การศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมอง

STUDY OF CLASSIFICATION TECHNIQUES USING MULTI-VIEW CLASSIFICATION

เปมิกา บุญเสริมส่ง

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2562

การศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมอง

เปมิกา บุญเสริมส่ง

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2562

STUDY OF CLASSIFICATION TECHNIQUES USING MULTI-VIEW CLASSIFICATION

PAYMIKA BOONSERMSONG

A Master's Project Submitted in partial Fulfillment of Requirements

for MASTER OF SCIENCE (Information Technology)

Faculty of Science Srinakharinwirot University

2019

สารนิพนธ์

เรื่อง

การศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมอง

ของ

เปมิกา บุญเสริมส่ง

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
ของมหาวิทยาลัยศรีนครินทรวิโรฒ

---------- คณบดีบัณฑิตวิทยาลัย

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณะกรรมการสอบปากเปล่าสารนิพนธ์

..... ที่ปรึกษาหลัก

(อาจารย์ ดร.ศิริสรรพ เหล่าหะเกียรติ)

..... ประธาน

()

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์)

..... กรรมการ

(อาจารย์ ดร.สุทธิพงศ์ ธัชยพงษ์)

ชื่อเรื่อง	การศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมอง
ผู้วิจัย	เปมิกา บุญเสริมส่ง
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2562
อาจารย์ที่ปรึกษา	อาจารย์ ดร. ศิริสรรพ เหล่าหะเกียรติ

การแยกแยะแบบหลายมุมมองใช้กับข้อมูลที่มีหลายมุมมองเช่นข้อมูลรูปภาพที่มีด้านหน้า และด้านข้าง โดยเราจะทำการจำแนกประเภทโดยรวมข้อมูลจากมุมมองต่างๆ เข้าด้วยกันทำให้ได้ประสิทธิภาพดีขึ้นเมื่อเปรียบเทียบกับการใช้ข้อมูลแบบมุมมองเดียวดังนั้นจึงมีนักวิจัยขยายแนวคิดการแยกแยะแบบหลายมุมมองมาใช้กับข้อมูลแบบมุมมองเดียว โดยแบ่งฟีเจอร์ออกเป็นหลายกลุ่มซึ่งแต่ละกลุ่มจะเป็นตัวแทนของหนึ่งมุมมอง อย่างไรก็ตามยังไม่มีใครเสนอวิธีการแยกแยะข้อมูลออกเป็นหลายมุมมองอย่างเป็นระบบ โดยนักวิจัยมักมีสมมติฐานว่าเรามีข้อมูลในหลายมุมมองอยู่แล้ว แต่ในการศึกษานี้ เรานำเสนอวิธีการที่เป็นระบบในการสร้างข้อมูลแบบหลายมุมมองขึ้นจากข้อมูลแบบมุมมองเดียว โดยใช้เทคนิคการเปลี่ยนแกนของข้อมูลได้แก่ PCA และ LDA การจำแนกแยกแยะแบบหลายมุมมองที่นำเสนอนี้ใช้เทคนิคการโหวตเสียงส่วนใหญ่ โดยการรวมผลการจำแนกแยกแยะจากตัวจำแนกหลายๆตัวที่ถูกเทรนด้วยข้อมูลจากมุมมองที่แตกต่างกัน ผลการทดลองแสดงให้เห็นว่าวิธีการนำเสนอนี้มีประสิทธิภาพมากกว่าวิธีจำแนกแยกแยะแบบมุมมองเดียว

คำสำคัญ : เทคนิคการเรียนรู้ของเครื่อง, การจำแนกแบบหลายมุมมอง, ต้นไม้ตัดสินใจ, พีซีเอ, แอลดีเอ

Title	STUDY OF CLASSIFICATION TECHNIQUES USING MULTI-VIEW CLASSIFICATION
Author	PAYMIKA BOONSERMSONG
Degree	MASTER OF SCIENCE
Academic Year	2019
Thesis Advisor	Dr. Sirisup Laohakiat

Multiview classification is a technique used with data sets with different views, for example image data sets taken from front and side views. This type of classification is performed collaboratively using data from different views leading to a better performance compared with using data from a single view. Recently, researchers have extended the idea to address ordinary data sets with single view by dividing their features to several sets of features, each of which is considered as one view. However, there is no systematic way in dividing whole features into different views. Many researchers assume that multiple view data are provided in advance. In this study, a systematic approach was proposed to generate multiview data sets from a single view data set using coordinate transformation algorithms including PCA and LDA. The proposed Multiview classification uses a majority vote to combine classification results from classifiers trained using data from different views. The experimental results showed that the proposed method outperformed the single-view classification method.

Keyword : Machine learning, Multiview Classification, Decision Tree, PCA, LDA

กิตติกรรมประกาศ

สารนิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จากดร.ศิริสรรพ เหล่าหะเกียรติ อาจารย์ที่ปรึกษา ที่ให้คำปรึกษา คำแนะนำในการจัดทำสารนิพนธ์ และสนับสนุนข้อมูลในการทำสารนิพนธ์ฉบับนี้

ขอขอบพระคุณคณะกรรมการสอบสารนิพนธ์ ที่ได้ให้คำแนะนำและข้อเสนอแนะปรับปรุงสารนิพนธ์ฉบับนี้

ขอกราบขอบพระคุณ ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ สำหรับสถานที่ในการทำวิจัยครั้งนี้

เปมิกา บุญเสริมส่ง

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ในปัจจุบันการนำเทคนิคการเรียนรู้ของเครื่องมาใช้ในการวิเคราะห์ข้อมูลนั้นมีกันอย่างแพร่หลาย แต่นักวิจัยส่วนมากจะใช้วิธีการวิเคราะห์ข้อมูลแบบมุมมองเดียว ซึ่งมีหลายวิธีด้วยกัน เช่น เทคนิคต้นไม้ตัดสินใจ (Decision Tree) (กุล ลินิน อนันต. 2014) เทคนิคการแบ่งกลุ่มข้อมูลแบบเคมีน (K-Means) (พรหม สุ พจน์ เฮง พระ.) เทคนิคการวิเคราะห์องค์ประกอบหลัก (Principal Component Analysis: PCA) (วิทวัส วิทยา ไกร เลิศ; พิลึก สุเมธ. 2559; เมืองแก้ว ธีร ยุทธ์. 2012; พุ่มลำเจียก ธนพล. 2559) เทคนิคการวิเคราะห์จำแนกประเภทเชิงเส้น (Linear Discriminant Analysis: LDA) (พุ่มลำเจียก ธนพล. 2559; Kumar Rajiv; และคนอื่น ๆ. 2012) เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-nearest neighbor: K-NN) (สินสมบูรณ์ทอง สุรวัชร ศรีเปารยะ และสายชล. 2560; ทองคำ ภูริพัทร์. 2559) วิธีการถดถอยลอจิสติก (Logistic Regression) (สินสมบูรณ์ทอง สุรวัชร ศรีเปารยะ และสายชล. 2560) เป็นต้น ถึงแม้เทคนิคข้างต้นที่ได้กล่าวมานั้นมีความแม่นยำในระดับหนึ่ง แต่ทว่า ในช่วงหลังมานี้ ได้มีการนำเทคนิคการวิเคราะห์แบบหลายมุมมอง (Zhao Jing; และคนอื่น ๆ. 2017) มาช่วยในการแยกแยะข้อมูลเพื่อเพิ่มประสิทธิภาพของการทำงาน

ในบทความนี้เรานำเสนอแนวความคิดการสร้างข้อมูลแบบหลายมุมมองขึ้นจากข้อมูลแบบมุมมองเดียว ซึ่งเป็นข้อมูลที่ใช้กันในปกติ โดยใช้เทคนิคของการเรียนรู้แบบกลุ่ม (ปรเมษฐ์ ธันวานนท์ ชัยกร ยิ่งเสร, และ วรพล พงษ์เพ็ชร.) มาช่วยเพิ่มประสิทธิภาพ โดยขั้นแรก เราจะสร้างข้อมูลในมุมมองที่แตกต่างกัน จากข้อมูลมุมมองเดียว โดยใช้เทคนิคการบิดแกนข้อมูลชนิดต่างๆ ได้แก่ เทคนิคการวิเคราะห์องค์ประกอบหลัก (Principal Component Analysis: PCA) (เมืองแก้ว ธีร ยุทธ์. 2012; พิลึก สุเมธ. 2559; พุ่มลำเจียก ธนพล. 2559; วิทวัส วิทยา ไกร เลิศ.) และเทคนิคการวิเคราะห์จำแนกประเภทเชิงเส้น (Linear Discriminant Analysis: LDA) (พุ่มลำเจียก ธนพล. 2559; Kumar Rajiv; และคนอื่น ๆ. 2012) จากนั้น เราจะนำข้อมูลแต่ละมุมมองไปสร้างตัวแยกแยะสำหรับแต่ละมุมมองขึ้นมา โดยใช้เทคนิคพื้นฐานที่ใช้กันอยู่ทั่วไป ได้แก่ เทคนิคต้นไม้ตัดสินใจ หรือ เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด เมื่อได้ตัวแยกแยะสำหรับแต่ละมุมมองที่แตกต่างกันแล้ว เราจะรวมคำตอบของตัวแยกแยะจากทุกมุมมองด้วยการใช้เทคนิค การเรียนรู้แบบกลุ่ม

โดยเทคนิคการเรียนรูปแบบกลุ่มนี้ ได้มีงานที่ศึกษามาก่อนหน้านี้ เช่น (ปรเมษฐ์ ธันวานนท์ ชัยกร ยิ่งเสร, และ วรพล พงษ์เพ็ชร.) ได้ใช้เทคนิคการเรียนรู้แบบรวมกลุ่มเพื่อเพิ่มประสิทธิภาพในการพยากรณ์ โดยใช้เทคนิคการจำแนกประเภทแบบการสุ่มป่าไม้ (Random Forest) (Zhao Jing; และคนอื่น ๆ. 2017) เทคนิคการจำแนกประเภทแบบซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines: SVM) (ทองคำ ภูริพัทร์. 2559) เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-nearest neighbor: K-NN) (สินสมบูรณ์ทอง สุรวัชร ศรีเปารยะ และสายชล. 2560; ทองคำ ภูริพัทร์. 2559) และเทคนิค Naïve Bayes (นพมาศปักเข็มและคณะ ข. 2560) ในการทำนายราคาหลักทรัพย์ในตลาดหลักทรัพย์แห่งประเทศไทย โดยเมื่อใช้เทคนิคการเรียนรู้แบบรวมกลุ่ม (Ensemble Model) (ปรเมษฐ์ ธันวานนท์ ชัยกร ยิ่งเสร, และ วรพล พงษ์เพ็ชร.) สามารถเพิ่มประสิทธิภาพในการพยากรณ์ได้ 5-14% และเพิ่มผลตอบแทนในการลงทุน 1-3% และปัจจัยหลักที่สำคัญที่ทำให้ประสิทธิภาพเพิ่มขึ้นคือ ข้อมูลดัชนีชี้วัดทางเทคนิค และจำนวนวันที่ถือหุ้น เดช ธรรมศิริ และพยุงมีสัจ (สัจ เดช ธรรม ศิริพยุง มี. 2011) ได้นำเสนอการจำแนกข้อมูลด้วยวิธีแบบร่วมกันตัดสินใจจากพื้นฐานของเทคนิคต้นไม้ตัดสินใจ เทคนิคโครงข่ายประสาทเทียม และเทคนิคซัพพอร์ตเวกเตอร์แมชชีนร่วมกับการเลือกตัวแทนที่เหมาะสมด้วยขั้นตอนวิธีเชิงพันธุกรรม โดยใช้ข้อมูล Australian Credit ผลที่ได้ใช้เทคนิคต้นไม้ตัดสินใจ 3 โมเดล และเทคนิคโครงข่ายประสาทเทียม 2 โมเดลผลลัพธ์ความถูกต้องเท่ากับ 89.01% ส่วนข้อมูล German Credit ได้ใช้เทคนิคต้นไม้ตัดสินใจ 6 โมเดล และเทคนิคโครงข่ายประสาทเทียม 5 โมเดล ได้ลัพธ์ความถูกต้องเท่ากับ 71.50% และสุดท้ายข้อมูล Bankruptcy Data ใช้การจำแนกข้อมูลทั้งหมด 10 โมเดลร่วมกัน ซึ่งประกอบด้วยเทคนิคซัพพอร์ตเวกเตอร์แมชชีน 4 โมเดล เทคนิคต้นไม้ตัดสินใจ 4 โมเดล และเทคนิคโครงข่ายประสาทเทียม 2 โมเดล ผลลัพธ์ความถูกต้องเท่ากับ 71.50% จากผลที่ได้นั้นทำให้เห็นว่าการใช้เทคนิคร่วมกันตัดสินใจสามารถช่วยเพิ่มประสิทธิภาพความแม่นยำในการจำแนกข้อมูลได้มากกว่าการใช้มุมมองเดียว

1.2 วัตถุประสงค์

(1) เพื่อศึกษาทฤษฎี และวิธีการแยกแยะประเภทแบบหลายมุมมอง โดยใช้ข้อมูลในหลายๆ มุมมอง มาวิเคราะห์

(2) เพื่อเปรียบเทียบการจำแนกประเภทโดยใช้ข้อมูลแบบมุมมองเดียวกับการจำแนกประเภทที่ใช้ข้อมูลแบบหลายมุมมอง

1.3 ขอบเขตของงานวิจัย

(1) ข้อมูลที่นำมาทดลองมาจาก Public Data จาก UCI

(2) สร้างแบบจำลองโดยใช้การแปลงพิกัดเพื่อสร้างข้อมูลแบบหลายมุมมอง ได้แก่ PCA และ LDA

(3) สร้างตัวแยกแยะโดยใช้ข้อมูลจากหลายมุมมองด้วยเทคนิค Decision Tree, K-NN แบบ Ensemble

(4) เปรียบเทียบผลกับแบบจำลองที่ใช้กันโดยทั่วไป

1.4 วิธีดำเนินการวิจัย

(1) ศึกษาและทบทวนวรรณกรรมและงานวิจัยที่เกี่ยวข้อง

(2) ศึกษาเทคนิค PCA, LDA และอัลกอริทึมการทำนายโดยใช้การเรียนรู้ของเครื่องแบบต่างๆ

(3) เปรียบเทียบผลการทำงานของแบบจำลองที่นำเสนอกับแบบจำลองที่ใช้โดยทั่วไป โดยใช้ชุดข้อมูลจาก UCI Data set

(4) วัดผลและเปรียบเทียบความแม่นยำในแบบต่างๆ

(5) ประเมินผล และสรุปผลงานวิจัย

1.5 ประโยชน์ที่คาดว่าจะได้รับจากการวิจัย

(1) ได้องค์ความรู้เกี่ยวกับการจำแนกประเภทแบบหลายมุมมอง

(2) ได้องค์ความรู้เกี่ยวกับการสร้างข้อมูลแบบหลายมุมมองจากข้อมูลมุมมองเดียว

(3) สามารถนำความรู้จากการจำแนกประเภทแบบหลายมุมมอง และการสร้างข้อมูลแบบหลายมุมมองไปใช้กับข้อมูลอื่นๆ ได้

บทที่ 2
ภูมิหลัง ทฤษฎี และงานวิจัยที่เกี่ยวข้อง

เอกสาร และงานวิจัยที่เกี่ยวข้องนี้จะมีเนื้อหาที่เกี่ยวกับภูมิหลัง ทฤษฎีในการเปรียบเทียบการแยกแยะแบบหลายมุมมอง การวัดผลลัพธ์ความถูกต้องแม่นยำในการทำนาย และงานวิจัยที่เกี่ยวข้องซึ่งแบ่งออกเป็น 2 หัวข้อย่อยๆ คือ

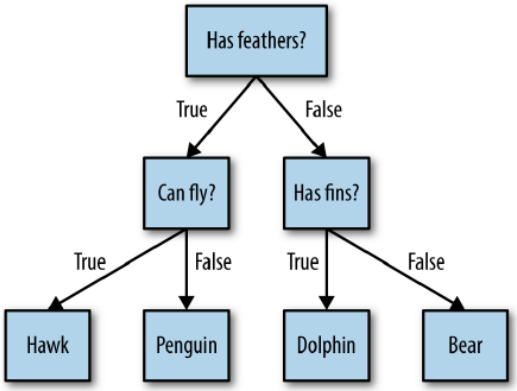
2.1 ทฤษฎีเกี่ยวกับ Machine Learning

2.1.1 Logistic Regression

สมการถดถอยลอจิสติค เป็นการวิเคราะห์ข้อมูลเพื่อศึกษาหาความสัมพันธ์ระหว่างตัวแปรตาม และตัวแปรอิสระ และนำสมการถดถอยที่ได้ไปพยากรณ์ค่าตัวแปรตามหรือไปประมาณเมื่อกำหนดค่าตัวแปรอิสระได้ โดย logistic นี้จะให้ค่า Class เพียง 2 ค่าคือ ใช่ หรือ ไม่ใช่เท่านั้น โดยพิจรณาจากค่าความน่าจะเป็นที่ได้จากโมเดล

