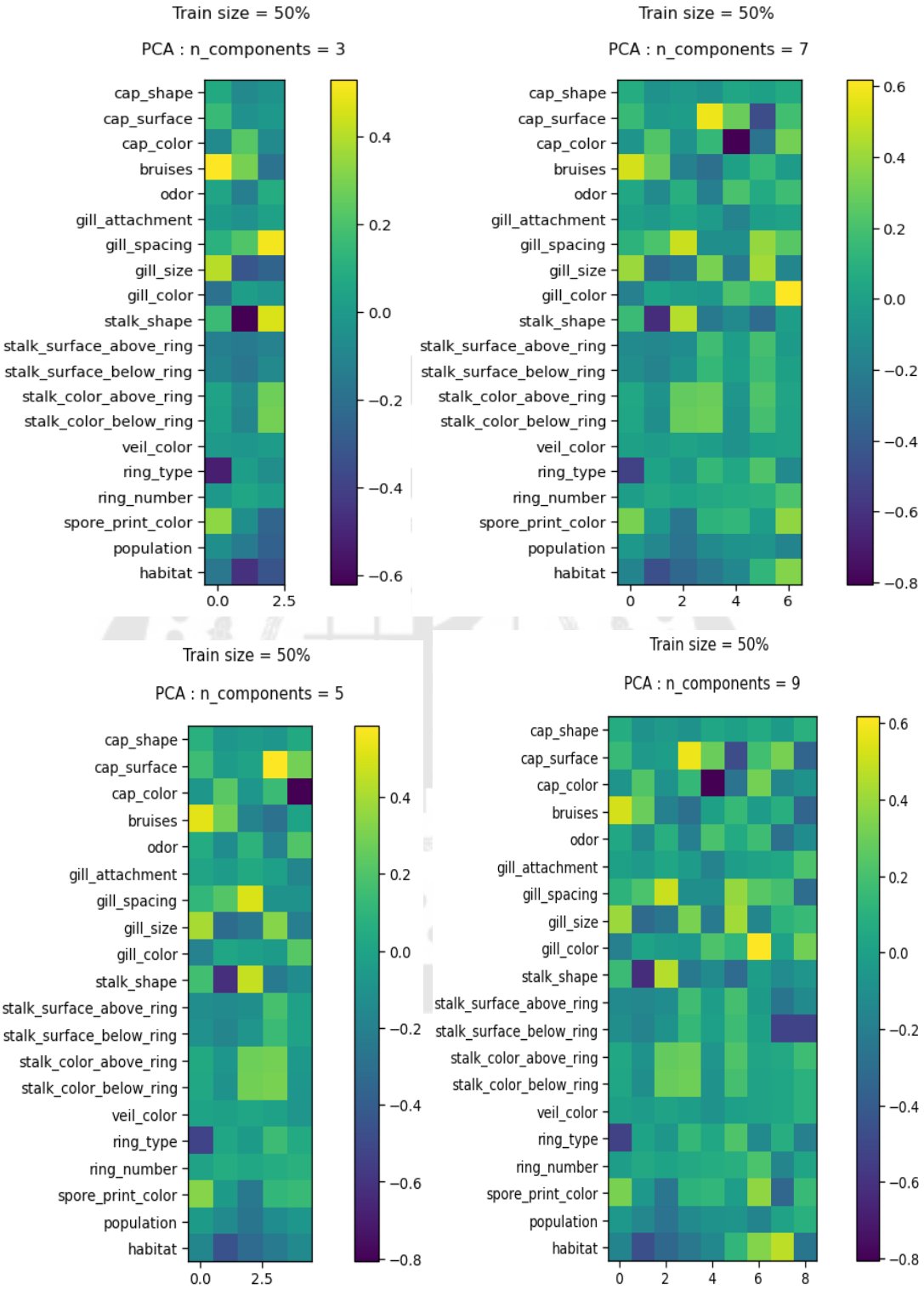
การใช้จำนวน components = 7 ของทั้ง 3 training size พบว่ามีความเหมือนกันคือ

- principal component ที่ 1 ถึง 3 เหมือนกับการใช้จำนวน components = 3
- principal component ที่ 4 ถึง 5 เหมือนกับการใช้จำนวน components = 5

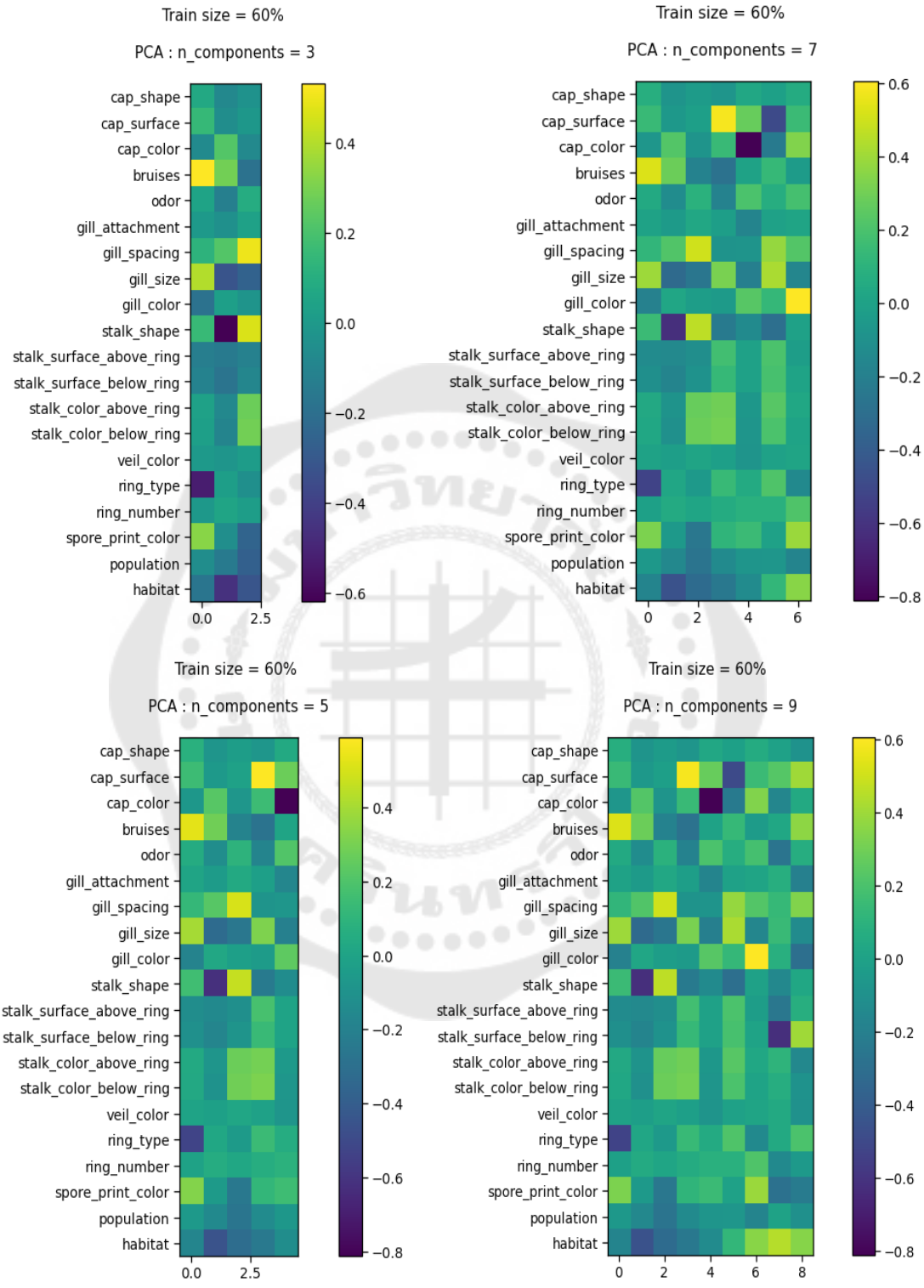
- principal component ที่ 6 : ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบมากที่สุด คือ “cap_surface”
- principal component ที่ 7 : ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “gill_color”