2.1.2 Decision Tree

ต้นไม้ตัดสินใจเป็นเทคนิคที่ให้ผลลัพธ์ในลักษณะของโครงสร้างต้นไม้ เมื่อมีข้อมูลที่ต้องการจัดกลุ่มก็จะนำคุณลักษณะต่างๆ ของข้อมูลไปเทียบกันตามกิ่งของต้นไม้จนกระทั่งถึงใบล่างสุด ซึ่งเป็นกลุ่มของข้อมูลที่เหมือนกัน โดยภายในต้นไม้จะประกอบด้วยโหนด โดยแต่ละโหนดมีคุณสมบัติเป็นตัวทดสอบกิ่งของต้นไม้ แสดงถึงค่าที่เป็นไปได้ของคุณลักษณะที่เลือกมาทดสอบและใบ ซึ่งเป็นสิ่งที่อยู่ล่างสุดของต้นไม้ตัดสินใจ แสดงถึงกลุ่มของข้อมูล ก็คือผลลัพธ์ที่ได้จากการทำนาย โหนดที่อยู่บนสุดของต้นไม้เรียกว่า โหนดราก



ที่มา: (Müller Sarah Guido and Andreas. 2016) Introduction to Machine Learning with Python (p. 70)

ภาพประกอบ 1 ตัวอย่าง Decision Tree

จากภาพประกอบ 1 จะเป็นการแสดงแบบจำลองการแยกแยะระหว่างสัตว์สี่ชนิดด้วยกันคือ เหยี่ยว นกเพนกวิน ปลาโลมา และหมี โดยใช้คุณลักษณะของต้นไม้ตัดสินใจคือ มีขนหรือไม่ สามารถบินได้ไหม และมีครีบไหม เป็นต้น

2.1.3 K-Nearest Neighbor

การหาเพื่อนบ้านใกล้ที่สุดเป็นการใช้การจัดแบบแบ่งคลาส โดยใช้การตัดสินใจว่าคลาสไหนที่จะแทนเงื่อนไขหรือกรณีใหม่ๆ ได้ โดยทำการตรวจสอบจำนวนบางจำนวน ของกรณีหรือเงื่อนไขที่เหมือนกันหรือมีความใกล้เคียงมากที่สุด โดยหาจากผลรวมของจำนวนเงื่อนไข หรือกรณีต่างๆ สำหรับแต่ละคลาส กำหนดเงื่อนไขใหม่ๆ ให้คลาสที่มีความเหมือนกันกับคลาสที่ใกล้เคียงกันมากที่สุด

วิธีการทำงาน

เทคนิคของขั้นตอนวิธีการหาเพื่อนบ้านที่ใกล้กันที่สุด เป็นการหาระยะห่างระหว่างแต่ละตัวแปรที่อยู่ในข้อมูล จากนั้นก็จะทำการคำนวณค่าออกมา ซึ่งวิธีนี้จะเหมาะสำหรับข้อมูลแบบตัวเลข ซึ่งตัวแปรที่เป็นค่าแบบไม่ต่อเนื่อง ก็ยังสามารถนำมาหาค่าได้เพียงแต่ต้องทำการจัดการแบบพิเศษ เช่น ถ้าหากเป็นสีจะใช้อะไรวัดความแตกต่างระหว่างสี จากนั้นต้องมีวิธีในการรวมค่าระยะห่างของ แอททริบิวต์ ในทุกค่าที่วัดได้ เมื่อสามารถคำนวณระยะห่างระหว่างเงื่อนไขต่างๆ ได้ จากนั้นก็เลือกชุดของเงื่อนไขที่จะนำมาทำการจัดคลาส มาเป็นฐานสำหรับกับการจัดคลาสของเงื่อนไขใหม่ๆ ได้ และจะตัดสินได้ว่าขอบเขตของจุดข้างเคียงควรจะมีขนาดเท่าไรและอาจมีการตัดสินใจด้วยว่าจะนับจำนวนจุดข้างเคียงได้อย่างไร โดยขั้นตอนมีดังนี้

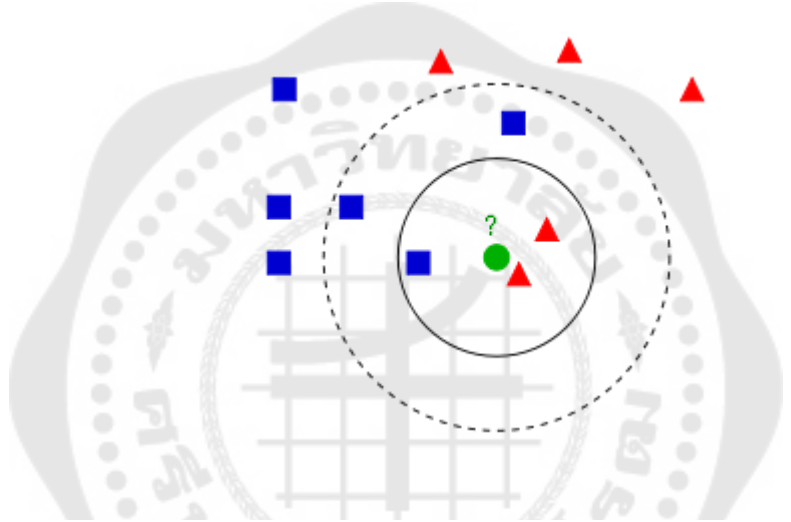
1. กำหนดขนาดของ K (ควรเป็นเลขคี่)

2. คำนวณระยะห่าง ของข้อมูลที่ต้องการพิจารณากับกลุ่มข้อมูลตัวอย่าง

3. จัดลำดับของระยะห่าง และเลือกชุดข้อมูลที่ใกล้จุดที่ต้องการพิจารณาตามจำนวน K ที่กำหนดไว้

4. พิจารณาข้อมูลจำนวน K ชุด และสังเกตว่ากลุ่มไหนที่อยู่ใกล้จุดที่พิจารณาจำนวนมากที่สุด

5. กำหนด class ให้กับจุดที่จะพิจารณาที่ใกล้จุดพิจารณามากที่สุด



ที่มา : https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

ภาพประกอบ 2 แสดงการทำงานของ K-NN

จากภาพประกอบ 2 ต้องการจะหาคำตอบของวงกลมว่าควรจะจัดอยู่ในกลุ่มใดของข้อมูล ซึ่งจากภาพจะเห็นเส้นวงกลมเส้นทึบ โดยได้มีการกำหนดค่าของ K ให้เท่ากับ 3 จึงมีการจับกลุ่มของข้อมูลที่ใกล้กับวงกลมที่ต้องการหาคำตอบจำนวน 3 ตัว ที่อยู่ในวงกลมเส้นทึบ พบว่าภายในเส้นวงกลมเส้นทึบนั้นมีภาพสามเหลี่ยมจำนวน 2 ภาพ ซึ่งมีจำนวนมากกว่าภาพสี่เหลี่ยมที่มีจำนวน 1 ภาพ จึงมีผลให้ วงกลมที่ต้องการหาคำตอบนั้น จะได้คำตอบเป็นสามเหลี่ยม และเมื่อมีการกำหนดค่า K ใหม่ให้ค่า K เท่ากับ 5 จะได้จำนวนข้อมูลที่อยู่ในวงกลมเส้นประที่มีจำนวนข้อมูลที่ใกล้กับวงกลมที่ต้องการหาคำตอบจำนวน 5 ข้อมูลด้วยกัน โดยจำนวนสี่เหลี่ยมมีทั้งหมด 3 ภาพ และจำนวนสามเหลี่ยมมีทั้งหมด 2 ภาพ ทำให้วงกลมที่ต้องการหาคำตอบนั้น จะตอบว่าเป็นสี่เหลี่ยมเพราะภาพสี่เหลี่ยมนั้นมีมากที่สุด

2.1.4 Principal Components Analysis (PCA)

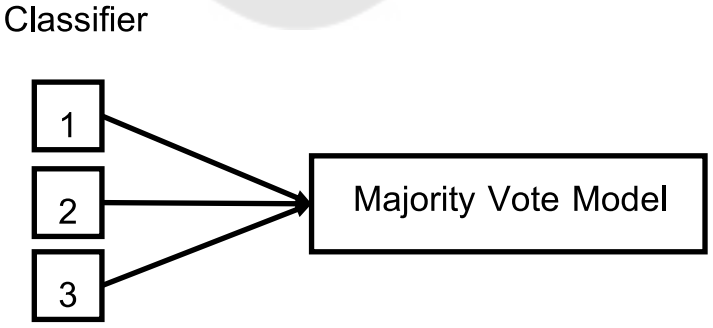
การวิเคราะห์องค์ประกอบหลัก เพื่อลดขนาดเมทริกของตัวแปร โดยเป็นเทคนิคการเปลี่ยนมุมของข้อมูลโดยการตั้งแกนใหม่ขึ้นมาโดยเปลี่ยนจากแกนเดิม ซึ่งจะเปลี่ยนไปในทิศทางที่ข้อมูลกระจายตัวกันอยู่มากที่สุด coordinate ของข้อมูลจะเปลี่ยนระยะพิกัดของข้อมูลจะเปลี่ยนและทำการตัดแกนที่ไม่สำคัญทิ้ง จะเป็นการลดข้อมูลส่วนที่ไม่สำคัญออก เพราะการกระจายตัวในทิศทางนี้น้อยมาก

2.1.5 Linear Discriminant Analysis (LDA)

การวิเคราะห์จำแนกประเภทเชิงเส้น เพื่อลดมิติของข้อมูล สมมุติถ้ามีข้อมูลมีทั้งหมด 3 มิติ ดังนั้น จะเหลือเป็น n-1 คือ 2 มิติ ไดเมนชั่นที่เหลือจะเหลือแค่จุดเดียวเป็นการนำเส้นตรงวัดระยะของข้อมูลเป็นการลากเส้น การหาระยะห่างระหว่างจุด 2 จุด เวลาที่จะวัดจะใช้ไดเมนชั่น 3 จุด โดยหาไดเมนชั่น 2 มิติ กรณีของ LDA ถ้ามี 4 กลุ่มแต่อยู่ใน 2 มิติ ก็จะใช้ LDA ไม่ได้จะใช้ได้ก็ต่อเมื่อจำนวนมิติมากกว่าจำนวนกลุ่มที่มีโดยหาทิศระหว่าง กลุ่มที่มีทั้งหมด และโพเจคไปในทิศนั้น จะใช้ได้ก็ต่อเมื่อจำนวนของมิติมีมากกว่าจำนวนกลุ่มที่มี ซึ่งจำนวนมิติที่เหลือ เท่ากับจำนวน class -1 จำนวนที่ถ้าข้อมูล 2 มิติ ต้องมีมิติกับ Feature และจำนวน Column ของข้อมูลน้อย เช่น ข้อมูลจริงเป็น 2 มิติ LDA จะทำการ ขั้นตอนแรกคือ ลดมิติ ขั้นตอนที่สองเลือกแกนใดแกนนึงที่ทำให้ข้อมูลสองกลุ่มแยกออกจากกัน และขั้นตอนสุดท้ายแกนที่เลือกนี้เลือกทิศทางไหนทิศทางนึง ที่คลาสแยกออกจากกัน และอยู่ห่างกันมากที่สุด

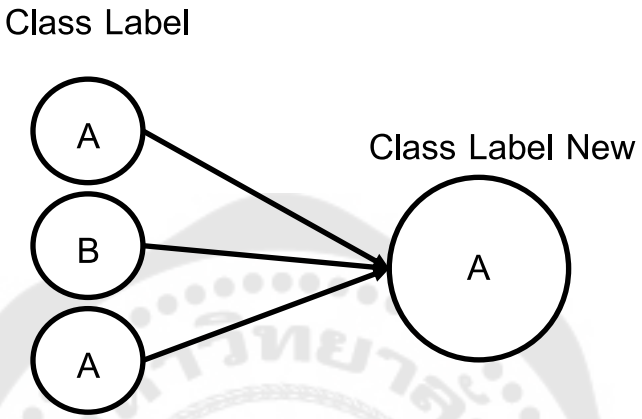
2.1.6 Majority Vote Classifier

เป็นการใช้ผลของการทำตัวจำแนกประเภท 3 วิธีมาทำการโหวตโดยถ้าผลของคำตอบไหนเหมือนกันก็ให้ใช้คำตอบนั้นเป็นคำตอบที่ถูกต้องที่สุด



ภาพประกอบ 3 Diagram แสดง Majority Vote

ภาพประกอบ 3 จะเห็นได้ว่าเป็นภาพการทำงานของวิธี Majority Vote คือ การนำตัวจำแนกประเภทจำนวน 3 วิธีมาทำการโหวตหาค่าคำตอบที่ตอบมากที่สุดและเลือกคำตอบนั้นเพื่อมาสร้างโมเดล Majority Vote



ภาพประกอบ 4 แสดงวิธีการเลือกคำตอบ

2.2 การวัดผลลัพธ์ความถูกต้องแม่นยำในการทำนาย

Accuracy score

ค่าความถูกต้องแม่นยำของการทำนายผล โดยในที่นี้จะใช้คำสั่ง accuracy_score ซึ่งเป็น ฟังก์ชั่นในการคำนวณ ความถูกต้อง แสดงในรูปของ เมทริกซ์คอนฟิวชั่น (Confusion Matrix) โดยความหมายของ เมทริกซ์คอนฟิวชั่น (Confusion Matrix) จะอธิบายได้ดังนี้

		Prediction Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

ที่มา : https://en.wikipedia.org/wiki/Confusion_matrix

ภาพประกอบ 5 อธิบายความหมายของเมทริกซ์คอนฟิวชั่น (Confusion Matrix)

Actual Class เป็นคำตอบจริง เช่น เป็น Label ข้อมูลจริงส่วน Prediction Class เป็นผลคำตอบที่ทำนาย

ค่า True Positive (TP) เป็นจำนวนข้อมูลที่อยู่ในคลาส True และนำมาทำนายได้ผลเป็น True

ค่า False Positive (FP) เป็นจำนวนข้อมูลที่อยู่ในคลาส False แต่เมื่อทำนายได้ผลเป็น True

ค่า True Negative (TN) เป็นจำนวนข้อมูลที่อยู่ในคลาส False และนำมาทำนายได้ผลเป็น False

ค่า False Negative (FN) เป็นจำนวนข้อมูลที่อยู่ในคลาส True แต่เมื่อทำนายได้ผลเป็น False

โดยสูตรการหาค่าความถูกต้องแม่นยำ Accuracy = (TP + TN) / Total Data Test สามารถเขียนได้ดังในสมการ (1)

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

2.3 งานวิจัยที่เกี่ยวข้อง

จากการทบทวนเอกสาร วรรณกรรม และบทความงานวิจัยที่เกี่ยวข้อง สามารถสรุปเป็นหัวข้อได้ดังต่อไปนี้

สุรวัชร ศรีเปารยะ และคณะ (สินสมบูรณ์ทอง สุรวัชร ศรีเปารยะ และสายชล. 2560) ได้ศึกษาวิจัยเรื่องการเปรียบเทียบประสิทธิภาพวิธีการจำแนกกลุ่มการเป็นโรคไตเรื้อรัง : กรณีศึกษาโรงพยาบาลแห่งหนึ่งในประเทศอินเดีย โดยมีวัตถุประสงค์ในการเปรียบเทียบประสิทธิภาพการจำแนกกลุ่ม ใช้ข้อมูลของผู้ป่วยโรคไตเรื้อรังของโรงพยาบาลอพอลโล ในประเทศอินเดีย ซึ่งมาจาก UCI Dataset จำนวนข้อมูลทั้งหมด 400 ระเบียน ซึ่งมีตัวแปรอิสระ 24 ตัวแปร ตัวแปรตาม 1 ตัวแปร วิเคราะห์ด้วยโปรแกรม WEKA โดยใช้วิธีถดถอยลอจิสติก ต้นไม้ตัดสินใจ โครงข่ายประสาทเทียม ซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ ในการวัดประสิทธิภาพจำแนกกลุ่มของผู้ป่วยไตเรื้อรังใน ร.พ.อพอลโล โดยแบ่งสัดส่วน 70, 30 ซึ่งจากการเปรียบเทียบค่าความคลาดเคลื่อนกำลังสองเฉลี่ย (mean square error, MSE) และความถูกต้อง (accuracy) จากการทดลองวิธีที่ดีที่สุดคือวิธีต้นไม้ตัดสินใจ ที่ใช้อัลกอริทึม J48 มีความถูกต้อง 100% คลาดเคลื่อนกำลังสอง = 0.0059

ปรเมษฐ์ ธันวานนท์ และคณะ (ปรเมษฐ์ ธันวานนท์ ชัยกร ยิ่งเสร, และ วรพล พงษ์เพ็ชร.) ได้ศึกษาเรื่องการประยุกต์ใช้โมเดลการเรียนรู้แบบรวมกลุ่ม เพื่อพยากรณ์แนวโน้มของราคาหลักทรัพย์ในตลาดหลักทรัพย์แห่งประเทศไทย ซึ่งมีวัตถุประสงค์ในการเพิ่มผลตอบแทนการลงทุนและเพิ่มประสิทธิภาพการพยากรณ์

โดยใช้ข้อมูล 4 ด้านคือ

1. จำนวนวันที่ถือหุ้นในตลาดหลักทรัพย์ (holding days)

2. ราคาการซื้อขายแต่ละหุ้น

3. ข้อมูลดัชนีทางเทคนิค (Indicators data)

4. จำนวนวันใช้คำนวณค่าดัชนีที่วัดทางเทคนิค โดยใช้เทคนิควิเคราะห์การเรียนรู้ของเครื่อง (Machine Learning) 4 เทคนิค ได้แก่ Random Forest, Support Vector Machine, K-NN, Naïve Bayes และเทคนิคการเรียนรู้แบบรวมกลุ่ม (Ensemble modeling) 2 เทคนิคคือ equal weight และ weight เพื่อทำนายแนวโน้มของราคาหุ้น โดยเป็นข้อมูลตั้งแต่เดือนมกราคม พ.ศ. 2554 ถึง เดือนธันวาคม พ.ศ. 2559 จำนวน 24 เดือน พบว่า เทคนิควิธีการเรียนรู้แบบรวมกลุ่มโดยการถ่วงน้ำหนักเพิ่มประสิทธิภาพการทำนายได้ 5-14 เปอร์เซ็นต์ และเพิ่มผลกำไรได้ 1-3 เปอร์เซ็นต์ ซึ่งจำนวนวันที่ถือนั้นเป็นปัจจัยสำคัญในการเพิ่มประสิทธิภาพการพยากรณ์และผลตอบแทนสำหรับการลงทุน

เดช ธรรมศิริ และพยุงมีสัจ (สัจ เดช ธรรม ศิริพยุง มี. 2011) ได้ศึกษาวิจัยเรื่อง การเรียนรู้แบบรวมกลุ่มด้วยโครงข่ายประสาทเทียบเอดาบูทสำหรับการจำแนกข้อมูล โดยงานวิจัยครั้งนี้ได้ใช้เทคนิคโครงข่ายประสาทเทียมร่วมกับวิธีรวมกลุ่มตัดสินใจแบบเอดาบูท มาทำการทดสอบข้อมูลโรคเบาหวาน (Diabetes Data จาก UCI) โดยใช้ข้อมูลตัวอย่าง 768 ข้อมูล 8 แอตทริบิว วัตถุประสงค์เพื่อดูความถูกต้องในการจำแนกโรค พบว่า เทคนิคร่วมกันตัดสินใจจากหลายโมเดลโดยผ่านการรวมกลุ่มตัดสินใจด้วยเทคนิคเอดาบูท (Neural Network+Adaboost) ให้ผลที่ดีกว่าการใช้เทคนิคที่เป็นโมเดลแบบเดี่ยว ใช้เทคนิค Decision Tree, Neural Netword, Support Vector Machin โดยข้อมูล Diabite Data ได้ผลความถูกต้องเท่ากับ 75.02 เปอร์เซ็นต์ ซึ่งมากกว่าโมเดลเดี่ยว

Kumar et al. (Kumar Rajiv; และคนอื่น ๆ. 2012) เสนอเรื่องUnconstrained Handwritten Numeral Recognition Using Majority Voting Classifier เป็นการรู้จำตัวเลขที่เขียนจากลายมือจากข้อมูล MNIST โดยใช้เทคนิค K-NN, ANN และการวิเคราะห์จำแนกประเภทเชิงเส้น (Linear Discriminant Analysis) ในการแบ่งชุดข้อมูล และคุณลักษณะแตกต่างที่ใช้นั้น

ประกอบไปด้วย Profile - Based Feature และ Kirsch Operator Based Feature โดยมีข้อมูล train 60,000 และ test 10,000 จากการทดลองพบว่าอัตราการรู้จำที่สูงสุดคือ 93% เมื่อใช้เทคนิคการจำแนกประเภทเพียงตัวเดียว และเมื่อใช้ Majority Voting ปรากฏว่าอัตราการรู้จำมีค่าเพิ่มสูงขึ้นเป็น 98.05%

บทที่ 3
วิธีดำเนินการวิจัย

จากการศึกษางานวิจัยที่เกี่ยวข้อง ผู้วิจัยพบว่าในงานวิจัยที่ได้กล่าวไว้ข้างต้นใช้วิธีรวมกลุ่มตัดสินใจในแบบต่างๆ มากมายเพื่อเพิ่มประสิทธิภาพในการหาผลลัพธ์ ผู้วิจัยจึงนำแนวความคิดดังกล่าวมาทำการทดลองเพิ่มเติมเพื่อเพิ่มประสิทธิภาพของงานวิจัยในชิ้นนี้โดยมีขั้นตอนดังนี้

3.1 แผนการดำเนินงาน

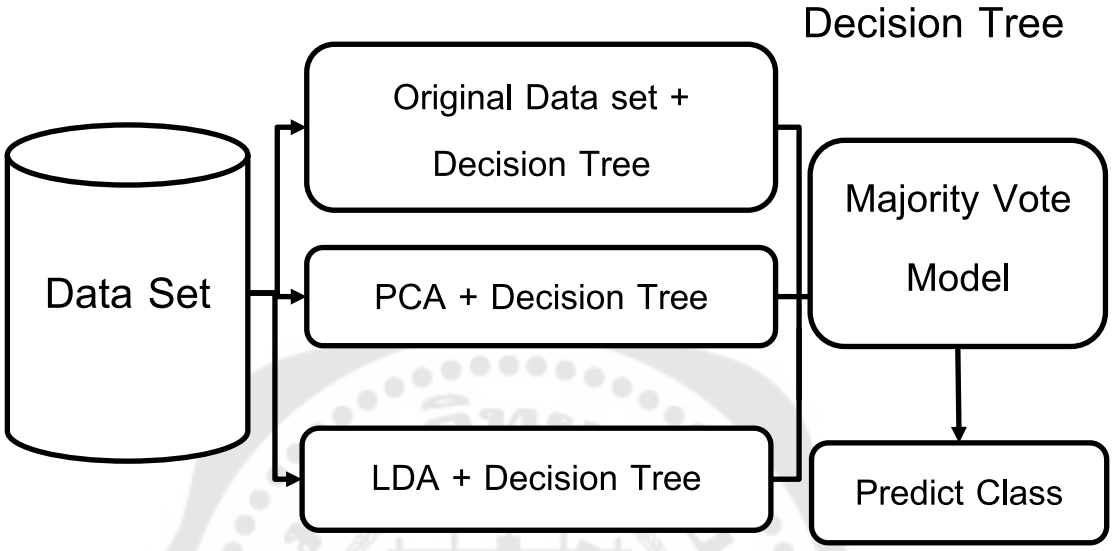
ตาราง 1 ตารางแผนการดำเนินงาน

กิจกรรม	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.
1. ทบทวนวรรณกรรม และงานวิจัยที่เกี่ยวข้อง	X	X				
2. วางแผนการวิจัย		X				
3. เตรียมข้อมูล			X			
4. เริ่มทำการทดลอง				X		
5. สรุป และแก้ไขแบบจำลอง					X	
6. สรุปผล						X

ตาราง 1 เป็นแผนการดำเนินงาน ตั้งแต่เริ่มการทำงานไปจนถึงจบการทำงาน โดยมีระยะเวลา ทั้งหมด 7 เดือน

3.2 ขั้นตอนวิธีการทำงานของระบบ

วิธีทดลอง



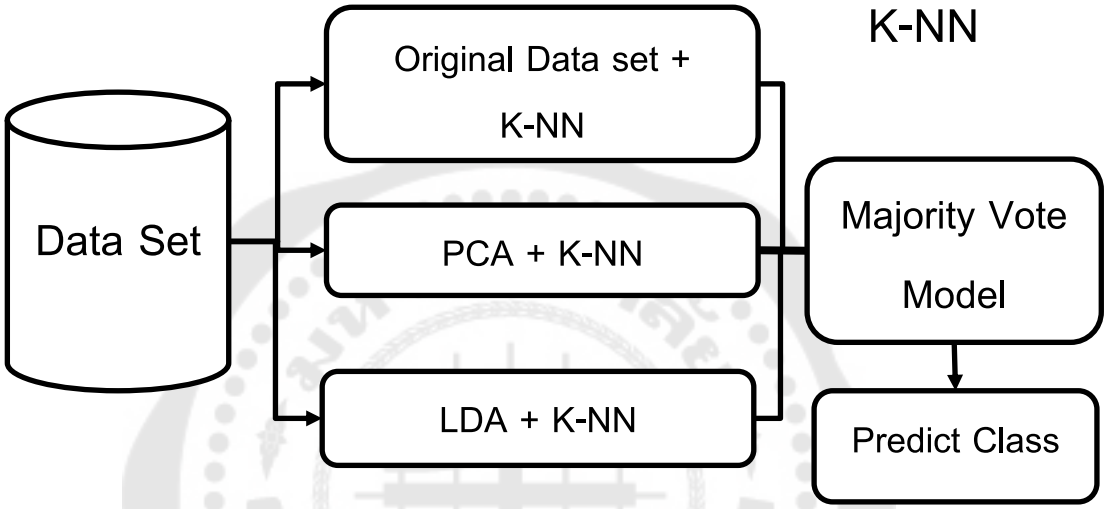
ภาพประกอบ 6 ขั้นตอนการทำงานที่ใช้เทคนิค Decision Tree

ภาพประกอบ 6 จะเป็นการแสดงขั้นตอนการทำงานของระบบโดยจากข้อมูลทั้ง 4 ข้อมูลที่นำมาจาก UCI ผู้วิจัยใช้เทคนิคการจำแนกประเภทของข้อมูลด้วยเทคนิคต้นไม้ตัดสินใจ (Decision Tree) นำมาสร้างมุมมองด้วยวิธีการที่แตกต่างกัน สามมุมมอง โดยในมุมมองแรก จะใช้ข้อมูลตั้งต้นเดิม (Full Feature) โดยใช้ฟีเจอร์ทั้งหมด ข้อมูลในมุมมองที่สอง จะสร้างโดยใช้เทคนิค PCA บิดแกนไปในทิศทางที่ข้อมูลอยู่มากที่สุดโดยแกนแรกชี้ไปในทิศที่ข้อมูลอยู่มากที่สุด แกนที่สองจะต้องตั้งฉากกับแกนแรกเสมอและจะมีข้อมูลกระจายตัวลองลงมาจากแกนแรก เป็นเช่นนี้ไปตามลำดับ ในงานวิจัยนี้ ผู้วิจัยได้ทำ PCA ก่อน จากนั้นจะเลือกแกนหลักเพียง 7 แกนหลัก (n_components เท่ากับ 7) แล้วจึงไปทำสร้างตัวแยกแยะแบบ Decision Tree ส่วนในมุมมองที่สามจะใช้เทคนิค LDA ในการเปลี่ยนแกนข้อมูล ทั้งนี้ เนื่องจากในแต่ละชุดข้อมูล จำนวนของมิติที่เหลือในเทคนิค LDA นี้ จะเท่ากับจำนวนของคลาสของข้อมูลลบด้วยหนึ่ง ดังนั้นในมุมมองนี้จำนวนมิติของข้อมูลจึงถูกกำหนดด้วยจำนวนคลาสของข้อมูลอยู่แล้ว หลังจากได้ตัวแยกแยะสำหรับทั้งสามมุมมองแล้ว เราจะหาคลาสของคำตอบรวมจากทั้งสามตัวแยกแยะโดยใช้ใช้วิธี Majority Vote

โดยผู้วิจัยได้กำหนด Parameter ที่ใช้เทคนิค Decision Tree ดังนี้

(1) แบ่ง Train 70 Test 30 และ random_state เท่ากับ 5

(2) Decision Tree ตัวจำแนกประเภทกำหนดค่า min_samples_split เท่ากับ 2 และ random_state เท่ากับ 5



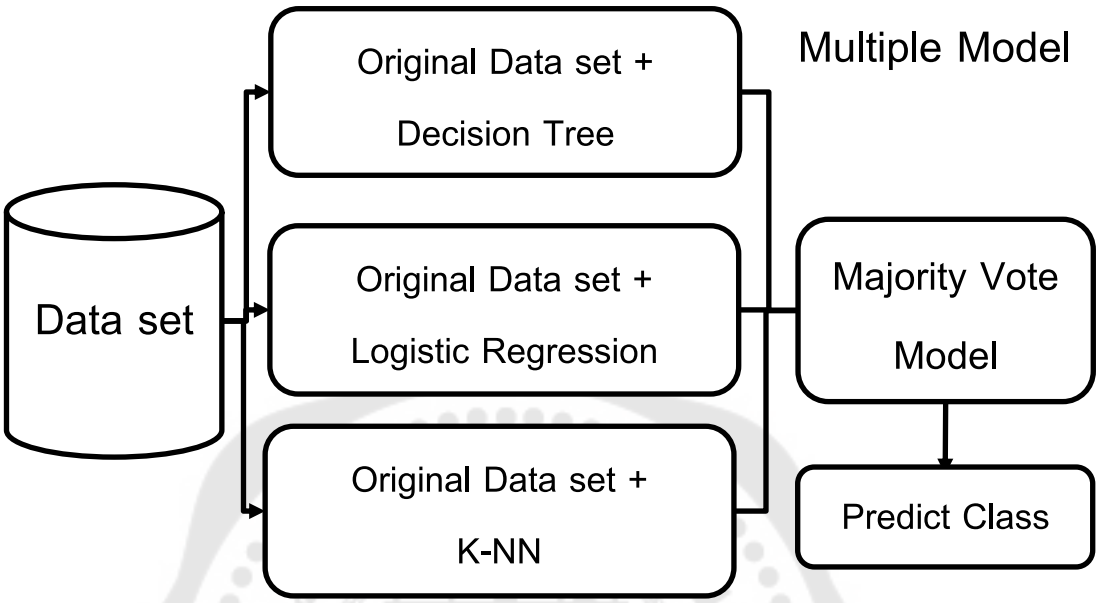
ภาพประกอบ 7 ขั้นตอนการทำงานที่ใช้เทคนิค K-NN

ภาพประกอบ 7 เป็นกระบวนการทำงานโดยจะใช้เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-nearest neighbor: K-NN) และดำเนินการตามขั้นตอนที่เหมือนกับ เทคนิคต้นไม้ตัดสินใจ (Decision Tree)

โดยผู้วิจัยได้กำหนด Parameter ที่ใช้เทคนิค K-NN ดังนี้

(1) แบ่ง Train 70 Test 30 และ random_state เท่ากับ 5

(2) KNeighbors ตัวจำแนกประเภทกำหนดค่า leaf_size เท่ากับ 30 และ n_neighbors เท่ากับ 5



ภาพประกอบ 8 แสดงการทำงานของ Multiple Model

ภาพประกอบ 8 จะแสดงการทำงานของ Multiple Model โดยใช้ข้อมูลตั้งต้นเดิม มาทำการวิเคราะห์ด้วย 4 วิธีคือ

(1) เทคนิคการถดถอยโลจิสติก (Logistic Regression)

(2) เทคนิคต้นไม้ตัดสินใจ (Decision Tree)

(3) เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-NN)

(4) Voting ตัวจำแนกประเภทโดยตั้งค่าแบบ Hard คือการนำเอาผลคำตอบสุดท้ายที่โมเดลแต่ละตัวทำนายมาโหวตกัน เช่น ในกรณี Binary Classification ผลลัพธ์จะได้คำตอบเป็น 0 หรือ 1 ถ้ามี 3 โมเดลคือ A, B, C และแต่ละตัวทำนายผลเป็น 0,0,1 ก็จะโหวตผลคำตอบสุดท้ายเป็น 0

โดยผู้วิจัยได้กำหนด Parameter แต่ละเทคนิคดังนี้

(1) Logistic Regression กำหนดค่า C เท่ากับ 1.0และmulti_class เท่ากับ multinomial

(2) Decision Tree ตัวจำแนกประเภทกำหนดค่า min_samples_split เท่ากับ 2 และ random_state เท่ากับ 5

(3) KNeighbors ตัวจำแนกประเภทกำหนดค่า leaf_size เท่ากับ 30 และ n_neighbors เท่ากับ 5