การใช้จำนวน components = 9 ของทั้ง 3 training size มีความคล้ายกันคือ

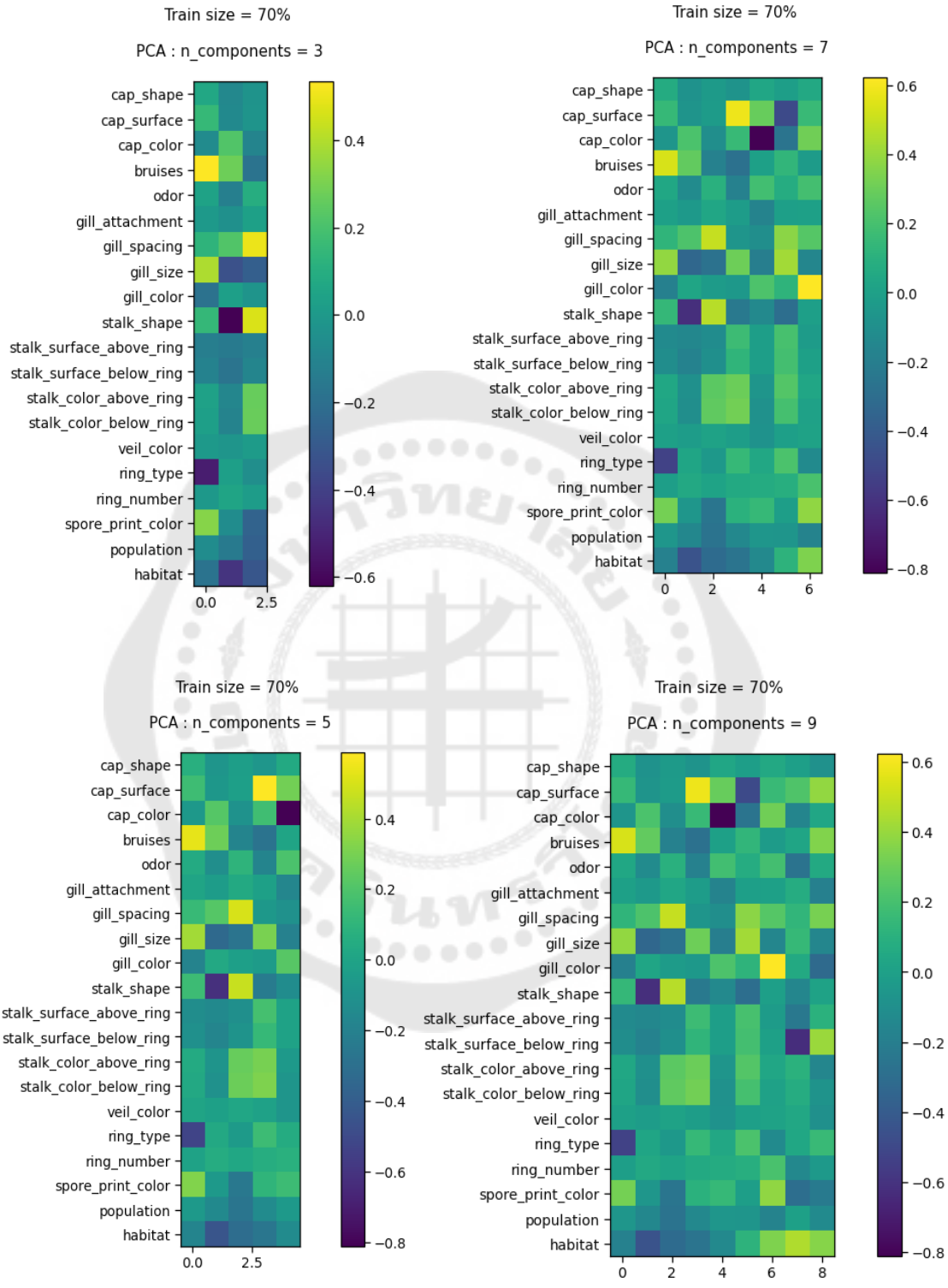
- principal component ที่ 1 ถึง 3 เหมือนกับการใช้จำนวน components = 3
- principal component ที่ 4 ถึง 5 เหมือนกับการใช้จำนวน components = 5
- principal component ที่ 6 ถึง 7 เหมือนกับการใช้จำนวน components = 7
- principal component ที่ 8 : ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวกมากที่สุด คือ “habitat” กับฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบมากที่สุด คือ “stalk_surface_below_ring”
- principal component ที่ 9 : สำหรับที่ Training size = 50% ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงลบ ได้แก่ “stalk_surface_below_ring”, “cap_surface”, “bruises” ส่วนที่ Training size = 60% และ 70% นั้นจะแตกต่างออกไปคือ ฟีเจอร์ที่มีความสำคัญโดยให้ผลเชิงบวก ได้แก่ “stalk_surface_below_ring”, “cap_surface”, “bruises”, “habitat”



ภาพประกอบ 91 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 50%



ภาพประกอบ 92 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 60%



ภาพประกอบ 93 Component loading ของการใช้ n_components เท่ากับ 3, 5, 7, 9 โดยใช้ training size เท่ากับ 70%

หลังจากนั้นนำแต่ละจำนวน components ไปใช้ในการสร้างแบบจำลองทั้ง 5 แบบจำลอง ร่วมกับการใช้เทคนิค PCA แล้วทำการตรวจสอบประสิทธิภาพ โดยผลแสดงได้ดังตาราง 7

สำหรับแบบจำลอง Logistic Regression พบว่าเมื่อมีการใช้จำนวน components มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น และพบว่า training size ที่แตกต่างกัน มีผลต่อประสิทธิภาพน้อย โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวน components เท่ากับ 9 และ training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 90.35% และ 90.33% ตามลำดับ

สำหรับแบบจำลอง Support Vector Machine พบว่าเมื่อมีการใช้จำนวน components มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น และพบว่า training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวน components เท่ากับ 9 โดยให้ค่า accuracy และ F1 score เท่ากับ 100%

สำหรับแบบจำลอง Decision Tree เมื่อมีการใช้จำนวน components มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวน components ตั้งแต่ 5 components ขึ้นไป แต่การใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวน components เท่ากับ 9 และ training size เท่ากับ 60% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.97%

สำหรับแบบจำลอง Random Forest เมื่อมีการใช้จำนวน components มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น แต่การใช้ training size ที่แตกต่างกัน มีผลต่อประสิทธิภาพในแต่ละ components ที่ไม่เหมือนกัน กล่าวคือ ที่ components เท่ากับ 3 พบว่า ที่ training size = 70% ให้ประสิทธิภาพที่ดีที่สุด รองลงมาคือที่ training size = 50% และสุดท้ายคือที่ training size = 60% ส่วนที่ components เท่ากับ 5, 9 พบว่า ที่ training size = 50% ให้ประสิทธิภาพที่ดีที่สุด รองลงมาคือ ที่ training size = 70% และสุดท้ายคือที่ training size = 60% ส่วนที่ components เท่ากับ 7 พบว่า ที่ training size = 60% ให้ประสิทธิภาพที่ดีที่สุด รองลงมาคือ ที่ training size = 70% และสุดท้ายคือที่ training size = 50% โดยการทดลองที่ให้ประสิทธิภาพที่ดีที่สุด คือ ที่จำนวน components เท่ากับ 9 และ training size เท่ากับ 50% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.29%