(4) Voting ตัวจำแนกประเภทกำหนดค่า max_iter เท่ากับ 100 และvoting เท่ากับ hard

บทที่ 4
ผลการดำเนินงานวิจัย

จากวิธีการทดลองขั้นต้นผู้วิจัยได้ใช้ข้อมูล 4 ข้อมูลนำมาทำการทดลองมีดังนี้

4.1 อธิบายข้อมูล

ข้อมูลที่นำมาทดลอง ใช้ข้อมูลจาก UCI (Dua Dheeru and Graff, Casey. 2017) โดยใช้ทั้งหมด 4 ข้อมูลดังนี้

(1) Image Segmentation เป็นข้อมูลของภาพที่ถูกถ่ายจากดาวเทียมเพื่อดูว่าภาพๆนั้นเป็นภาพอะไร ซึ่งข้อมูลที่ได้เป็นข้อมูลตัวเลขที่มีการแบ่งภาพดิจิทัลออกเป็นหลายส่วน เป้าหมายของการแบ่งส่วนคือการทำให้ง่ายขึ้นและเปลี่ยนการเป็นตัวแทนของภาพเป็นสิ่งที่มีความหมายและง่ายต่อการวิเคราะห์

Image Segmentation Data Set

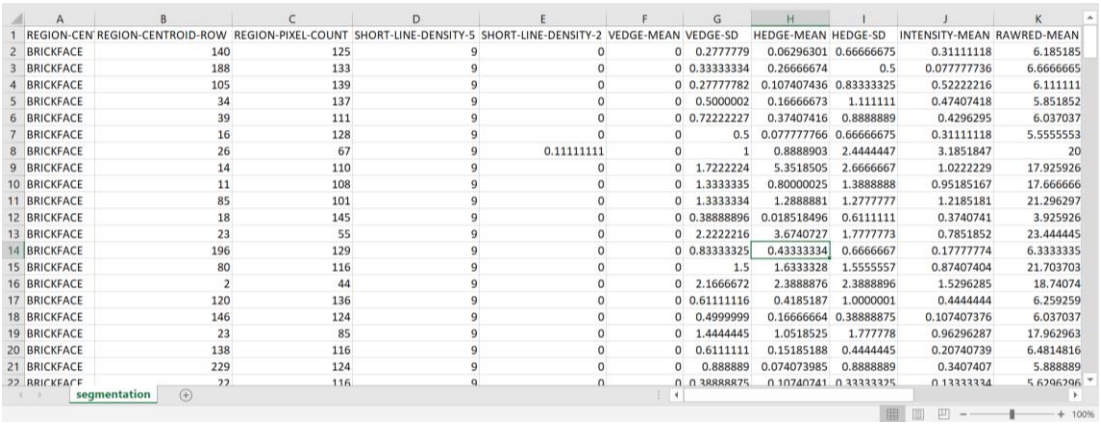
Download: Data Folder, Data Set Description

Abstract: Image data described by high-level numeric-valued attributes, 7 classes

Data Set Characteristics:	Multivariate	**Number of Instances:**	2310	**Area:**	N/A
Attribute Characteristics:	Real	**Number of Attributes:**	19	**Date Donated**	1990-11-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	171260

ภาพประกอบ 9 รายละเอียดของชุดข้อมูล Image Segmentation

ภาพประกอบ 9 เป็นข้อมูลที่เกี่ยวกับการจำแนกวัตถุต่างๆ ที่ถูกถ่ายจากดาวเทียมจะแสดงลักษณะของกลุ่มข้อมูล Image Segmentation ซึ่งเป็นข้อมูลภาพถ่าย โดยจำแนกว่าภาพนั้นเป็นภาพของอะไร เช่น แก้ว หญ้า เป็นต้น โดยมีค่าคลาส 7 คลาส จากฐานข้อมูลภาพกลางแจ้ง 7 ภาพ แต่ละอินสแตนซ์เป็นภาพ 3x3 พิกเซล โดยเป็นข้อมูลที่มาจาก UCI ในปี 1990 ประกอบไปด้วย 19 แอตทริบิวต์



	A	B	C	D	E	F	G	H	I	J	K
1	REGION-CEN	REGION-CENTROID-ROW	REGION-PIXEL-COUNT	SHORT-LINE-DENSITY-5	SHORT-LINE-DENSITY-2	VEDGE-MEAN	VEDGE-SD	HEDGE-MEAN	HEDGE-SD	INTENSITY-MEAN	RAWRED-MEAN
2	BRICKFACE	140	125	9	0	0	0.2777779	0.06296301	0.66666675	0.31111118	6.185185
3	BRICKFACE	188	133	9	0	0	0.33333334	0.26666674	0.5	0.077777736	6.6666665
4	BRICKFACE	105	139	9	0	0	0.27777782	0.107407436	0.83333325	0.52222216	6.111111
5	BRICKFACE	34	137	9	0	0	0.5000002	0.16666673	1.111111	0.47407418	5.851852
6	BRICKFACE	39	111	9	0	0	0.72222227	0.37407416	0.8888889	0.4296295	6.037037
7	BRICKFACE	16	128	9	0	0	0.5	0.077777766	0.66666675	0.31111118	5.5555553
8	BRICKFACE	26	67	9	0.11111111	0	1	0.8888903	2.4444447	3.1851847	20
9	BRICKFACE	14	110	9	0	0	1.7222224	5.3518505	2.6666667	1.0222229	17.925926
10	BRICKFACE	11	108	9	0	0	1.3333335	0.80000025	1.3888888	0.95185167	17.666666
11	BRICKFACE	85	101	9	0	0	1.3333334	1.2888881	1.2777777	1.2185181	21.296297
12	BRICKFACE	18	145	9	0	0	0.38888896	0.018518496	0.6111111	0.3740741	3.925926
13	BRICKFACE	23	55	9	0	0	2.2222216	3.6740727	1.7777773	0.7851852	23.444445
14	BRICKFACE	196	129	9	0	0	0.83333325	0.43333334	0.6666667	0.17777774	6.3333335
15	BRICKFACE	80	116	9	0	0	1.5	1.6333328	1.5555557	0.87407404	21.703703
16	BRICKFACE	2	44	9	0	0	2.1666672	2.3888876	2.3888896	1.5296285	18.74074
17	BRICKFACE	120	136	9	0	0	0.61111116	0.4185187	1.0000001	0.4444444	6.259259
18	BRICKFACE	146	124	9	0	0	0.4999999	0.16666664	0.38888875	0.107407376	6.037037
19	BRICKFACE	23	85	9	0	0	1.4444445	1.0518525	1.777778	0.96296287	17.962963
20	BRICKFACE	138	116	9	0	0	0.6111111	0.15185188	0.4444445	0.20740739	6.4814816
21	BRICKFACE	229	124	9	0	0	0.888889	0.074073985	0.8888889	0.3407407	5.888889
22	BRICKFACE	22	116	9	0	0	0.38888875	0.10740741	0.33333325	0.13333334	5.6296296

segmentation

ภาพประกอบ 10 ตัวอย่างข้อมูล Image Segmentation

ภาพประกอบ 10 แสดงตัวอย่างข้อมูล Image Segmentation โดยคอลัมน์แรก region-centroid-col เป็นคำตอบของข้อมูล คอลัมน์ที่สอง region-centroid-row เป็นแถวพิกเซลของภาพ the row of the center pixel of the region เป็นจำนวนพิกเซลของภาพ และคอลัมน์อื่นๆ ที่แสดงถึงค่าของการวัดสี ความสว่างของสี ความเข้มของสีโดยมีหลักคือ Red Green Blue เป็นต้น

(2) Spambase เป็นการจำแนกประเภทของอีเมล โดยมีข้อมูลอยู่ 2 กลุ่มด้วยกันคือ กลุ่มแรกเป็นข้อมูลของอีเมลปกติ และกลุ่มที่สองเป็นข้อมูลที่เป็นอีเมลสแปม

Spambase Data Set

Download: Data Folder, Data Set Description

Abstract: Classifying Email as Spam or Non-Spam

Data Set Characteristics:	Multivariate	**Number of Instances:**	4601	**Area:**	Computer
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	57	**Date Donated**	1999-07-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	410645

ภาพประกอบ 11 รายละเอียดของชุดข้อมูล Spambase

ภาพประกอบ 11 จะแสดงลักษณะของกลุ่มข้อมูล Spambase ซึ่งเป็นข้อมูลจำแนกประเภทของอีเมลว่าเป็น อีเมลปกติหรือเป็นอีเมลสแปม โดยมีค่าเป็นตัวเลขต่อเนื่องจำนวนเต็มและจำนวนจริง จากฐานข้อมูลทั้งสองกลุ่มโดยเป็นข้อมูลที่มาจาก UCI ในปี 1999 ประกอบไปด้วย 57 แอตทริบิวต์

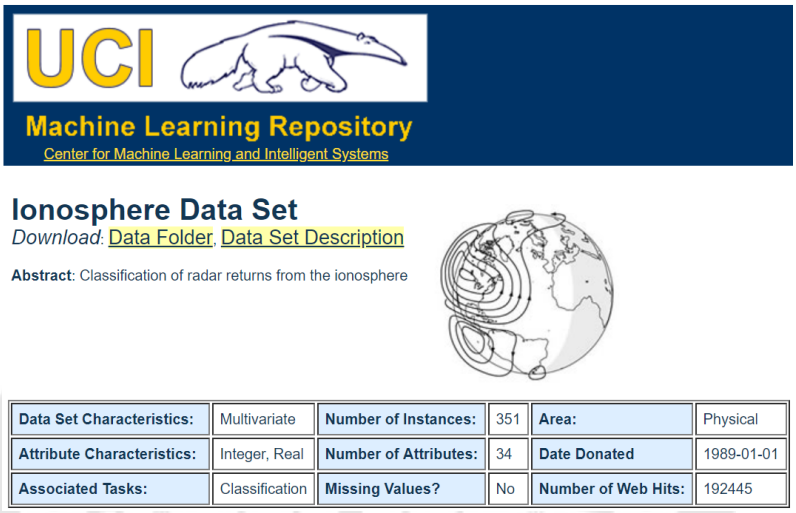
ตาราง 2 ตารางอธิบายข้อมูล Spambase

Name	Description	Type
word_freq_WORD	จำนวนครั้งที่คำปรากฏใน E-mail	48 continuous real [0,100]
char_freq_CHAR	จำนวนอักขระทั้งหมดในอีเมล	6 continuous real [0,100]
capital_run_length_average	ความยาวเฉลี่ยตัวอักษรพิมพ์ใหญ่	1 continuous real [1,...]
capital_run_length_longest	ความยาวของตัวอักษรพิมพ์ใหญ่ที่ต่อเนื่องกัน	1 continuous integer [1,...]
capital_run_length_total	ผลรวมของความยาวตัวอักษรพิมพ์ใหญ่ จำนวนตัวอักษรทั้งหมดในอีเมล	1 continuous integer [1,...]
spam	คำตอบว่าเป็นสแปมหรือไม่	1 nominal class

ตาราง 2 เป็นการแสดงการอธิบายข้อมูล Spambase และคุณลักษณะของข้อมูลที่นำมาใช้ทำการทดสอบ

(3) Ionosphese เป็นข้อมูลการส่งเรดาร์จากชั้นบรรยากาศโลก เพื่อตรวจสอบว่าสัญญาณที่ทำการส่งมายังโลกนั้น มีสัญญาณที่มีคุณภาพหรือไม่ ที่รวบรวมโดยระบบใน Goose Bay, Labrador ซึ่งระบบนี้ประกอบด้วยชุดเสาอากาศความถี่สูงจำนวน 16 ส่วนพร้อมกำลังส่งทั้งหมดตามลำดับ 6.4 กิโลวัตต์เป้าหมายคือ อิเล็กตรอนอิสระในชั้นบรรยากาศ โดยมีผลการรับเรดาร์ “ดี” คือแสดงให้เห็นโครงสร้างบางประเภทในบรรยากาศรอบนอกโลก หากเป็นผล “แย่” คือ

สัญญาณไม่สามารถผ่านชั้นบรรยากาศได้ สัญญาณที่ได้รับนั้นประมวลผลจากฟังก์ชั่น autocorrelation โดยข้อโต้แย้งคือเวลาของพัลส์ และหมายเลขพัลส์ มีหมายเลขชีพจร 17 หมายเลขสำหรับระบบ Goose Bay อินสแตนซ์ใน database นี้ โดยอธิบายได้ 2 คุณลักษณะต่อหมายเลขพัลส์ซึ่งสอดคล้องกับค่าที่ซับซ้อนที่ส่งคืนจากสัญญาณแม่เหล็กไฟฟ้าที่ซับซ้อน ข้อมูลมี 34 ข้อมูล โดยเป็นตัวเลขต่อเนื่อง


UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

Ionosphere Data Set

Download: Data Folder, Data Set Description

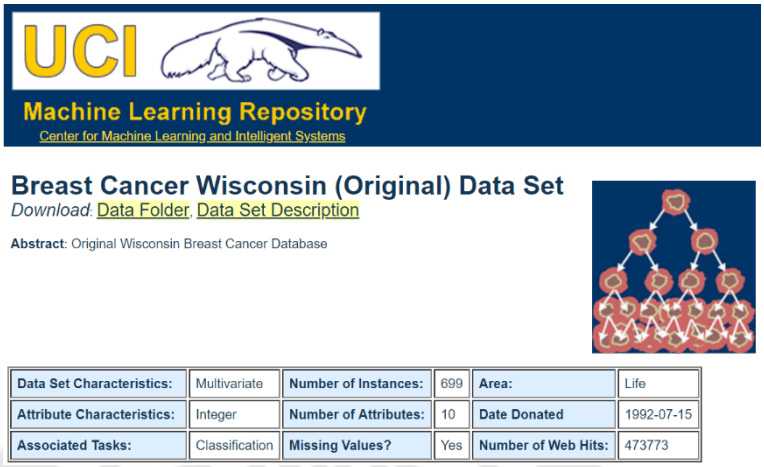
Abstract: Classification of radar returns from the ionosphere

Data Set Characteristics:	Multivariate	**Number of Instances:**	351	**Area:**	Physical
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	34	**Date Donated**	1989-01-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	192445

ภาพประกอบ 12 รายละเอียดของชุดข้อมูล Ionosphere

ภาพประกอบ 12 จะแสดงลักษณะข้อมูล Ionosphere ซึ่งเป็นข้อมูลการจำแนกประเภทของการรับเรดาร์จากชั้นบรรยากาศ โดยมีค่าเป็นจำนวนเต็ม และจำนวนจริง จากการรวบรวมโดยระบบใน Goose Bay, Labrador ที่มีชุดเสาอากาศความถี่สูงจำนวน 16 ส่วน ที่มีกำลังส่ง 6.4 กิโลวัตต์ โดยเป็นข้อมูลที่มาจาก UCI ในปี 1989 ประกอบไปด้วย 34 แอตทริบิวต์โดย 1-34 ค่าทั้งหมดเป็นค่าต่อเนื่องจำนวนจริงทั้งหมด และคำตอบจะอยู่ที่ แถวที่ 35 จะมีผลคำตอบเป็น “good” หรือ “bad”

(4) Breast Cancer Wisconsin เป็นข้อมูลจากการตรวจเซลล์ต่างๆ เพื่อนำมาจำแนกประเภทว่าเป็นมะเร็งเต้านมหรือไม่เป็นมะเร็งเต้านม



Data Set Characteristics:	Multivariate	**Number of Instances:**	699	**Area:**	Life
Attribute Characteristics:	Integer	**Number of Attributes:**	10	**Date Donated**	1992-07-15
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	473773

ภาพประกอบ 13 รายละเอียดของชุดข้อมูล Breast Cancer Wisconsin (Original)

ภาพประกอบ 13 แสดงลักษณะข้อมูล Breast Cancer Wisconsin (Original) ซึ่งเป็นข้อมูลการจำแนกประเภทของเซลล์ โดยมีค่าเป็นจำนวนเต็ม จากการรวบรวมโดยDr. Wolberg ข้อมูลจากคลินิก โดยมาจาก UCI Dataset ในปี 1992 ประกอบไปด้วย 10 แอตทริบิวต์

ตาราง 3 อธิบายข้อมูลของ Breast Cancer Wisconsin (Original)

Name	Description	Ex.	Type
Sample code number	ID Number	1000025	Category
Clump Thickness	ความหนา	5	Continuous
Uniformity of Cell Size	ขนาดของเซลล์	1	Continuous
Uniformity of Cell Shape	รูปร่างของเซลล์	1	Continuous
Marginal Adhesion	การยึดติดกับผิว	1	Continuous
Single Epithelial Cell Size	ขนาดเยื่อบุผิว	2	Continuous
Bare Nuclei	นิวเคลียส	1	Continuous
Bland Chromatin	ผิวสัมผัสของโครมาติ	3	Continuous
Normal Nucleoli	นิวคลีโอลี	1	Continuous
Mitoses	Mitoses	1	Continuous
Class	คลาสของคำตอบจะแทนด้วย 2 ค่า คือ 2 เป็นเซลล์ดี และ 4 เป็นเซลล์มะเร็ง	2	Category