สำหรับแบบจำลอง XGBoost เมื่อมีการใช้จำนวน components มากขึ้น พบว่ามีประสิทธิภาพมากขึ้น โดยประสิทธิภาพเริ่มคงที่ที่จำนวนฟีเจอร์ตั้งแต่ 5 ฟีเจอร์ขึ้นไป แต่การใช้ training size ที่แตกต่างกัน ไม่มีผลต่อประสิทธิภาพมากนัก โดยการทดลองที่ให้ประสิทธิภาพที่ดี

ที่สุด คือ ที่จำนวนฟีเจอร์เท่ากับ 9 และ training size เท่ากับ 60% โดยให้ค่า accuracy และ F1 score เท่ากับ 99.78%

ตาราง 7 ประสิทธิภาพของแบบจำลอง โดยใช้เทคนิคการสกัดฟีเจอร์ด้วยวิธี PCA

Algorithm	No. of components	Training size	Accuracy (%)	F1 score (%)	AUC score (%)
Logistic Regression	3	50%	85.25	85.25	89.38
		60%	85.17	85.17	89.10
		70%	84.91	84.90	88.83
	5	50%	88.53	88.52	90.40
		60%	88.68	88.67	90.06
		70%	88.06	88.06	89.64
	7	50%	88.92	88.90	93.38
		60%	88.52	88.50	93.39
		70%	88.06	88.04	93.28
	9	50%	90.35	90.33	94.90
		60%	90.09	90.07	95.02
		70%	90.32	90.30	95.17
Support Vector Machine	3	50%	91.48	91.48	96.90
		60%	91.42	91.41	96.86
		70%	90.73	90.73	96.76
	5	50%	98.52	98.52	99.53
		60%	98.62	98.62	99.51
		70%	98.48	98.48	99.36
	7	50%	99.95	99.95	100
		60%	99.88	99.88	100
		70%	99.88	99.88	100
	9	50%	100	100	100
		60%	100	100	100
		70%	100	100	100

ตาราง 7 (ต่อ)

Algorithm	No. of components	Training size	Accuracy (%)	F1 score (%)	AUC score (%)
Decision Tree	3	50%	98.70	98.69	99.87
		60%	98.71	98.71	99.91
		70%	98.56	98.56	99.83
	5	50%	99.41	99.41	99.99
		60%	99.63	99.63	99.98
		70%	99.63	99.63	99.99
	7	50%	99.73	99.73	100
		60%	99.72	99.72	100
		70%	99.75	99.75	100
	9	50%	99.88	99.88	100
		60%	99.97	99.97	100
		70%	99.92	99.92	100
Random Forest	3	50%	96.63	96.63	98.56
		60%	95.85	95.85	98.77
		70%	97.33	97.33	99.02
	5	50%	98.92	98.92	99.28
		60%	97.88	97.88	98.71
		70%	98.32	98.32	99.12
	7	50%	97.34	97.34	98.67
		60%	98.28	98.28	99.29
		70%	98.24	98.24	99.52
	9	50%	99.29	99.29	99.58
		60%	97.63	97.63	98.61
		70%	98.48	98.48	99.11

ตาราง 7 (ต่อ)

Algorithm	No. of components	Training size	Accuracy (%)	F1 score (%)	AUC score (%)
XGBoost	3	50%	97.32	97.32	99.64
		60%	97.45	97.45	99.70
		70%	97.62	97.62	99.69
	5	50%	98.79	98.79	99.92
		60%	98.92	98.92	99.94
		70%	99.22	99.22	99.94
	7	50%	98.87	98.87	99.93
		60%	99.26	99.26	99.96
		70%	99.22	99.22	99.97
	9	50%	99.56	99.56	99.99
		60%	99.78	99.78	100
		70%	99.67	99.67	100

จากผลการศึกษาทั้งหมดพบว่าเมื่อพิจารณาจากทั้งประสิทธิภาพของแบบจำลอง, เทคนิคการคัดเลือก/การสกัดฟีเจอร์, training size ที่ใช้ และจำนวนฟีเจอร์ พบว่า แบบจำลอง Decision Tree ที่ใช้เทคนิค RFE โดยใช้ training size เท่ากับ 60% และใช้จำนวนฟีเจอร์เท่ากับ 3 ฟีเจอร์ (“odor“, “gill_size“, “spore_print_color”) มีความเหมาะสมที่สุด (Accuracy และ F1 score = 99.32%)

เนื่องจาก การใช้ฟีเจอร์ทั้งหมด 20 ฟีเจอร์ นั้นเป็นจำนวนที่มากและหากนำไปใช้ในการจำแนกประเภทเห็ดโดยพัฒนาบนเว็บแอปพลิเคชัน นั้นค่อนข้างใช้งานได้ลำบาก เนื่องจากประชาชนอาจไม่ทราบถึงรายละเอียดของเห็ด ได้แก่ อาจไม่ทราบถึงข้อมูล population (ลักษณะการกระจายตัว / การเกาะกลุ่มกันของเห็ด) ซึ่งเห็ดชนิดเดียวกันอาจมีการกระจายตัวที่มากกว่า 1 แบบ หรือหากดูที่สีของหมวกเห็ด (cap_color) ก็อาจมีความแตกต่างกันของการมองสี

เมื่อพิจารณาเทคนิค PCA ที่เป็นการสกัดฟีเจอร์ ก็ยังจำเป็นที่ผู้ใช้งานต้องทราบถึงฟีเจอร์ทั้งหมดเพื่อนำมาสกัดเป็น components และการลดมิติข้อมูลของเทคนิค PCA นี้ไม่ได้ดู

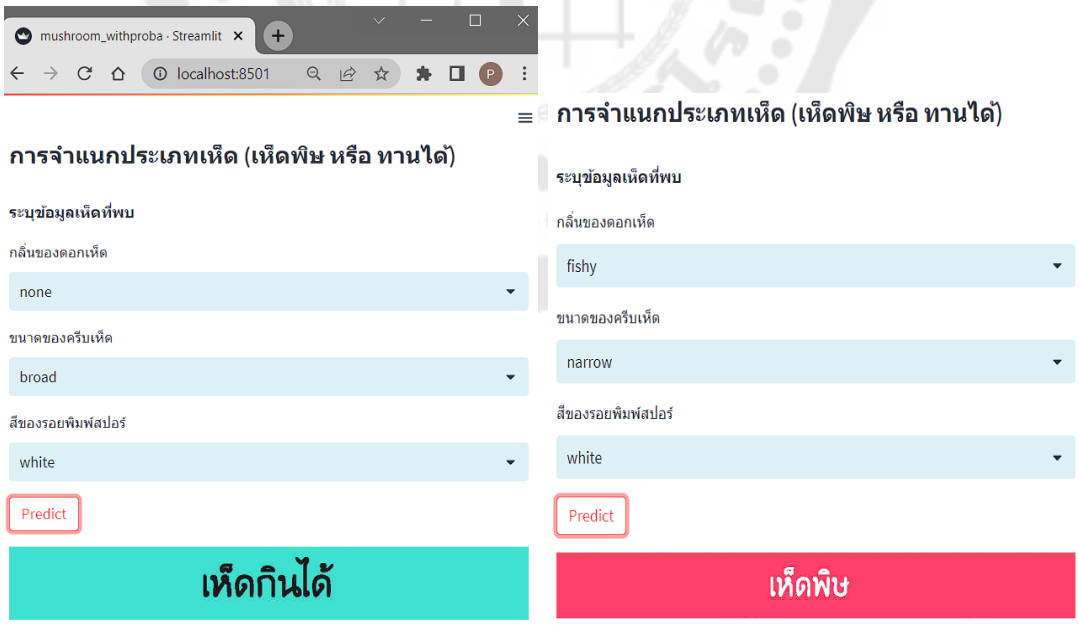
ความสัมพันธ์ของฟีเจอร์กับประเภทของเห็ด แต่ผลจากส่วนของ component loading นั้นสามารถสนับสนุนการใช้ฟีเจอร์ที่ชื่อ “gill_size” และ “spore_print_color” เล็กน้อย