ตาราง 3 เป็นการอธิบายข้อมูลของชุดข้อมูล Breast Cancer Wisconsin (Original) ที่มีทั้งหมด 10 แอตทริบิวต์

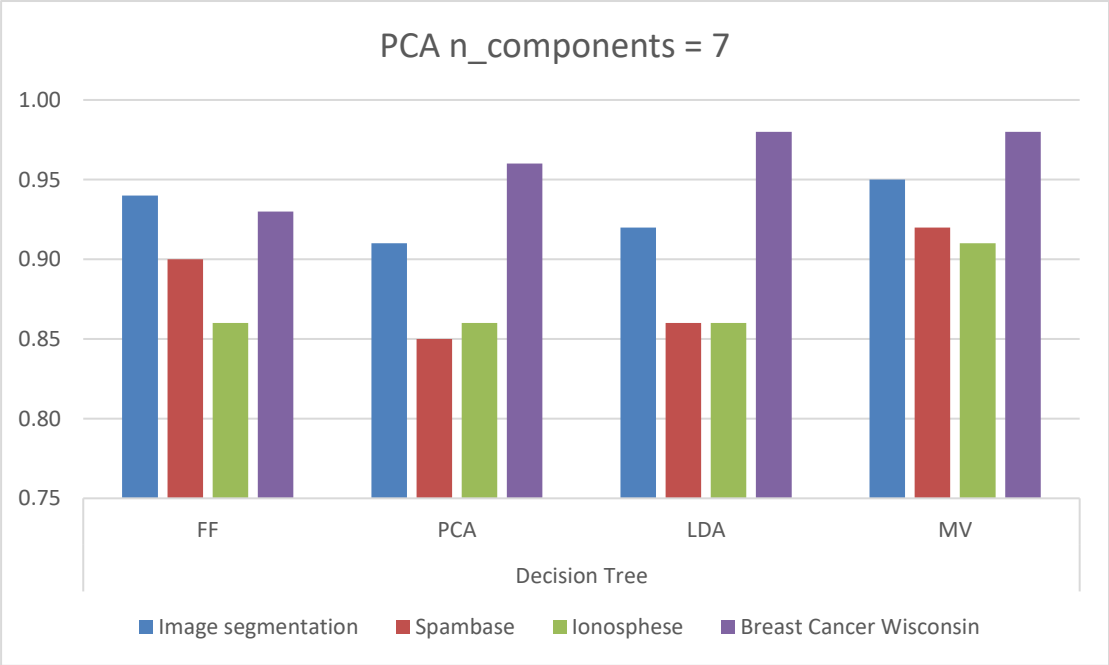
4.2 ผลการทดลอง

จากข้อมูลข้างต้นได้นำมาทำการทดลองโดยนำข้อมูลที่ได้มาทำการปรับทิศทางของข้อมูลในหลากหลายวิธีเช่น PCA และ LDA และใช้ตัวจำแนกประเภทแบบ Decision Tree และ K-NN เมื่อได้ผลของแต่ละวิธีก็นำมาทำ Multi-View แล้วนำผลมาเปรียบเทียบกับการทำ Multiple Model เพื่อเปรียบเทียบผลลัพธ์ที่ได้

ตาราง 4 ผลการทดสอบแบบ Decision Tree

DATA	Full Feature	PCA	LDA	Multi-View
Image Segmentation	0.94	0.91	0.92	0.95
Spambase	0.90	0.85	0.86	0.92
Ionosphese	0.86	0.86	0.86	0.91
Breast Cancer Wisconsin	0.93	0.96	0.98	0.98

ตาราง 4 จะเห็นได้ว่าค่าความแม่นยำของการใช้ตัวแยกแยะที่ใช้การรวมค่าจากหลายมุมมอง ได้ผลดีกว่าการแยกแยะโดยใช้ข้อมูลจากแต่ละมุมมอง ได้แก่ การนำฟีเจอร์ทั้งหมดมาใช้ การใช้เทคนิค PCA และ การใช้เทคนิค LDA มาทำการทดสอบ



ภาพประกอบ 14 การทดสอบด้วยเทคนิค Decision Tree โดยกำหนดค่า PCA ให้ n_components เท่ากับ 7

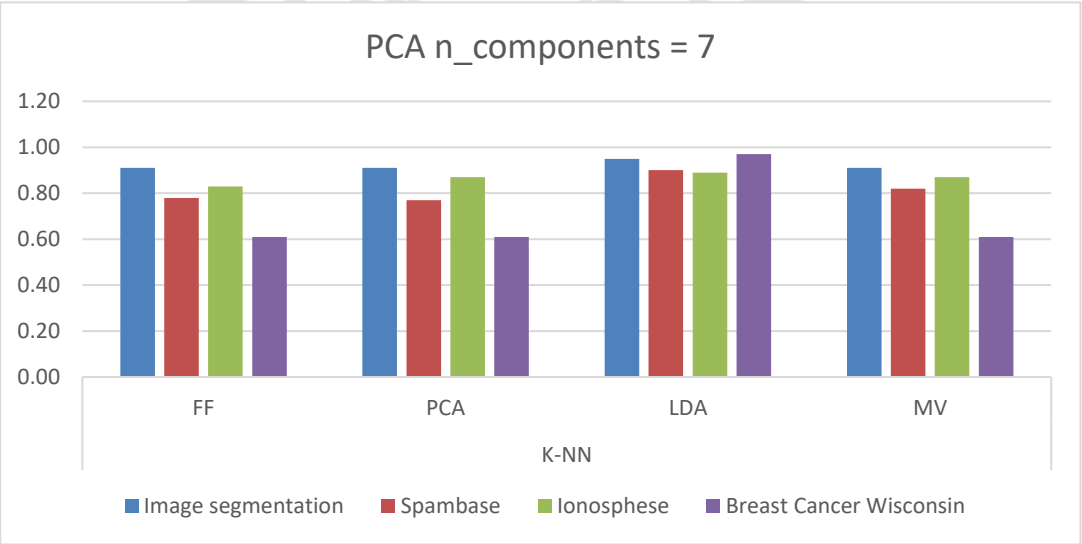
ภาพประกอบ 14 จะเห็นว่าข้อมูล Breast Cancer Wisconsin นั้น มีค่าที่ใกล้เคียง 1.0 มากที่สุดเมื่อเทียบกับข้อมูลอื่นๆ โดยวิธีที่ให้ผลมากที่สุดจะเป็นวิธี LDA และ Multi-view ที่ได้ผลถึง 0.98 ซึ่งถ้าเทียบกับวิธี PCA ที่ใช้ ข้อมูลชุดเดียวกันที่ได้ 0.96 ซึ่งมีค่ามากกว่าถึง 0.02

การทดลองที่สอง เราสร้างตัวแยกแยะสำหรับข้อมูลแต่ละมุมมองด้วยด้วยเทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-NN)

ตาราง 5 ผลตารางการทดสอบแบบ K-NN

DATA	Full Feature	PCA	LDA	Multi-View
Image Segmentation	0.91	0.91	0.95	0.91
Spambase	0.78	0.77	0.90	0.80
Ionosphese	0.83	0.87	0.89	0.87
Breast Cancer Wisconsin	0.61	0.61	0.97	0.61

ตาราง 5 จะเห็นได้ว่าค่าของ Accuracy ของการใช้ LDA ได้ผลดีกว่าในกรณีอื่นทั้งหมด รวมทั้งยังดีกว่าในกรณีของการใช้ข้อมูลแบบหลายมุมมองอีกด้วย ทั้งนี้ เนื่องจาก การใช้เทคนิคแบบหลายมุมมองนั้นจะให้คะแนนที่มีน้ำหนักมากกว่าเป็นคำตอบที่ถูกต้อง แต่การจำแนกแยกแยะของสองประเภท คือ PCA และ Full Feature ในแบบ K-NN นั้นให้คำตอบที่ผิดทำให้เมื่อนำคำตอบที่ได้จาก PCA และ Full Feature มาทำการจำแนกแบบหลายมุมมองจึงทำให้ค่าของผลแบบ Multi-View ได้ค่าน้อยกว่า LDA เพราะ LDA ที่ได้ค่ามากค่าเดียวไม่สามารช่วยให้ได้ผลลัพธ์ที่ดีขึ้นได้



ภาพประกอบ 15 การทดสอบด้วยเทคนิค K-NN โดยกำหนดค่า PCA ให้ n_components เท่ากับ 7

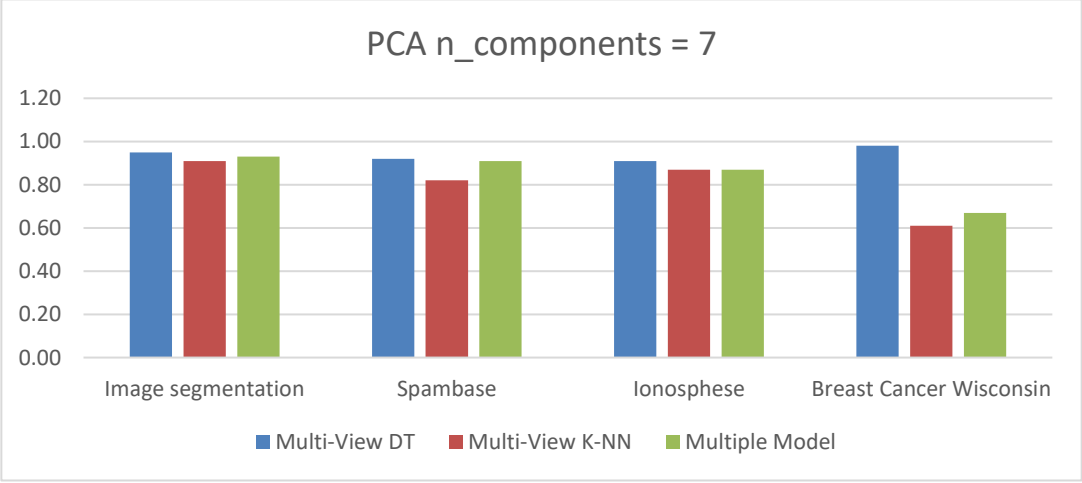
ภาพประกอบ 15 จะพบว่าเทคนิค K-NN แบบ LDA จะได้ค่าใกล้เคียง 1.0 ของทุกข้อมูลที่นำมาทดลองโดยข้อมูลที่ได้ผลลัพธ์มากที่สุดนั้นเป็นข้อมูล Breast Cancer Wisconsin โดยได้ค่าเท่ากับ 0.97 และลองลงมาเป็นข้อมูล Image Segmentation ได้ค่าเท่ากับ 0.95

ในการทดลองสุดท้าย เราเปรียบเทียบความแม่นยำระหว่าง การใช้เทคนิคการแยกแยะแบบหลายมุมมอง กับการใช้เทคนิคการแยกแยะแบบหลายโมเดล โดยโมเดลที่นำมาใช้ทดลองได้แก่ โมเดล logistic regression และ random forest และ K-NN แสดงในตารางต่อไปนี้

ตาราง 6 เปรียบเทียบ Multi View กับ Multiple Model

DATA	Multi View Decision Tree	Multi View KNN	Multiple Model
Image Segmentation	0.95	0.91	0.93
Spambase	0.92	0.82	0.91
Ionosphese	0.91	0.87	0.87
Breast Cancer Wisconsin	0.98	0.61	0.67

ตาราง 6 จะเห็นได้ว่าค่าของ Accuracy ของการใช้เทคนิคการแยกแยะแบบหลายมุมมอง โดยใช้ตัวแยกแยะ แบบ Decision Tree ให้ค่าความแม่นยำที่สูงที่สุด



ภาพประกอบ 16 การเปรียบเทียบ Multi-View ด้วยเทคนิค Decision Tree และ K-NN กับ Multiple Model โดยกำหนดค่า PCA n_componants เท่ากับ 7

ภาพประกอบ 16 จะพบว่าข้อมูล Breast Cancer Wisconsin นั้นได้ค่าที่สูงเมื่อใช้เทคนิค Multi-View ด้วยเทคนิค Decision Tree โดยมีค่าเท่ากับ 0.98 แต่หากใช้เทคนิคอื่นๆ จะได้ค่าน้อย แต่หากดูที่ข้อมูลของ Image segmentation นั้นจะพบว่าได้ค่าที่มากทั้งหมดทุกๆ เทคนิค คือทั้งเทคนิค Multi-View DT ที่ได้ค่าเท่ากับ 0.95 Multi-View K-NN ได้ค่าเท่ากับ 0.91 และ Multiple Model ได้ค่าเท่ากับ 0.93 ซึ่งทำให้พบว่า การที่มีข้อมูลการจำแนกประเภทแบบหลายประเภทอาจทำให้ผลที่ได้นั้นมีค่าสูงขึ้นได้

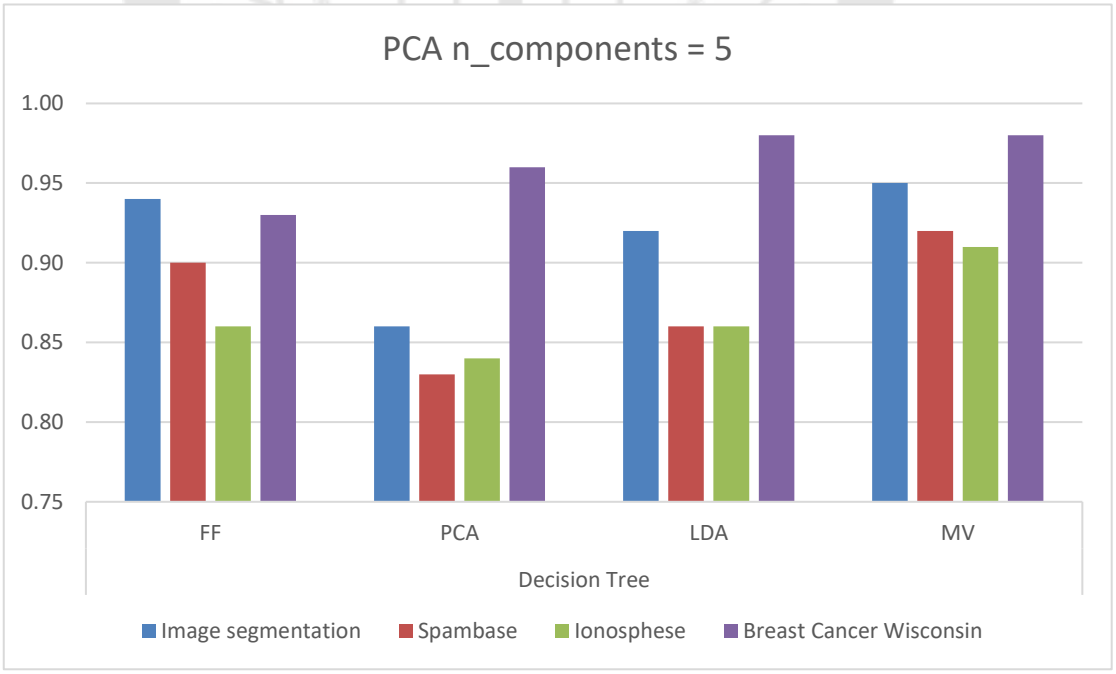
จากการจำแนกประเภทแบบหลายมุมมองในงานวิจัยนี้เพื่อเปรียบเทียบการจำแนกประเภทแบบหลายมุมมอง ผู้วิจัยได้ทำการทดลองโดยวิธีต่างๆ หลายๆ วิธี โดยนำเสนอวิธีการโดยการใช้ข้อมูลในหลายลักษณะ คือ 1. Full Feature 2. PCA 3. LDA และนำทั้ง 3 แบบมาทำ Multi-View (MV) โดยแต่ละวิธีจะได้ผลลัพธ์ที่แตกต่างกันออกไป และเมื่อได้ผลลัพธ์ของแต่ละวิธีก็นำมาเปรียบเทียบกับวิธี Multiple Model (MM) เพื่อหาผลลัพธ์ที่ดีที่สุด จากการทำการทดลอง การใช้ข้อมูลหลากหลายรูปแบบ Multi-View (MV) โดยใช้เทคนิคเดียว แล้วนำผลมาเปรียบเทียบกับวิธีดั้งเดิมที่เคยมีอยู่ คือวิธี Majority Vote Model ร่วมกับเทคนิคการเรียนรู้ของเครื่องกับการใช้อัลกอลิทึมในแบบต่างๆ ได้แก่ เทคนิคการถดถอยโลจิสติก (Logistic Regression), เทคนิคต้นไม้ตัดสินใจ (Decision Tree), เทคนิคการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-NN), Voting ตัวจำแนกประเภทโดยตั้งค่าแบบ Hard นั้น ทำให้ได้ค่าที่ดีที่สุด โดยได้สร้างตัวแยกแยะสำหรับข้อมูลแต่ละมุมมองด้วยเทคนิคต้นไม้ตัดสินใจ โดย Full Feature (FF) หมายถึง การนำข้อมูลเริ่มต้นมาใช้สร้างตัวแยกแยะ PCA หมายถึง การนำข้อมูลที่ได้จากการใช้เทคนิคการลดมิติด้วย PCA มาทำการสร้างตัวแยกแยะ และ LDA หมายถึง การนำข้อมูลที่ได้จากการใช้เทคนิคการลดมิติด้วย LDA มาใช้สร้างตัวแยกแยะ

โดยผู้วิจัยได้ทำการทดลองตามคำแนะนำของกรรมการภายนอกจากการสอบปากเปล่า โดยได้มีการเปลี่ยนค่าพารามิตเตอร์ของ PCA ที่ได้มีการทดลองไว้ จาก n_componant เท่ากับ 7 เป็น 5 เพื่อเปรียบเทียบผลที่ได้ดังนี้

ตาราง 7 ผลการทดสอบแบบ Decision Tree โดยเปลี่ยนค่าพารามิเตอร์ PCA n_componant เท่ากับ 5

DATA	Full Feature	PCA	LDA	Multi-View
Image Segmentation	0.94	0.86	0.92	0.95
Spambase	0.90	0.83	0.86	0.92
Ionosphese	0.86	0.84	0.86	0.91
Breast Cancer Wisconsin	0.93	0.96	0.98	0.98

ตาราง 7 แสดงให้เห็นค่าของผลการทดลองแบบ Decision Tree โดยเปลี่ยนค่าพารามิเตอร์ PCA n_componant เท่ากับ 5 พบว่าผลของการหาด้วย Multi-View นั้นในทุกข้อมูลเมื่อเปรียบเทียบกับเทคนิคอื่นๆ นั้นได้ค่าที่สูงที่สุดจะเป็นข้อมูลของ Breast Cancer Wisconsin ซึ่งมีค่าเท่ากับ 0.98 และลองลงมาจะเป็นข้อมูลของ Image Segmentation ที่มีค่าเท่ากับ 0.95



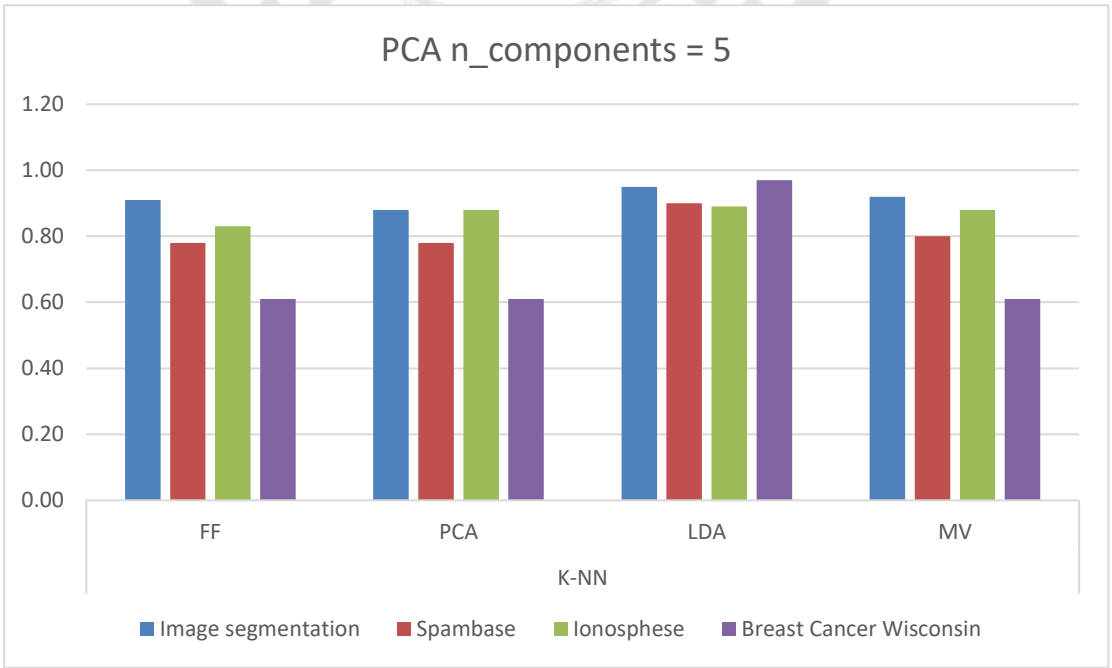
ภาพประกอบ 17 การทดสอบแบบ Decision Tree โดยกำหนดค่า PCA ให้ n_components เท่ากับ 5

ภาพประกอบ 17 พบหากต้องการให้ได้ผลลัพธ์สูงที่สุดคือเทคนิค Multi-View โดยค่าที่สูงที่สุดคือข้อมูลของ Breast Cancer Wisconsin โดยมีค่าเท่ากับ 0.98 และลองลงมาคือข้อมูลของ Image segmentation มีค่าเท่ากับ 0.95

ตาราง 8 ผลการทดสอบแบบ K-NN โดยเปลี่ยนค่าพารามิเตอร์ PCA n_componant เท่ากับ 5

DATA	Full Feature	PCA	LDA	Multi-View
Image Segmentation	0.91	0.88	0.95	0.92
Spambase	0.78	0.78	0.90	0.80
Ionosphese	0.83	0.88	0.89	0.88
Breast Cancer Wisconsin	0.61	0.61	0.97	0.61

ตาราง 8 แสดงให้เห็นค่าของผลการทดลองแบบ K-NN โดยเปลี่ยนค่าพารามิเตอร์ PCA n_componant เท่ากับ 5 จะเห็นได้ว่าแม้จะมีการเปลี่ยนพารามิเตอร์เพื่อทดสอบผลที่ได้แล้วแต่ค่าของ LDA ก็ยังชนะค่าที่ได้จาก ผลของ Multi-View จึงทำให้ผลที่ได้นั้นมีค่าต่ำกว่าการใช้ LDA อยู่โดยค่าที่สูงที่สุดคือข้อมูลBreast Cancer Wisconsin โดยมีค่าเท่ากับ 0.97 และลองลงมาคือ Image Segmentation โดยมีค่าเท่ากับ 0.95



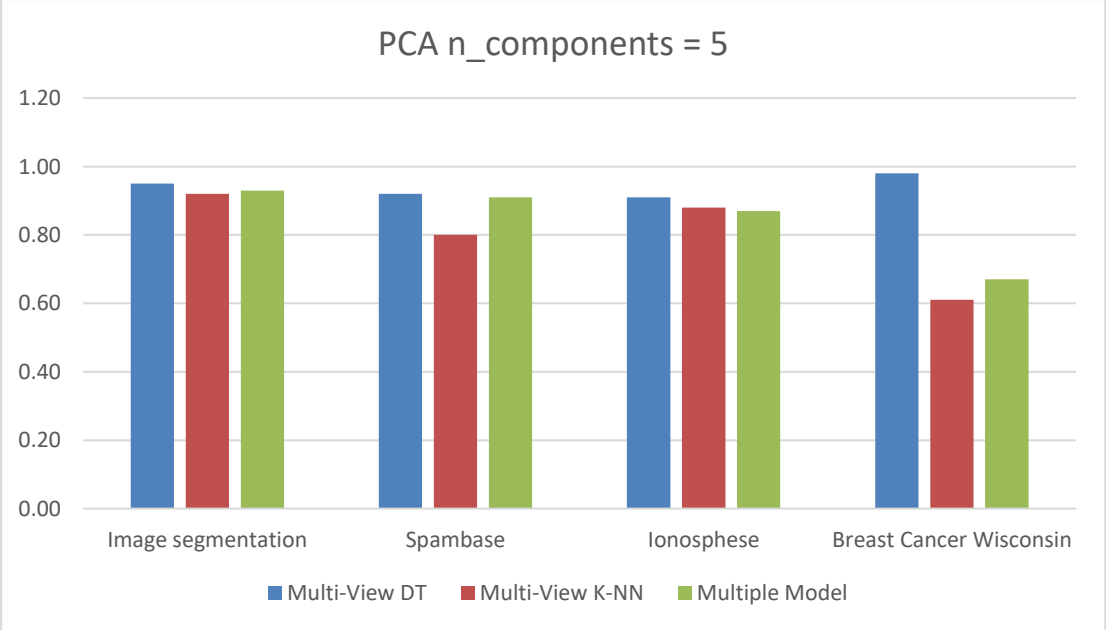
ภาพประกอบ 18 แสดงการทดสอบแบบ K-NN โดยกำหนดค่า PCA ให้ n_components เท่ากับ 5

ภาพประกอบ 18 จะพบว่าวิธีที่ได้ผลที่ดีที่สุดของเทคนิค K-NN คือวิธี LDA ซึ่งผลที่ได้นั้นมีค่าสูงในทุกข้อมูลที่นำมาทำการทดลองโดยข้อมูลที่มีผลลัพธ์สูงที่สุดคือ Breast Cancer Wisconsin ที่ได้ผลเท่ากับ 0.97 ซึ่งเมื่อเทียบกับ Multi-View นั้นได้เท่ากับ 0.61 ซึ่งต่างกันถึง 0.36

ตาราง 9 เปรียบเทียบ Multi View กับ Multiple Model โดยกำหนดค่า PCA ให้ n_components เท่ากับ 5

DATA	Multi View Decision Tree	Multi View KNN	Multiple Model
Image Segmentation	0.95	0.92	0.93
Spambase	0.92	0.80	0.91
Ionosphese	0.91	0.88	0.87
Breast Cancer Wisconsin	0.98	0.61	0.67

ตาราง 9 จะเห็นได้ว่าค่าของ Accuracy ที่มีการเปลี่ยนค่า n_components = 5 ของการใช้เทคนิคการแยกแยะแบบหลายมุมมอง โดยใช้ตัวแยกแยะ แบบ Decision Tree ให้ค่าความแม่นยำที่สูงที่สุด



ภาพประกอบ 19 การเปรียบเทียบ Multi-View ด้วยเทคนิค Decision Tree และ K-NN กับ Multiple Model โดยกำหนดค่า PCA n_componants เท่ากับ 5

ภาพประกอบ 19 จากภาพพบว่าการหาค่าแบบ Multi-View ด้วยเทคนิค Decision Tree ให้ผลลัพธ์สูงที่สุดในทุกข้อมูลนั้นได้ค่าที่สูงที่สุดโดยข้อมูลที่ได้ผลลัพธ์ที่สูงคือข้อมูลของ Breast Cancer Wisconsin ที่ได้ผลลัพธ์เท่ากับ 0.98 และลองลงมาคือ Image segmentation ที่ได้ผลลัพธ์เท่ากับ 0.95

บทที่ 5
สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ

งานวิจัยฉบับนี้ได้นำเสนอการศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมอง โดยผู้วิจัยใช้เทคนิคการเรียนรู้ของเครื่องในหลายวิธี และได้วัดผลของการศึกษาโดยใช้ค่า Accuracy โดยใช้วิธี Multi-View เพื่อหาค่าที่ดีที่สุด

จากผลการทดลองจึงสรุปได้ว่าการใช้การแยกแยะแบบหลายมุมมอง โดยใช้ตัวแยกแยะแบบต้นไม้ตัดสินใจ (Decision Tree) จะให้ค่าความแม่นยำ สูงกว่า การใช้เทคนิคแบบอื่น ไม่ว่าจะเป็นการแยกแยะแบบมุมมองเดียว หรือ การแยกแยะโดยใช้หลายแบบจำลอง แต่ทั้งนี้ การใช้เทคนิคการแยกแยะแบบหลายมุมมองนี้ ยังขึ้นกับชนิดของตัวแยกแยะที่นำมาใช้เป็นอย่างมาก เมื่อเราสร้างตัวแยกแยะการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-NN) จะได้ผลที่ต่ำกว่าการใช้ต้นไม้ตัดสินใจเป็นอย่างมาก ดังนั้น จึงต้องมีการศึกษาเพิ่มเติมถึงสาเหตุที่ทำให้ประสิทธิภาพการทำงานของทั้งสองกรณีมีความแตกต่างกันเช่นนี้ ต่อไป

5.1 สรุปผลการวิจัย

ในการวิจัยนี้ได้ใช้ข้อมูลทั้งหมด 4 ชุดด้วยกันคือ 1) Image segmentation 2) Spambase 3) Ionosphese 4) Breast Cancer Wisconsin จากเว็บไซต์ Data Sets - UCI Machine Learning Repository โดยนำข้อมูลมาทำการลดมิติของข้อมูลเป็น 3 ลักษณะ ด้วยกันคือ 1) Full feature 2) PCA 3) LDA และนำทั้ง 3 แบบมาทำ Multi-View เพื่อให้ได้ค่าที่ดีที่สุด โดยได้เลือกใช้อัลกอริทึมหลายวิธีด้วยกัน ได้แก่ 1. Decision Tree 2. K-NN เป็นต้น ซึ่งจากการทำการศึกษาพบว่าการทำ Multi-View ด้วยเทคนิค Decision Tree นั้นให้ค่าผลลัพธ์ที่ดีกว่าวิธีอื่นๆ เมื่อนำผลที่ได้มาทำการเปรียบเทียบกัน

5.2 อภิปรายผล

จากผลการทดลองจึงสรุปได้ว่าการใช้การแยกแยะแบบหลายมุมมอง โดยใช้ตัวแยกแยะแบบต้นไม้ตัดสินใจจะให้ค่าความแม่นยำ สูงกว่า การใช้เทคนิคแบบอื่น ไม่ว่าจะเป็นการแยกแยะแบบมุมมองเดียว หรือ การแยกแยะโดยใช้หลายแบบจำลอง แต่ทั้งนี้การใช้เทคนิคการแยกแยะแบบหลายมุมมองนี้ยังขึ้นกับชนิดของตัวแยกแยะที่นำมาใช้เป็นอย่างมาก เมื่อเราสร้างตัวแยกแยะการหาเพื่อนบ้านใกล้เคียงกันมากที่สุดจะได้ผลที่ต่ำกว่าการใช้ต้นไม้ตัดสินใจเป็นอย่างมาก

5.3 ข้อเสนอแนะ

ต้องมีการศึกษาเพิ่มเติมถึงสาเหตุที่ทำให้ประสิทธิภาพการทำงานของกรณีที่ใช้ตัวแยกแยะการหาเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-NN) ได้ผลที่ต่ำกว่าการใช้ต้นไม้ตัดสินใจ (Decision Tree) ถึงมีความแตกต่างกัน

บรรณานุกรม

Dua Dheeru and Graff, Casey. (2017). *Uci Machine Learning Repository.* จาก http://archive.ics.uci.edu/ml

Kumar Rajiv; และคนอื่น ๆ. (2012). *Unconstrained Handwritten Numeral Recognition Using Majority Voting Classifier.* IEEE. จาก

Müller Sarah Guido and Andreas. (2016). *Introduction to Machine Learning with Python.*

Zhao Jing; และคนอื่น ๆ. (2017). Multi-View Learning Overview: Recent Progress and New Challenges. *Information Fusion*. 38: 43-54.

เมืองแก้ว ธีร ยุทธ์. (2012). การ วิเคราะห์ องค์ประกอบ ของ แบบ ประเมิน ใน การ ประหยัด พลังงาน ไฟฟ้า.

กุล ลินิน อนันต. (2014). การ ประยุกต์ ใช้ เหมือง ข้อมูล เพื่อ การ ทำนาย สถานภาพ ของ นักศึกษา วิทยาลัย เทคโนโลยี ภาค ใต้. การ ประชุม วิชาการ ระดับ นานาชาติ และ ระดับ ชาติ วิศว ศึกษา. 1(1): 25-40.

ทองคำ ภูริพัทร์. (2559). อัลกอริทึมแบบรวมสำหรับการเลือกคุณสมบัติของข้อมูล. วิทยานิพนธ์. มหาวิทยาลัยธรรมศาสตร์.

นพมาศปักเข็มและคณะ ข. (2560). การจำแนกประเภทภูมิปัญญาท้องถิ่นของไทยแบบอัตโนมัติโดยวิธีการทางเหมืองข้อมูล. มหาวิทยาลัยทักษิณ. 20: 300-307.

ปรเมษฐ์ ธันวานนท์ ชัยกร ยิ่งเสร, และ วรพล พงษ์เพ็ชร. การประยุกต์ใช้โมเดลการเรียนรู้แบบรวมกลุ่มเพื่อพยากรณ์แนวโน้มของราคาหลักทรัพย์ในตลาดหลักทรัพย์แห่งประเทศไทย. *Journal of Information Science and Technology*. 7: 12-21.

พรหม สุ พจน์ เฮง พระ. กลุ่ม ก้อน ตัว จำแนก ประเภท กำหนดการ พันธุกรรม สำหรับ ข้อมูล ไมโครอาร์เรย์. จุฬาลงกรณ์ มหาวิทยาลัย.