และเมื่อพิจารณาถึงเทคนิค Chi-square นั้นพบว่าทุกๆแบบจำลอง และทุกๆ training size ที่ใช้จำนวนฟีเจอร์เท่ากับ 3 ประสิทธิภาพยังไม่ดีเท่ากับเทคนิค RFE

และเมื่อพิจารณาถึงแบบจำลองอื่นๆ ในเทคนิค RFE เดียวกันนี้ พบว่าแบบจำลอง Random Forest ที่ training size เท่ากับ 60% และใช้จำนวนฟีเจอร์เท่ากับ 3 ฟีเจอร์ ซึ่งใช้ฟีเจอร์เดียวกันกับ Decision Tree นั้น (“odor“, “gill_size“, “spore_print_color”) ก็ให้ประสิทธิภาพที่เท่ากัน แต่แบบจำลอง Decision Tree นั้นมีความซับซ้อนน้อยกว่า Random Forest

ดังนั้นผู้วิจัยจึงเลือกแบบจำลอง Decisoin Tree ที่ใช้เทคนิค RFE โดยใช้ training size เท่ากับ 60% และใช้จำนวนฟีเจอร์เท่ากับ 3 ฟีเจอร์ (“odor“, “gill_size“, “spore_print_color”) ไปพัฒนาเป็นต้นแบบ (prototype) บนเว็บแอปพลิเคชัน โดยใช้เครื่องมือที่ชื่อ Streamlit โดยสามารถแสดง ตามภาพประกอบ 94

แต่อย่างไรก็ตามในส่วนของฟีเจอร์ “spore_print_color” (สีของรอยพิมพ์สปอร์) ก็อาจมีความแตกต่างกันของการมองสี เช่น “Brown” กับ “Chocolate” จึงควรสอบถามผู้เชี่ยวชาญเพื่อสร้างไกด์ไลน์ของระดับสีที่แม่นยำ



(ก) (ข)

ภาพประกอบ 94 ตัวอย่างเว็บแอปพลิเคชันต้นแบบ

(ก) ตัวอย่างผลการทำนายเห็ดรับประทานได้ (ข) ตัวอย่างผลการทำนายเห็ดพิษ

บทที่ 5
สรุป อภิปรายผล และข้อเสนอแนะ

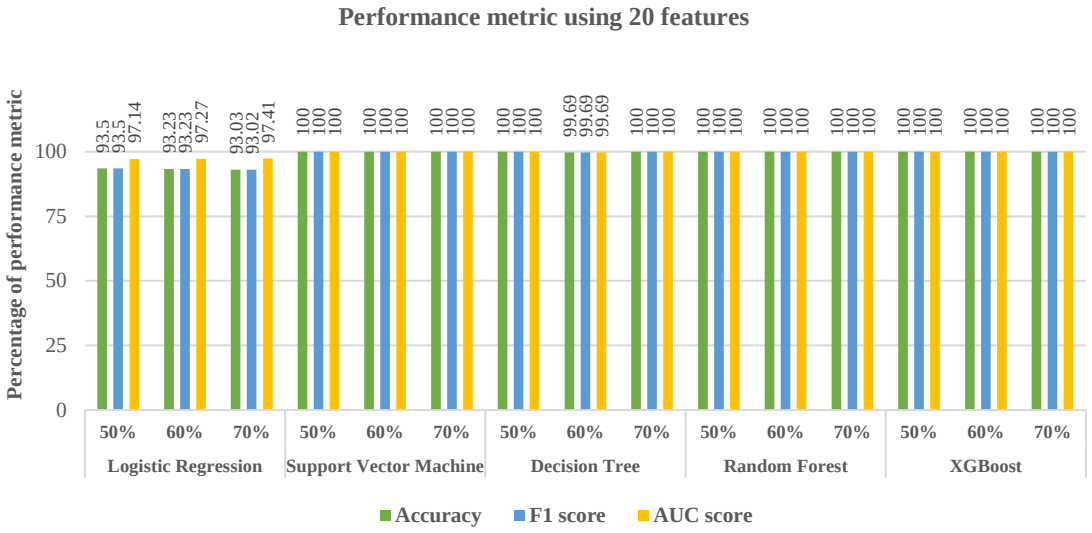
ในการวิจัยเรื่องการจำแนกประเภทของเห็ดระหว่างเห็ดมีพิษและไม่มีพิษโดยใช้เทคนิคการเรียนรู้ของเครื่อง ผู้วิจัยได้วัดประสิทธิภาพของแบบจำลองในแต่ละอัลกอริธึม และแต่ละเทคนิคการคัดเลือก/การสกัดฟีเจอร์ แล้วนำมาเปรียบเทียบและสรุปผล โดยสามารถแบ่งหัวข้อในการสรุปผลได้ดังต่อไปนี้

1. สรุปผลการวิจัย และอภิปรายผล
2. ข้อเสนอแนะ

1. สรุปผลการวิจัย และอภิปรายผล

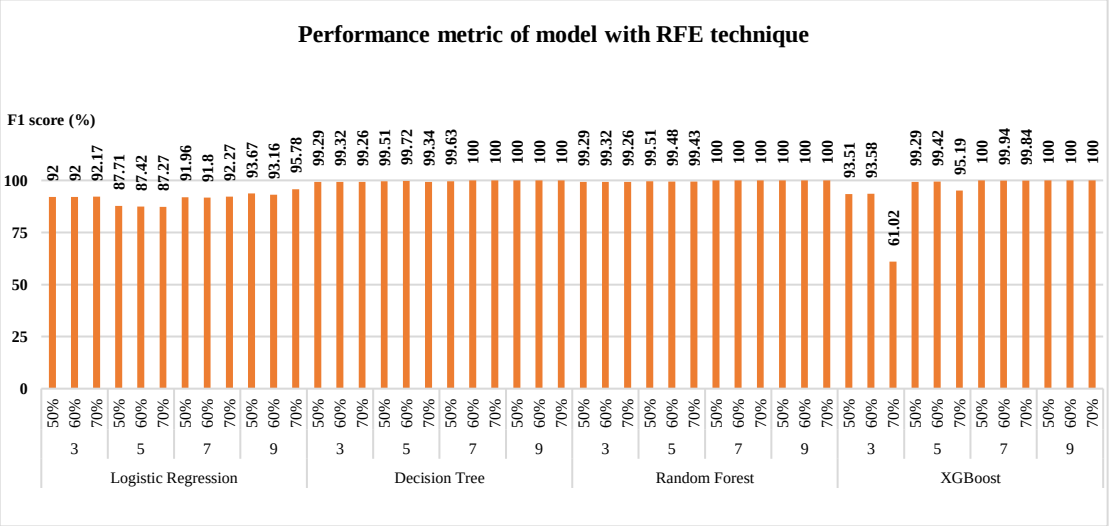
ในงานวิจัยนี้เกิดขึ้นจากปัญหาการจำแนกประเภทของเห็ดระหว่างเห็ดพิษและเห็ดรับประทานได้ ซึ่งในประเทศไทยได้พบผู้ป่วยที่เป็นโรคอาหารเป็นพิษ และผู้เสียชีวิตจากการรับประทานเห็ดพิษเป็นจำนวนหนึ่ง ผู้วิจัยจึงได้นำเทคนิคการเรียนรู้ของเครื่องมาใช้ในการจำแนกประเภทของเห็ดจากคุณลักษณะของเห็ด (ฟีเจอร์) ซึ่งได้ใช้ข้อมูลจากแหล่งข้อมูลสาธารณะ