พิลึก สุเมธ. (2559). การวิเคราะห์องค์ประกอบสมรรถภาพทางวิชาชีพของผู้สำเร็จการศึกษาสาขาวิชาคอมพิวเตอร์ธุรกิจตามความต้องการของตลาดแรงงาน. 96-109.

พุ่มลำเจียก ธนพล. (2559). การรู้จำอาจมณ์ใบหน้าจากวีดีโอ โดยใช้ตัวกรองกาบอร์ วิธีการวิเคราะห์องค์ประกอบหลักและการวิเคราะห์จำแนกประเภทเชิงเส้น. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

วิทวัส วิทยา ไกร เลิศ. การ วิเคราะห์ ภาพ ใบหน้า 3 มิติ สำหรับ การ ระบุ ตัว ตน และ ยืนยัน ตัว บุคคล.

สัจ เดช ธรรม ศิริพยุง มี. (2011). การ จำแนก ข้อมูล ด้วย วิธี แบบ ร่วม กัน ตัดสินใจ จาก พื้นฐาน ของ เทคนิค ต้นไม้ ตัดสินใจ เทคนิค โครง ข่าย ประสาท เทียม และ เทคนิค ซัพพอร์ต เวก เตอร์ แมชชีน ร่วม กับ การ เลือก ตัวแทน ที่ เหมาะสม ด้วย ขั้น ตอน วิธี เชิง พันธุกรรม. *Journal of King Mongkut's University of Technology North Bangkok*. 21(2): 293-303.

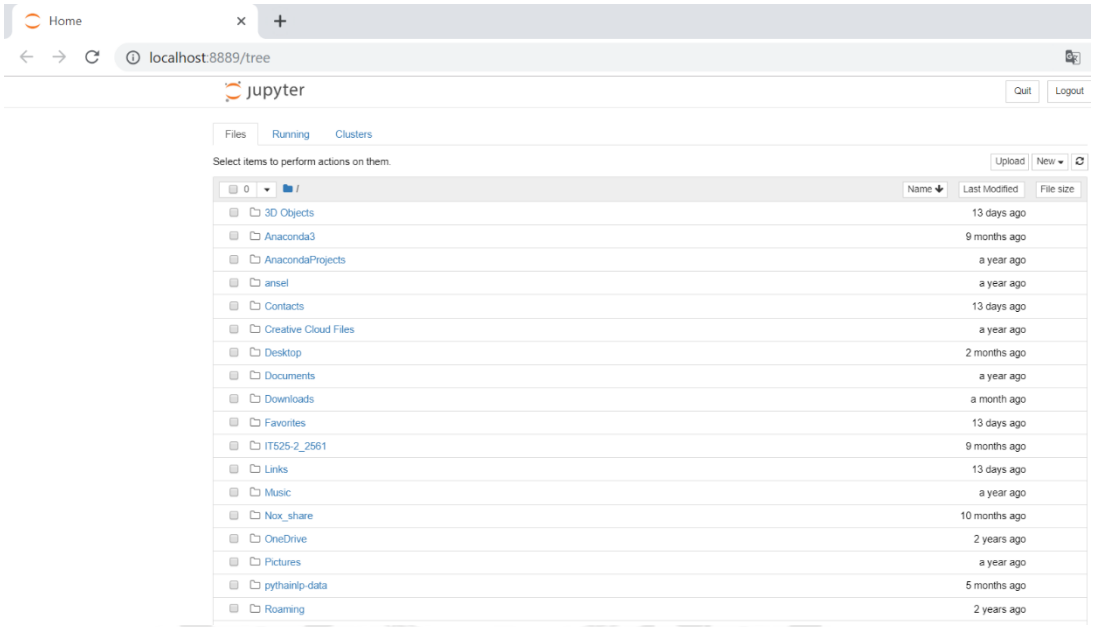
สินสมบูรณ์ทอง สุรวัชร ศรีเปารยะ และสายชล. (2560). การเปรียบเทียบประสิทธิภาพวิธีการจำแนกกลุ่มการเป็นโรคไตเรื้อรัง : กรณีศึกษาโรงพยาบาลแห่งหนึ่งในประเทศอินเดีย. วารสารวิทยาศาสตร์และเทคโนโลยี. (Vol.25 No.5 (September - October 2017)): 839-853.

ภาคผนวก

ในงานวิจัยเรื่องการศึกษาเทคนิคจำแนกประเภทแบบหลายมุมมองนี้ได้ใช้ภาษา Python เวอร์ชั่น 3.6.7 บนโปรแกรม Jupyter Notebook ของ Anaconda โดยมีรายละเอียดดังนี้

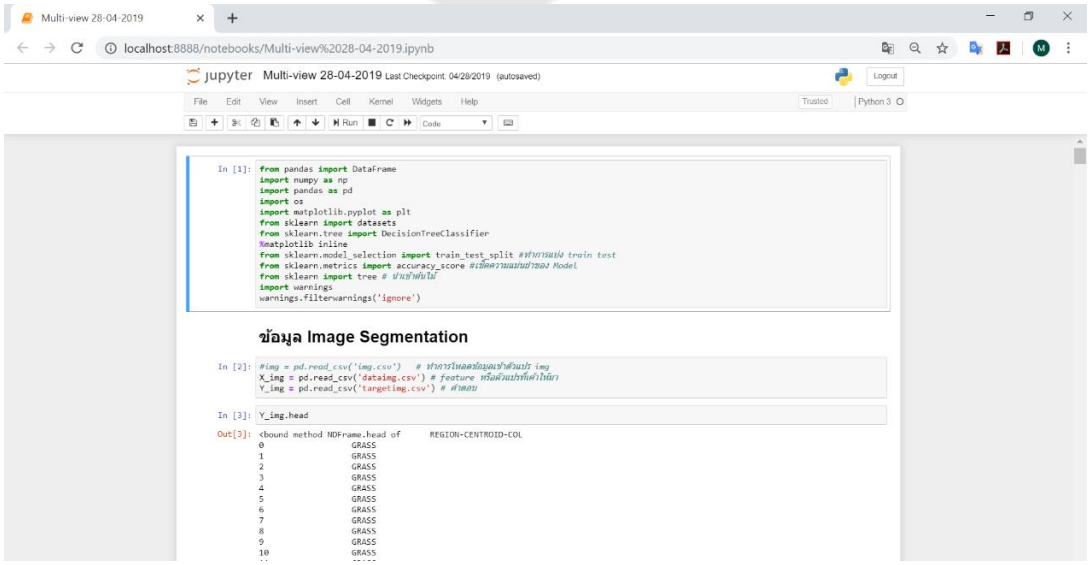
ภาพประกอบ 20 รายละเอียดโปรแกรม Jupyter notebook

เมื่อเปิดโปรแกรมเข้ามาจะแสดงหน้าต่างโปรแกรมขึ้นมาดังนี้



ภาพประกอบ 21 หน้าโปรแกรม Jupyter Notebook

ตัวอย่างการทำงานของโปรแกรม Jupyter Notebook เมื่อทำการรันตามแสดงดังนี้



ภาพประกอบ 22 ตัวอย่างการทำงานของโปรแกรม Jupyter Notebook

โค้ด Import Library ของโปรแกรม

```
In [1]: from pandas import DataFrame
        import numpy as np
        import pandas as pd
        import os
        import matplotlib.pyplot as plt
        from sklearn import datasets
        from sklearn.tree import DecisionTreeClassifier
        %matplotlib inline
        from sklearn.model_selection import train_test_split #ทำการแบ่ง train test
        from sklearn.metrics import accuracy_score #เช็คความแม่นยำของ Model
        from sklearn import tree # นำเข้าต้นไม้
        import warnings
        warnings.filterwarnings('ignore')
```

ข้อมูล Image Segmentation

```
In [2]: #img = pd.read_csv('img.csv')   # ทำการโหลดข้อมูลเข้าตัวแปร img
        X_img = pd.read_csv('dataimg.csv') # feature หรือตัวแปรที่เค้าให้มา
        Y_img = pd.read_csv('targetimg.csv') # คำตอบ
```

```
In [3]: Y_img.head
```

```
Out[3]: <bound method NDFrame.head of      REGION-CENTROID-COL
        0                  GRASS
        1                  GRASS
        2                  GRASS
        3                  GRASS
        4                  GRASS
        5                  GRASS
        6                  GRASS
        7                  GRASS
        8                  GRASS
        9                  GRASS
        10                 GRASS
        11                 GRASS
        12                 GRASS
```

ภาพประกอบ 23 โค้ด Import Library ที่ใช้ในโปรแกรม

โค้ดดูจำนวน Data และจำนวน Feature

```
In [4]: X_img.tail()# 19 feature 2099 data
```

Out[4]:

	REGION-CENTROID-ROW	REGION-PIXEL-COUNT	SHORT-LINE-DENSITY-5	SHORT-LINE-DENSITY-2	VEDGE-MEAN	VEDGE-SD	HEDGE-MEAN	HEDGE-SD	INTENSITY-MEAN	RAWRED-MEAN	RAWBLUE-MEAN	RAWGREEN-MEAN	EXRED-MEAN	EXBLUE-MEA
2095	32	158	9	0.000000	0.0	0.944445	0.862963	0.833333	0.611111	7.962963	6.333334	11.888889	5.666666	-4.88888
2096	8	162	9	0.111111	0.0	1.611111	2.062962	0.333333	0.133333	8.370370	6.666666	12.000000	6.444445	-5.11111
2097	128	161	9	0.000000	0.0	0.555555	0.251852	0.777778	0.162963	7.148148	5.555555	10.888889	5.000000	-4.77777
2098	150	158	9	0.000000	0.0	2.166667	1.633334	1.388889	0.418518	8.444445	7.000000	12.222222	6.111111	-4.33333
2099	124	162	9	0.111111	0.0	1.388889	1.129630	2.000000	0.888889	10.037037	8.000000	14.555555	7.555555	-6.11111

```
In [5]: X_img.head()
```

Out[5]:

	REGION-CENTROID-ROW	REGION-PIXEL-COUNT	SHORT-LINE-DENSITY-5	SHORT-LINE-DENSITY-2	VEDGE-MEAN	VEDGE-SD	HEDGE-MEAN	HEDGE-SD	INTENSITY-MEAN	RAWRED-MEAN	RAWBLUE-MEAN	RAWGREEN-MEAN	EXRED-MEAN	EXBLUE-MEAN
0	110	189	9	0.0	0.0	1.000000	0.666667	1.222222	1.186342	12.925926	10.888889	9.222222	18.666668	-6.111111
1	86	187	9	0.0	0.0	1.111111	0.720082	1.444444	0.750309	13.740741	11.666667	10.333334	19.222221	-6.222222
2	225	244	9	0.0	0.0	3.388889	2.195113	3.000000	1.520234	12.259259	10.333334	9.333334	17.111110	-5.777778
3	47	232	9	0.0	0.0	1.277778	1.254621	1.000000	0.894427	12.703704	11.000000	9.000000	18.111110	-5.111111
4	97	186	9	0.0	0.0	1.166667	0.691215	1.166667	1.005540	15.592592	13.888889	11.777778	21.111110	-5.111111

ภาพประกอบ 24 โค้ดดูจำนวน Data และจำนวน Feature

โค้ดการสร้าง Multi-view เทคนิค Decision Tree และการทำ PCA สำหรับ Image segmentation

สร้าง multiple view จากการใช้เครื่องมือ PCA and LDA แทนที่จะเป็นการแบ่ง feature ตามปกติ

In [6]:
```
#Added by sirisup
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
pca = PCA(n_components=7) #ต้องเขียนว่าเอากี่ class
X_view1 = pca.fit(X_img).transform(X_img)
lda = LinearDiscriminantAnalysis(n_components=6)  # class-1
X_view2 = lda.fit(X_img, Y_img.iloc[:,0]).transform(X_img)
```

view 1 obtained from PCA view 2 obtained from LDA view 3 use full features data

In [7]:
```
X_train,X_test,Y_train,Y_test = train_test_split(X_img,Y_img,test_size = 0.3, random_state = 5)
trn_index=Y_train.index.tolist()
tst_index = Y_test.index.tolist()
X_trnv1=X_view1[trn_index,:]
X_tstv1=X_view1[tst_index,:]
X_trnv2=X_view2[trn_index,:]
X_tstv2=X_view2[tst_index,:]
```

PCA DT Img

In [8]:
```
estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
estimator.fit(X_trnv1,Y_train) #ทำการ train Model
pred1 = estimator.predict(X_tstv1) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred1)
print(Acc)
diff_v1=list()
for ind in range(len(pred1)):
    if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
print(diff_v1)
```

```
0.9142857142857143
[23, 29, 41, 46, 47, 51, 58, 78, 87, 117, 118, 143, 144, 160, 161, 164, 187, 219, 222, 236, 239, 243, 253, 258, 270, 286, 299,
333, 335, 339, 355, 359, 368, 383, 396, 397, 418, 433, 469, 502, 516, 521, 546, 550, 579, 582, 583, 586, 588, 589, 597, 598, 59
9, 621]
```

ภาพประกอบ 25 โค้ดการสร้าง Multi-view

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree และทำ LDA และFull Feature สำหรับ Image segmentation

LDA DT Img

In [9]:
```
estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
estimator.fit(X_trnv2,Y_train) #ทำการ train Model
pred2 = estimator.predict(X_tstv2) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred2)
print(Acc)
diff_v2=list()
for ind in range(len(pred2)):
    if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
print(diff_v2)
```

```
0.9238095238095239
[13, 25, 41, 46, 48, 61, 78, 83, 96, 97, 99, 117, 119, 125, 164, 181, 187, 191, 198, 201, 229, 239, 249, 284, 293, 299, 316, 32
2, 348, 353, 355, 387, 398, 401, 433, 441, 444, 464, 465, 469, 496, 514, 546, 550, 563, 580, 589, 599]
```

Full Feature DT Img

In [10]:
```
estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
estimator.fit(X_train,Y_train) #ทำการ train Model
predf = estimator.predict(X_test) #ทำการทำนายผล
Acc = accuracy_score(Y_test,predf)
print(Acc)
diff_vf=list()
for ind in range(len(predf)):
    if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
print(diff_vf)
```

```
0.9412698412698413
[19, 25, 35, 51, 63, 87, 90, 117, 128, 142, 143, 144, 196, 198, 219, 226, 249, 258, 299, 322, 347, 353, 355, 401, 433, 458, 46
0, 462, 464, 468, 469, 496, 546, 579, 588, 589, 594]
```

ภาพประกอบ 26 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree และทำ LDA และFull Feature สำหรับ Image segmentation

โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับ Image segmentation

Multi-view DT Img

```
In [11]: #i=594
#print(Y_test.iloc[i,0])    # ค่าที่ถูกต้อง
#print(predf[i],pred1[i],pred2[i])  # ค่าที่ได้จากการใช้แต่ละ view: predf=full features, pred1=PCA, pred2=LDA
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า คำตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(pf) #ให้นำค่าใน total_dat ไปเช็คในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.953968253968254
```

ภาพประกอบ 27 โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับ Image segmentation

โค้ด การสร้าง Multi-view ด้วยเทคนิค K-NN วิธี PCA และ LDA สำหรับ Image segmentation

KNN Img

PCA KNN Img

```
In [12]: from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier(n_neighbors = 5)#ทำการเรียกใช้ KNN
KNN.fit(X_trnv1,Y_train) #ทำการ train Model
pred1 = KNN.predict(X_tstv1) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred1)
print(Acc)
diff_v1=list()
for ind in range(len(pred1)):
    if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
print(diff_v1)
```

```
0.9158730158730158
[1, 13, 17, 23, 25, 32, 48, 51, 58, 78, 96, 117, 125, 143, 164, 166, 181, 187, 196, 201, 213, 217, 219, 249, 254, 258, 267, 28
0, 286, 299, 322, 333, 339, 355, 359, 363, 376, 384, 400, 464, 469, 514, 516, 546, 550, 557, 563, 579, 589, 597, 599, 601, 621]
```

LDA KNN Img

```
In [13]: KNN = KNeighborsClassifier(n_neighbors = 5) #ทำการเรียกใช้ KNN
KNN.fit(X_trnv2,Y_train) #ทำการ train Model
pred2 = KNN.predict(X_tstv2) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred2)
print(Acc)
diff_v2=list()
for ind in range(len(pred2)):
    if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
print(diff_v2)
```

```
0.9523809523809523
[13, 25, 78, 96, 99, 117, 125, 144, 161, 166, 177, 181, 196, 198, 201, 219, 280, 322, 343, 353, 355, 384, 398, 401, 433, 469, 5
50, 594, 599, 606]
```

ภาพประกอบ 28 โค้ด Multi-view เทคนิค K-NN ด้วยวิธี PCA และ LDA สำหรับ Image segmentation

โค้ด Multi-view เทคนิค K-NN ด้วยวิธี Full Feature และ Multiple view สำหรับ Image segmentation