ผลลัพธ์จากการสร้างแบบจำลองจำแนกประเภทเห็ดระหว่างเห็ดมีพิษและไม่มีพิษ โดยใช้ฟีเจอร์จำนวนทั้งหมด 20 ฟีเจอร์พบว่าแบบจำลอง Logistic Regression ให้ประสิทธิภาพที่ต่ำที่สุด เนื่องจาก Logistic regression เป็น linear model ซึ่งมีความสามารถในการประมวลผลข้อมูลที่มีความซับซ้อนมากๆ ได้ไม่ดีเมื่อเทียบกับแบบจำลองอื่นๆในงานวิจัยนี้ ส่วนแบบจำลอง Decision Tree ให้ประสิทธิภาพดี ค่า F1 score และ accuracy ส่วนใหญ่เท่ากับ 100% ยกเว้นที่ใช้ Training size = 60% ที่ให้ค่า F1 score, accuracy และ AUC score เท่ากับ 99.69% ซึ่งเนื่องจาก hyperparameter ที่แบบจำลอง Decision Tree นั้นปรับค่าได้นั้นมีความแตกต่างกันใน training size = 50% และ 60%, 70% และอาจเกิดจากความแตกต่างของการแบ่งชุดข้อมูลที่เกิดขึ้นในแต่ละ training size ทำให้แบบจำลอง Decision Tree ที่ใช้ training size ที่แตกต่างกันเรียนรู้ได้ต่างกัน ส่วนแบบจำลองอีก 3 แบบจำลอง คือ Support vector Machine, Random Forest และ XGBoost ให้ประสิทธิภาพดี คือให้ ค่า F1 score, accuracy และ AUC score เท่ากับ 100% ดังแสดงในกราฟดังภาพประกอบ 95



ภาพประกอบ 95 กราฟแสดงประสิทธิภาพของแต่ละแบบจำลอง โดยใช้ฟีเจอร์ทั้งหมด 20 ฟีเจอร์

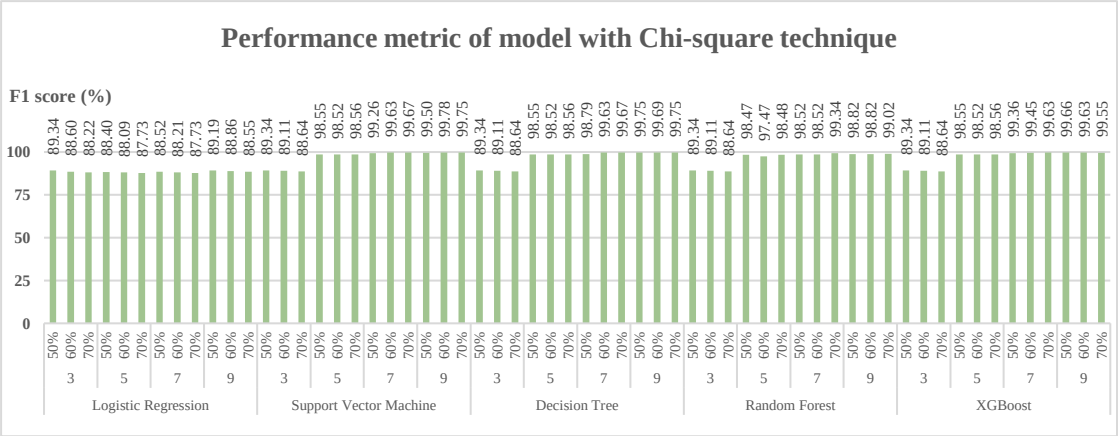
สำหรับผลลัพธ์การสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Recursive Feature Elimination (RFE) นั้น ได้สรุปประสิทธิภาพที่ได้เป็นค่า F1 score เนื่องจากมีค่าใกล้เคียงกับ Accuracy และการคำนวณหาค่า F1 score มีการนำ Precision และ Recall มาใช้ในสูตรคำนวณ ซึ่งจากกราฟตามภาพประกอบ 96 สามารถแสดงผลได้ว่า ในภาพรวมแบบจำลอง Logistic Regression ให้ประสิทธิภาพที่ค่อนข้างน้อยกว่าแบบจำลองอื่นๆ เนื่องจาก Logistic regression เป็น linear model ซึ่งมีความสามารถในการประมวลผลข้อมูลที่มีความซับซ้อนมากๆ ได้ไม่ดีเมื่อเทียบกับแบบจำลองอื่นๆในงานวิจัยนี้ และในแบบจำลอง Logistic Regression ที่ใช้จำนวนฟีเจอร์เท่ากับ 5 ก็ยังให้ค่าประสิทธิภาพน้อยกว่าการใช้จำนวนฟีเจอร์เท่ากับ 3, 7 และ 9 ซึ่งมีความสอดคล้องกับกราฟแสดง cross validation score (accuracy) ตามที่ได้แสดงไว้ก่อนหน้า (ภาพประกอบ 76) แต่ในส่วนของแบบจำลอง XGBoost พบว่าที่ training size เท่ากับ 70% และใช้จำนวนฟีเจอร์เท่ากับ 3 ให้ประสิทธิภาพที่ต่ำที่สุดในการศึกษาทั้งหมด ซึ่งพบว่าฟีเจอร์ที่ได้นั้นค่อนข้างแตกต่างจากแบบจำลองอื่นที่ใช้จำนวนฟีเจอร์เท่ากัน (เท่ากับ 3) คือ ไม่พบว่าเลือกฟีเจอร์ชื่อ “odor” หรือ “spore_print_color” มาใช้ในการประมวลผล และอาจเกิดจากความแตกต่างของการแบ่งชุดข้อมูลที่เกิดขึ้นในแต่ละ training size



ภาพประกอบ 96 ประสิทธิภาพของแต่ละแบบจำลองโดยคัดเลือกฟีเจอร์ด้วยวิธี RFE

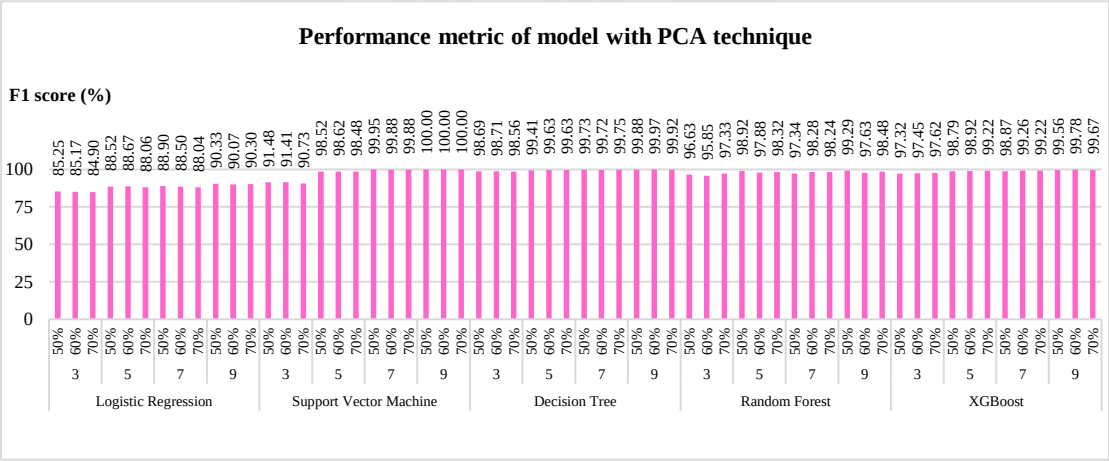
สำหรับผลลัพธ์การสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Chi-square นั้น ได้สรุปประสิทธิภาพที่ได้เป็นค่า F1 score เช่นเดียวกับในเทคนิค RFE ซึ่งจากกราฟตามภาพประกอบ 97 สามารถแสดงผลได้ว่า แบบจำลอง Logistic Regression ในทุกๆ training size และจำนวนฟีเจอร์ต่างๆ ให้ประสิทธิภาพที่ใกล้เคียงกันและค่อนข้างต่ำ เนื่องจาก Logistic regression เป็น linear model ซึ่งมีความสามารถในการประมวลผลข้อมูลที่มีความซับซ้อนมากๆ ได้ไม่ดีเมื่อเทียบกับแบบจำลองอื่นๆในงานวิจัยนี้ ส่วนแบบจำลองอื่นๆ พบว่าที่จำนวนฟีเจอร์เท่ากับ 3 ในทุกๆ training size นั้นให้ประสิทธิภาพที่ใกล้เคียงกัน และน้อยกว่าที่จำนวนฟีเจอร์อื่นๆ

นอกจากนี้เมื่อเปรียบเทียบผลลัพธ์ระหว่างการสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากการคัดเลือกด้วยวิธี Chi-square กับ วิธี RFE นั้นพบว่าที่จำนวนฟีเจอร์น้อยๆ การใช้เทคนิค RFE มีประสิทธิภาพของแบบจำลองที่ดีกว่า เนื่องจาก RFE เป็นเทคนิคคัดเลือกฟีเจอร์ ที่คำนวณจากการเรียนรู้ในการสร้างแบบจำลอง ในขณะที่ Chi-square เป็นเทคนิคที่คัดเลือกฟีเจอร์ก่อนเข้าสู่แบบจำลอง



ภาพประกอบ 97 ประสิทธิภาพของแต่ละแบบจำลองโดยคัดเลือกฟีเจอร์ด้วยวิธี Chi-square

สำหรับผลลัพธ์การสร้างแบบจำลองโดยใช้ฟีเจอร์ที่ได้จากวิธี PCA นั้น ได้สรุปประสิทธิภาพที่ได้เป็นค่า F1 score เช่นเดียวกับในเทคนิคก่อนหน้า ซึ่งจากกราฟตามภาพประกอบ 98 สามารถแสดงผลได้ว่า ในภาพรวมนั้นแบบจำลอง Logistic Regression ให้ประสิทธิภาพที่ค่อนข้างน้อยกว่าแบบจำลองอื่นๆ เนื่องจาก Logistic regression เป็น linear model ซึ่งมีความสามารถในการประมวลผลข้อมูลที่มีความซับซ้อนมากๆ ได้ไม่ดีเมื่อเทียบกับแบบจำลองอื่นๆในงานวิจัยนี้ และแบบจำลอง Support Vector Machine ที่ใช้จำนวน components เท่ากับ 9 ให้ค่า F1 score ที่มากที่สุด คือ 100% เนื่องจาก SVM ที่มี kernel เป็น Polynomial สามารถประมวลผลในการใช้ฟีเจอร์ที่มีมิติสูงๆ ได้ดี



ภาพประกอบ 98 ประสิทธิภาพของแต่ละแบบจำลองโดยการสกัดฟีเจอร์ด้วยวิธี PCA

โดยสามารถกล่าวโดยสรุปได้ว่า

การคัดเลือกฟีเจอร์ด้วยเทคนิค RFE ใช้จำนวนฟีเจอร์เพียง 3 ฟีเจอร์ ก็ให้ค่า F1 score ประมาณ 99 % ในขณะที่เทคนิค chi-square ใช้จำนวนฟีเจอร์ถึง 5 ฟีเจอร์ ให้ค่า F1 score ประมาณ 98 - 99 % สามารถกล่าวได้ว่า การคัดเลือกฟีเจอร์ด้วยเทคนิค RFE สามารถค้นหาฟีเจอร์ที่สำคัญออกมาโดยใช้จำนวนฟีเจอร์น้อยกว่าเทคนิค chi-square และมีประสิทธิภาพที่ใกล้เคียงกัน

และเมื่อใช้เทคนิคการคัดเลือกฟีเจอร์แล้ว แบบจำลอง Decision tree และ Random Forest มีประสิทธิภาพในการจำแนกประเภทเห็ดมากที่สุด โดยเท่ากัน ที่ training size = 60% โดยใช้จำนวนฟีเจอร์น้อยที่สุด เพียง 3 ฟีเจอร์ โดยใช้เทคนิค RFE

จากผลการวัดประสิทธิภาพของแบบจำลองร่วมกับการใช้เทคนิคการคัดเลือกฟีเจอร์ ส่วนใหญ่ให้ F1 score ประมาณ 99 % ขึ้นไป สามารถสรุปได้ว่า เทคนิคการเรียนรู้ของเครื่องสามารถจำแนกประเภทของเห็ด ได้เทียบเท่ามนุษย์เป็นอย่างน้อย

2. ข้อเสนอแนะ

งานวิจัยนี้สามารถนำไปพัฒนา ปรับปรุงและต่อยอด ได้ดังต่อไปนี้

2.1 เนื่องจากการวิจัยนี้ใช้ข้อมูลของเห็ดจากแหล่งข้อมูลดั้งเดิมคือ เป็นเห็ดที่มีการบันทึกในหนังสือ The Audubon Society Field Guide to North American Mushrooms (1981) ซึ่งเป็นเห็ดที่พบในฝั่งอเมริกาเหนือ ซึ่งอาจแตกต่างกับเห็ดภายในประเทศไทย ดังนั้นการศึกษาครั้งถัดไปจึงควรมีการเก็บข้อมูลของเห็ดที่พบภายในประเทศไทยเพิ่มเติม

2.2 การศึกษาครั้งถัดไปอาจนำรูปภาพมาใช้ในการประมวลผลภาพ (Image processing) เพื่อใช้ในการจำแนกประเภทเห็ด ร่วมกับการใช้แบบจำลองโครงข่ายประสาทเทียม (Neural Network) หรือเทคนิคการเรียนรู้เชิงลึก (Deep learning techniques)

2.3 การศึกษาครั้งถัดไปควรมีการแบ่งชุดข้อมูลโดยใช้เทคนิคอื่นๆ ที่ทำให้สามารถควบคุมสัดส่วนข้อมูลในแต่ละฟีเจอร์ให้เท่ากัน ในแต่ละ training size เพื่อสามารถตรวจสอบสมมติฐานว่าเมื่อ training size เพิ่มขึ้น แบบจำลองมีประสิทธิภาพดีขึ้น

2.4 การศึกษาครั้งถัดไปควรมีการศึกษาแบบจำลองอื่นๆ เพิ่มเติม ได้แก่ k-nearest neighbors, Naïve Bayes, VotingClassifier หรือ AutoSklearnClassifier (แบบจำลองที่มีการปรับจูนหาแบบจำลองและ hyperparameter ที่เหมาะสมอย่างอัตโนมัติ)

บรรณานุกรม

1. ธนพล นิ่มสมบูรณ์ และ นันทนา นิ่มสมบูรณ์. เห็ดพิษ. วารสารเภสัชกรรมโรงพยาบาล. 2564;31(2):73 - 81.
2. บารมี สกลรักษ์, กิตติมา ด้วงแค, จันจิรา อายะวงศ์, กฤษณา พงษ์พานิช และ วินันท์ดา หิมะมาน. เห็ดครีบ: กลุ่มป่าแก่งกระจาน เขตรักษาพันธุ์สัตว์ป่าเขียงดาว และเขตรักษาพันธุ์สัตว์ป่าภูเขียว. ส่วนวิจัยการอนุรักษ์ป่าไม้ สำนักวิจัยการอนุรักษ์ป่าไม้และพันธุ์พืช กรมอุทยานแห่งชาติ สัตว์ป่า และพันธุ์พืช; 2559.
3. . Edible mushroom identification using machine learning. 2022 International Conference on Computer Communication and Informatics (ICCCI); 2022.
4. นันทนา แต้ประเสริฐ. เห็ดพิษ. โรงพิมพ์โจเซฟ จ.นครราชสีมา: สำนักงานป้องกันควบคุมโรคที่ 5 นครราชสีมา กรมควบคุมโรค กระทรวงสาธารณสุข; 2552.
5. พรสุดา โสวรรณี, สิรินาถ เทียนคำ และ ณัฐพล ปัญญา. ความรู้ ความเชื่อ การปฏิบัติในการเลือกเห็ดเพื่อปรุงอาหารและการปฐมพยาบาลเบื้องต้น เมื่อป่วยด้วยโรคเห็ดพิษของเชียนเห็ด อำเภอตระการพืชผล จังหวัดอุบลราชธานี. วารสารวิชาการสาธารณสุข. 2563;29(6):995 - 1005.
6. อุทัยวรรณ แสงวณิช, พูนพิไล สุวรรณฤทธิ์, อัจฉรา พยัพพานนท์, เจนนิเฟอร์ เหลืองสอาด, อนงค์ จันทร์ศรีกุล และ บารมี สกลรักษ์. บัญชีรายการทรัพย์สินชีวภาพ เห็ด. 1. กรุงเทพฯ: สำนักงานพัฒนาเศรษฐกิจจากฐานชีวภาพ (องค์การมหาชน); 2556. 374.
7. ศูนย์พิษวิทยา สถาบันวิจัยวิทยาศาสตร์สาธารณสุข. เห็ดพิษที่พบบ่อยในประเทศไทย2560.
8. สำนักงานกองทุนสนับสนุนการสร้างเสริมสุขภาพ (สสส.). บริโภค"เห็ดพิษ"อันตรายถึงตาย [อินเทอร์เน็ต]. 2560 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://www.thaihealth.or.th/Content/36848-บริโภค“เห็ดพิษ”อันตรายถึงตาย.html
9. สำนักงานกองทุนสนับสนุนการสร้างเสริมสุขภาพ (สสส.). เตือนประชาชนเก็บหรือซื้อเห็ดป่า อาจเป็น"เห็ดพิษ" [อินเทอร์เน็ต]. 2560 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://www.thaihealth.or.th/Content/39244-เตือนประชาชนเก็บหรือซื้อเห็ดป่า%20อาจเป็น“เห็ดพิษ”.html
10. สำนักงานกองทุนสนับสนุนการสร้างเสริมสุขภาพ (สสส.). แพทย์เตือนระวังเห็ดพิษ ก่อนเก็บ-ซื้อมาปรุงอาหาร [อินเทอร์เน็ต]. 2561 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก:

https://www.thaihealth.or.th/Content/44371-แพทย์เตือนระวังเห็ดพิษ%20ก่อนเก็บ-ซื้อมาปรุงอาหาร.html

11. กรมควบคุมโรค กระทรวงสาธารณสุข. กินเห็ดพิษอันตรายถึงชีวิต [อินเทอร์เน็ต]. 2563 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://www.brh.go.th/index.php/2019-02-27-04-12-21/362-2020-10-05-00-56-27
12. กองระบาดวิทยา/กองโรคติดต่อทั่วไป กรมควบคุมโรค. กรมควบคุมโรค เตือนประชาชนในช่วงหน้าฝนนี้ ระมัดระวังการเก็บหรือซื้อเห็ดป่ามารับประทาน อาจเป็น ‘เห็ดพิษ’ เสี่ยงเสียชีวิตได้ [อินเทอร์เน็ต]. 2563 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://ddc.moph.go.th/brc/news.php?news=14527
13. สำนักงานกองทุนสนับสนุนการสร้างเสริมสุขภาพ (สสส.). ห่วงบริโภคเห็ดมีพิษ ช่วงหน้าฝน [อินเทอร์เน็ต]. 2563 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://www.thaihealth.or.th/ห่วงบริโภคเห็ดมีพิษ-ช่ว/
14. สำนักสื่อสารความเสี่ยงฯ กรมควบคุมโรค. กรมควบคุมโรค เผยแพร่พยากรณ์โรคฯ ฉบับที่ 23/2564 "เตือนประชาชนช่วงฤดูฝน จะมีเห็ดป่าขึ้นเองตามธรรมชาติจำนวนมาก หากไม่แน่ใจ ไม่รู้จัก หรือสงสัยว่าจะเป็นเห็ดพิษ ไม่ควรเก็บหรือซื้อมาปรุงอาหารรับประทาน" [อินเทอร์เน็ต]. 2564 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://ddc.moph.go.th/brc/news.php?news=19111&deptcode=brc
15. กองโรคติดต่อทั่วไป กรมควบคุมโรค. รอบรู้เรื่องโรคและภัยสุขภาพ เรื่อง สคร.6 ชลบุรี เตือนระวังการนำเห็ดป่ามาปรุงอาหาร อาจเป็น ‘เห็ดพิษ’ เสี่ยงเสียชีวิตได้ [อินเทอร์เน็ต]. 2565 [เข้าถึงเมื่อ 22 ก.ย. 2565]. เข้าถึงได้จาก: https://ddc.moph.go.th/odpc6/news.php?news=27313&deptcode=odpc6
16. UCI MACHINE LEARNING. Mushroom classification [Internet]. 2016 [cited 2022 Sep 22]. Availble from: https://www.kaggle.com/datasets/uciml/mushroom-classification
17. Schlimmer J. Mushroom data set [Internet]. Machine Learning Repository; 1987 [cited 2022 Sep 22]. Available from: https://archive.ics.uci.edu/ml/datasets/Mushroom
18. Dua D, Graff C. Uci machine learning repository [Internet]. University of California, Irvine, School of Information and Computer Sciences; 2017 [cited 2022 Sep 22]. Availble from: http://archive.ics.uci.edu/ml
19. Islam MD. Cultivation techniques of edible mushrooms: Agaricus bisporus, pleurotus

spp., lentinula edodes and volvariella volvocea. 2015.

20. Preechasuk J, Chaowalit O, Pensiri F, Visutsak P. Image analysis of mushroom types classification by convolution neural networks. Proceedings of the 2019 2nd Artificial Intelligence and Cloud Computing Conference; Kobe, Japan. Association for Computing Machinery; 2019. 82–8. https://doi.org/10.1145/3375959.3375982
21. Denchev C, Denchev T, Polemis E, Venturella G, Gargano M, Zervakis G. General aspects of mushroom fungi. [Internet]. 2013. 4–15. Availble from: https://www.researchgate.net/publication/281625151_General_Aspects_of_Mushroom_Fungi
22. บัญชา ปะสีละเตสัง. สร้างการเรียนรู้สำหรับ ai ด้วย python machine learning. กรุงเทพฯ: ซีเอ็ดยูเคชั่น; 2564.
23. Harrington P. Machine learning in action. New York: Manning Publications Co.; 2012.
24. Noparat N. มาทำความรู้จักกับ การเรียนรู้แบบเสริมกำลัง (reinforcement learning) [อินเทอร์เน็ต]. Big Data Thailand; 2563 [เข้าถึงเมื่อ 12 ต.ค. 2565]. เข้าถึงได้จาก: https://bigdata.go.th/big-data-101/introduction-to-reinforcement-learning/
25. Agarwal D. Introduction to svm(support vector machine) along with python code [Internet]. Data Science Blogathon; 2021 [cited 2022 Oct 12]. Availble from: https://www.analyticsvidhya.com/blog/2021/04/insight-into-svm-support-vector-machine-along-with-code/
26. Gupta P. Decision trees in machine learning [Internet]. Towards Data Science; 2017 [cited 2022 Nov 6]. Available from: https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052
27. Saini A. Decision tree algorithm – a complete guide. Data Science Blogathon; 2021 [cited 2022 Oct 12]. Availble from: https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/
28. What is a random forest? [Internet]. TIBCO Software Inc.; 2022 [cited 2022 Oct 12]. Availble from: https://www.tibco.com/reference-center/what-is-a-random-forest
29. mariajesusbigml. Introduction to boosted trees [Internet]. 2017. bigml; [cited 2022 Oct 12]. Available from: https://blog.bigml.com/2017/03/14/introduction-to-boosted-

trees/

30. xgboost developers. Introduction to boosted trees [Internet]. 2021 [cited 2022 Sep 22]. Available from: https://xgboost.readthedocs.io/en/stable/tutorials/model.html
31. scikit-learn developers. Cross-validation: Evaluating estimator performance [Internet]. scikit-learn 1.1.2; [cited 2022 Oct 12]. Availble from: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation
32. Brownlee J. Train-test split for evaluating machine learning algorithms [Internet]. Machine Learning Mastery; 2020 [cited 2022 Oct 12]. Available from: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/
33. Bronshtein A. Train/test split and cross validation in python [Internet]. Towards Data Science; 2017 [cited 2022 Oct 12]. Available from: https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6
34. Joseph VR. Optimal ratio for data splitting. Statistical Analysis and Data Mining: The ASA Data Science Journal. 2022;15(4):531-8.
35. Roy B. All about feature scaling [Internet]. Towards Data Science; 2020 [cited 2022 Oct 12]. Available from: https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35
36. Brownlee J. How to scale data with outliers for machine learning [Internet]. Machine Learning Mastery; 2020 [cited 2022 Oct 12]. Available from: https://machinelearningmastery.com/robust-scaler-transforms-for-machine-learning/
37. Marlaithong T, Thamjaroenporn P. เพิ่มประสิทธิภาพของ machine learning model ด้วย hyperparameter optimization [อินเทอร์เน็ต]. 2564 [สืบค้นเมื่อ 12 ต.ค. 2565]. สืบค้นจาก: https://bigdata.go.th/big-data-101/machine-learning-model-hyperparameter-optimization/
38. ES S, Bajaj A. Hyperparameter tuning in python: A complete guide [Internet]. 2022 [cited 2022 Oct 12]. Available from: https://neptune.ai/blog/hyperparameter-tuning-

in-python-complete-guide

39. Madaan M. Hyperparameter tuning: Beginners tutorial [Internet]. Info Edge (India) Ltd; 2022 [cited 2022 Oct 12]. Availble from: https://www.naukri.com/learning/articles/hyperparameter-tuning-beginners-tutorial/#hyper
40. Verma V. A comprehensive guide to feature selection using wrapper methods in python [Internet]. Data Science Blogathon; 2020 [cited 2022 Oct 12]. Available from: https://www.analyticsvidhya.com/blog/2020/10/a-comprehensive-guide-to-feature-selection-using-wrapper-methods-in-python/
41. Saeys Y, Inza I, Larrañaga P. A review of feature selection techniques in bioinformatics. Bioinformatics. 2007;23(19):2507-17. eng. 2007/08/28.
42. Hira ZM, Gillies DF. A review of feature selection and feature extraction methods applied on microarray data. Advances in Bioinformatics. 2015;2015:1-13.
43. Brownlee J. How to choose a feature selection method for machine learning [Internet]. Machine Learning Mastery; 2020 [cited 2022 Oct 12]. Available from: https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/
44. Alkronz ES, Moghayer KA, Meimeh M, Gazzaz M, Abu-Nasser BS, Abu-Naser SS. Prediction of whether mushroom is edible or poisonous using back-propagation neural network. International Journal of Academic and Applied Research (IJAAR). 2019;3(2):1-8.
45. Hamonangan R, Saputro MB, Bagus C, Dinata S, Atmaja K. Accuracy of classification poisonous or edible of mushroom using naïve bayes and k-nearest neighbors. Journal of Soft Computing Exploration (JOSCEX). 2021;2(1):53-60.
46. Tutuncu K, Cinar I, Kursun R, Koklu M. Edible and poisonous mushrooms classification by machine learning algorithms 2021 10th Mediterranean conference on embedded computing (MECO); Budva, Montenegro. IEEE; 2021.
47. Tarawneh O, Tarawneh M, Sharrab Y, Altarawneh M. Mushroom classification using machine-learning techniques. International Computer Sciences and Informatics

Conference (ICSIC-2022); Amman arab university, Jordan. 2022.

48. . Behavioural features for mushroom classification. 2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE); 2018.
49. Verma S, Dutta M. Mushroom classification using ann and anfis algorithm. IOSR Journal of Engineering (IOSRJEN). 2018;08(01):26-32.
50. Tank K. A comparative study on mushroom classification using supervised machine learning algorithms. International Journal of Trend in Scientific Research and Development (IJTSRD). 2021;5(5):716-23.
51. Vanitha V, Ahil MN, Rajathi N. Classification of mushrooms to detect their edibility based on key attributes. Bioscience Biotechnology Research Communications. 2020;13(11):37-41.
52. . Performance comparison of mushroom types classification using k-nearest neighbor method and decision tree method. 2020 3rd International Conference on Information and Communications Technology (ICOIACT); 2020: IEEE.
53. Agrawal SK. Metrics to evaluate your classification model to take the right decisions [Internet]. Analytics Vidhya; 2021 [cited 2022 Oct 12]. Available from: https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/
54. Narkhede S. Understanding auc - roc curve [Internet]. 2018 [cited 2022 Oct 12]. Available from: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
55. scikit-learn developers. Sklearn.Metrics.Roccurvedisplay [Internet]. scikit-learn 1.1.2; [cited 2022 Oct 12]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.RocCurveDisplay.html
56. Streamlit documentation [Internet]. Streamlit Inc.; 2023 [cited 2023 Feb 15]. Available from: https://docs.streamlit.io/

ประวัติผู้เขียน