Full Feature KNN Img ¶

In [14]:
```
KNN = KNeighborsClassifier(n_neighbors = 5)
KNN.fit(X_train,Y_train) #ทำการ train Model
predf = KNN.predict(X_test) #ทำการทำนายผล
Acc = accuracy_score(Y_test,predf)
print(Acc)
diff_vf=list()
for ind in range(len(predf)):
    if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
print(diff_vf)
```

```
0.9158730158730158
[1, 13, 17, 23, 25, 32, 48, 51, 58, 78, 96, 117, 125, 143, 164, 166, 181, 187, 196, 201, 213, 217, 219, 249, 254, 258, 267, 28
0, 286, 299, 322, 333, 339, 355, 359, 363, 376, 384, 400, 464, 469, 514, 516, 546, 550, 557, 563, 579, 589, 597, 599, 601, 621]
```

Multi-view KNN Img|

In [15]:
```
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า ค่าตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(p2) #ให้นำค่าใน total_dat ไปเช็คในตัวแปร p2 เพราะว่าได้ค่ามากที่สุด
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.9158730158730158
```

ภาพประกอบ 29 โค้ด Multi-view เทคนิค K-NN ด้วยวิธี Full Feature และ Multiple view สำหรับ Image segmentation

โค้ด Multiple Model สำหรับ Image segmentation

Multiple Model Img

In [16]:
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier

clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',random_state=5)
clf2 = tree.DecisionTreeClassifier(random_state = 5)
clf3 = KNeighborsClassifier(n_neighbors = 5)

eclf = VotingClassifier(estimators=[('lr', clf1), ('dt', clf2), ('KNN', clf3)], voting='hard')

for clf, label in zip([clf1, clf2, clf3, eclf], ['Logistic Regression', 'DecisionTree', 'KNeighbors', 'Ensemble']):
    scores = cross_val_score(clf, X_img, Y_img, cv=5, scoring='accuracy')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))
```

```
Accuracy: 0.89 (+/- 0.05) [Logistic Regression]
Accuracy: 0.93 (+/- 0.06) [DecisionTree]
Accuracy: 0.90 (+/- 0.04) [KNeighbors]
Accuracy: 0.93 (+/- 0.04) [Ensemble]
```

ภาพประกอบ 30 โค้ด Multiple Model สำหรับ Image segmentation

โค้ด Import Data Spambase และการ Import Library ต่างๆ

Import Data Spambase

```
In [17]: X_spam = pd.read_csv('dataspam.csv') # feature หรือตัวแปรที่เค้าให้มา
         Y_spam = pd.read_csv('targetspam.csv') # คำตอบ
```

```
In [18]: X_spam.tail()# 57 feature 4599 data
```

Out[18]:

	0	0.64	64.1	0.1	0.32	0.2	0.3	0.4	0.5	0.6	...	0.39	0.40	0.41	0.42	0.778	0.43	0.44	3.756	61	278
4595	0.31	0.0	0.62	0.0	0.00	0.31	0.0	0.0	0.0	0.0	...	0.0	0.000	0.232	0.0	0.000	0.0	0.0	1.142	3	88
4596	0.00	0.0	0.00	0.0	0.00	0.00	0.0	0.0	0.0	0.0	...	0.0	0.000	0.000	0.0	0.353	0.0	0.0	1.555	4	14
4597	0.30	0.0	0.30	0.0	0.00	0.00	0.0	0.0	0.0	0.0	...	0.0	0.102	0.718	0.0	0.000	0.0	0.0	1.404	6	118
4598	0.96	0.0	0.00	0.0	0.32	0.00	0.0	0.0	0.0	0.0	...	0.0	0.000	0.057	0.0	0.000	0.0	0.0	1.147	5	78
4599	0.00	0.0	0.65	0.0	0.00	0.00	0.0	0.0	0.0	0.0	...	0.0	0.000	0.000	0.0	0.125	0.0	0.0	1.250	5	40

5 rows × 57 columns

```
In [19]: from sklearn.decomposition import PCA
         from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
         pca = PCA(n_components=7) #ต้องเขียนว่าเอากี่ class
         X_spam_view1 = pca.fit(X_spam).transform(X_spam)
         lda = LinearDiscriminantAnalysis(n_components=6)  # class-1
         X_spam_view2 = lda.fit(X_spam, Y_spam.iloc[:,0]).transform(X_spam)
```

```
In [20]: X_train,X_test,Y_train,Y_test = train_test_split(X_spam,Y_spam,test_size = 0.3, random_state = 5)
         trn_index=Y_train.index.tolist()
         tst_index = Y_test.index.tolist()
         X_trnv1=X_spam_view1[trn_index,:]
         X_tstv1=X_spam_view1[tst_index,:]
         X_trnv2=X_spam_view2[trn_index,:]
         X_tstv2=X_spam_view2[tst_index,:]
```

ภาพประกอบ 31 โค้ด Import Data Spambase และการ Import Library ต่างๆ

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับ Spambase

PCA DT Spambase

```
In [21]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียนใช้ต้นไม้
         estimator.fit(X_trnv1,Y_train) #ทำการ train Model
         pred1 = estimator.predict(X_tstv1) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred1)
         print(Acc)
         diff_v1=list()
         for ind in range(len(pred1)):
             if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
         print(diff_v1)
```

```
0.8507246376811595
[15, 16, 20, 39, 50, 52, 54, 62, 68, 78, 81, 97, 101, 105, 112, 120, 122, 137, 141, 143, 163, 165, 174, 176, 183, 196, 208, 21
5, 216, 217, 218, 221, 229, 231, 233, 246, 263, 264, 277, 278, 281, 287, 293, 298, 303, 308, 311, 321, 322, 323, 324, 331, 333,
334, 336, 339, 345, 355, 362, 364, 391, 395, 401, 409, 424, 432, 458, 461, 469, 471, 474, 487, 492, 507, 512, 522, 524, 533, 53
6, 547, 555, 573, 578, 579, 582, 589, 594, 600, 603, 610, 616, 619, 621, 622, 623, 628, 634, 635, 638, 646, 660, 675, 681, 692,
695, 714, 717, 720, 721, 727, 730, 736, 737, 740, 742, 750, 763, 772, 790, 791, 794, 798, 804, 807, 812, 823, 830, 836, 839, 84
6, 856, 880, 881, 886, 888, 891, 899, 901, 904, 907, 912, 922, 956, 964, 965, 988, 995, 1005, 1006, 1013, 1014, 1028, 1030, 103
5, 1038, 1045, 1060, 1065, 1066, 1070, 1071, 1074, 1075, 1078, 1087, 1095, 1098, 1107, 1110, 1111, 1113, 1116, 1127, 1133, 114
5, 1147, 1160, 1176, 1182, 1186, 1204, 1211, 1213, 1220, 1226, 1229, 1231, 1237, 1248, 1265, 1267, 1268, 1277, 1278, 1280, 129
8, 1312, 1316, 1321, 1324, 1347, 1351, 1358, 1364, 1365, 1371]
```

ภาพประกอบ 32 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับ Spambase

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับ Spambase

LDA DT Spambase

In [22]:
```
estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
estimator.fit(X_trnv2,Y_train) #ทำการ train Model
pred2 = estimator.predict(X_tstv2) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred2)
print(Acc)
diff_v2=list()
for ind in range(len(pred2)):
    if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
print(diff_v2)
```

```
0.8659420289855072
[12, 18, 41, 49, 50, 76, 95, 100, 103, 106, 107, 112, 121, 122, 130, 132, 138, 142, 143, 150, 172, 178, 180, 191, 197, 207, 20
8, 218, 224, 230, 231, 246, 254, 261, 278, 293, 300, 305, 308, 320, 324, 326, 331, 334, 342, 343, 358, 367, 395, 399, 406, 408,
417, 432, 454, 455, 458, 474, 479, 480, 483, 486, 495, 501, 517, 524, 525, 533, 578, 579, 589, 596, 603, 610, 616, 621, 626, 62
8, 638, 643, 651, 654, 659, 677, 683, 686, 688, 694, 695, 709, 717, 718, 720, 739, 740, 741, 748, 750, 763, 771, 772, 776, 777,
780, 781, 783, 786, 791, 798, 803, 804, 809, 812, 817, 830, 836, 845, 853, 860, 867, 869, 872, 887, 906, 917, 934, 947, 963, 96
8, 994, 1001, 1022, 1023, 1034, 1038, 1046, 1051, 1054, 1060, 1064, 1065, 1074, 1078, 1081, 1086, 1095, 1119, 1130, 1131, 1136,
1139, 1147, 1148, 1152, 1159, 1171, 1179, 1180, 1182, 1186, 1190, 1192, 1211, 1219, 1248, 1256, 1259, 1260, 1265, 1267, 1271, 1
275, 1277, 1286, 1302, 1304, 1314, 1324, 1330, 1334, 1340, 1347, 1356, 1363, 1370]
```

ภาพประกอบ 33 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับ Spambase

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับ Spambase

Full Feature DT Spambase

In [23]:
```
estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
estimator.fit(X_train,Y_train) #ทำการ train Model
predf = estimator.predict(X_test) #ทำการทำนายผล
Acc = accuracy_score(Y_test,predf)
print(Acc)
diff_vf=list()
for ind in range(len(predf)):
    if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
print(diff_vf)
```

```
0.9043478260869565
[20, 49, 54, 92, 106, 120, 153, 160, 183, 208, 221, 240, 243, 253, 260, 282, 299, 305, 309, 320, 343, 348, 353, 374, 387, 391,
396, 424, 436, 439, 458, 460, 467, 477, 494, 505, 512, 517, 521, 536, 537, 538, 547, 587, 596, 603, 610, 616, 622, 623, 628, 63
4, 635, 638, 654, 656, 659, 672, 718, 720, 722, 730, 737, 740, 750, 771, 772, 778, 785, 791, 798, 802, 812, 813, 818, 836, 839,
861, 869, 877, 878, 880, 888, 899, 904, 917, 919, 922, 956, 958, 960, 984, 988, 990, 1005, 1008, 1027, 1043, 1046, 1048, 1059,
1060, 1070, 1078, 1081, 1087, 1099, 1110, 1115, 1118, 1139, 1186, 1192, 1211, 1222, 1225, 1243, 1247, 1253, 1255, 1264, 1265, 1
277, 1285, 1296, 1304, 1312, 1324, 1328, 1331, 1371, 1374]
```

ภาพประกอบ 34 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับ Spambase

โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับ Spambase

Multi-view DT Spambase

In [24]:
```
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า คำตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(pf) #ให้นำค่าใน total_dat ไปเช็คในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.9231884057971015
```

ภาพประกอบ 35 โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับ Spambase

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับ Spambase

KNN Spambase

PCA KNN Spambase

In [25]:
```
from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier(n_neighbors = 5) # 2.คัดแกนข้อมูลที่มีน้อยๆออกไป  1.บีบแกนไปในทิศที่ข้อมูลกระจายมากสุด
KNN.fit(X_trnv1,Y_train) #ทำการ train Model
pred1 = KNN.predict(X_tstv1) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred1)
print(Acc)
diff_v1=list()
for ind in range(len(pred1)):
    if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
print(diff_v1)
```

```
0.7753623188405797
[4, 6, 8, 10, 20, 22, 24, 26, 32, 39, 44, 46, 50, 52, 57, 73, 83, 88, 92, 93, 96, 105, 106, 121, 125, 126, 129, 130, 138, 139,
140, 142, 144, 147, 148, 151, 153, 155, 157, 160, 161, 165, 167, 175, 196, 202, 208, 209, 214, 216, 217, 218, 221, 225, 228, 22
9, 230, 231, 243, 247, 260, 264, 270, 273, 287, 290, 292, 296, 297, 298, 299, 303, 310, 311, 323, 330, 331, 335, 336, 338, 343,
354, 356, 358, 361, 366, 370, 371, 373, 378, 380, 387, 388, 391, 395, 396, 405, 409, 410, 411, 419, 421, 424, 429, 432, 433, 43
8, 444, 447, 448, 453, 458, 460, 469, 472, 477, 483, 503, 512, 514, 520, 522, 559, 564, 566, 568, 576, 578, 579, 585, 587, 597,
603, 608, 610, 616, 621, 634, 635, 641, 643, 646, 647, 654, 660, 663, 666, 672, 674, 678, 691, 692, 694, 700, 714, 717, 718, 72
0, 721, 723, 727, 728, 730, 736, 740, 743, 749, 750, 754, 755, 763, 766, 772, 776, 781, 783, 784, 791, 794, 801, 806, 812, 813,
830, 832, 835, 836, 839, 844, 846, 852, 857, 860, 868, 877, 878, 880, 881, 888, 891, 899, 901, 906, 907, 919, 921, 922, 923, 93
4, 935, 936, 938, 940, 945, 952, 954, 958, 965, 967, 977, 980, 985, 987, 999, 1005, 1008, 1018, 1019, 1024, 1027, 1037, 1038, 1
043, 1046, 1062, 1064, 1067, 1069, 1070, 1074, 1075, 1078, 1079, 1081, 1083, 1084, 1091, 1095, 1098, 1110, 1113, 1116, 1119, 11
20, 1127, 1131, 1140, 1145, 1147, 1160, 1167, 1171, 1172, 1179, 1182, 1186, 1192, 1196, 1209, 1212, 1213, 1216, 1218, 1220, 122
8, 1232, 1243, 1247, 1248, 1253, 1259, 1262, 1265, 1267, 1269, 1274, 1276, 1277, 1280, 1287, 1292, 1293, 1299, 1312, 1316, 132
1, 1324, 1327, 1328, 1330, 1335, 1336, 1349, 1351, 1353, 1358, 1371, 1372, 1374, 1378]
```

ภาพประกอบ 36 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับ Spambase

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับ Spambase

LDA KNN Spambase

In [26]:
```
KNN = KNeighborsClassifier(n_neighbors = 5) #
KNN.fit(X_trnv2,Y_train) #ทำการ train Model
pred2 = KNN.predict(X_tstv2) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred2)
print(Acc)
diff_v2=list()
for ind in range(len(pred2)):
    if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
print(diff_v2)
```

```
0.9065217391304348
[20, 41, 49, 50, 51, 76, 106, 121, 129, 130, 137, 138, 142, 143, 161, 178, 204, 207, 208, 218, 221, 224, 231, 233, 293, 294, 30
0, 305, 310, 320, 321, 326, 331, 334, 343, 358, 367, 395, 406, 417, 432, 454, 458, 475, 483, 486, 501, 525, 561, 579, 596, 603,
610, 616, 620, 621, 628, 635, 638, 651, 654, 659, 668, 677, 695, 696, 718, 720, 737, 739, 748, 750, 754, 763, 771, 772, 775, 77
6, 777, 783, 786, 790, 791, 798, 804, 812, 817, 830, 836, 853, 855, 865, 869, 872, 879, 911, 917, 940, 988, 1022, 1023, 1038, 1
046, 1060, 1074, 1078, 1081, 1086, 1095, 1139, 1147, 1159, 1179, 1186, 1192, 1209, 1211, 1260, 1277, 1286, 1304, 1309, 1315, 13
24, 1328, 1331, 1340, 1349, 1364]
```

ภาพประกอบ 37 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับ Spambase

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับ Spambase

Full feature KNN Spambase

In [27]:
```
KNN = KNeighborsClassifier(n_neighbors = 5) #
KNN.fit(X_train,Y_train) #ทำการ train Model
predf = KNN.predict(X_test) #ทำการทำนายผล
Acc = accuracy_score(Y_test,predf)
print(Acc)
diff_vf=list()
for ind in range(len(predf)):
    if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
print(diff_vf)
```

```
0.7876811594202898
[4, 6, 8, 10, 20, 22, 24, 32, 39, 44, 46, 50, 52, 57, 67, 73, 83, 88, 92, 93, 105, 106, 121, 129, 130, 136, 138, 139, 142, 144,
147, 148, 151, 153, 157, 160, 165, 167, 175, 202, 208, 209, 214, 217, 218, 221, 222, 225, 229, 231, 243, 247, 259, 264, 270, 27
3, 287, 290, 292, 296, 297, 298, 299, 303, 310, 323, 331, 335, 336, 338, 343, 354, 358, 360, 361, 366, 370, 371, 378, 380, 387,
388, 391, 396, 405, 409, 410, 411, 419, 421, 424, 429, 432, 433, 438, 444, 447, 453, 458, 460, 469, 472, 477, 483, 507, 512, 51
4, 520, 522, 559, 564, 566, 568, 576, 579, 581, 603, 608, 610, 616, 619, 634, 635, 638, 641, 643, 646, 647, 654, 659, 660, 663,
666, 672, 674, 675, 678, 691, 692, 694, 700, 714, 717, 718, 720, 727, 728, 730, 736, 740, 743, 749, 750, 754, 755, 763, 766, 77
2, 776, 781, 782, 783, 784, 791, 804, 806, 812, 813, 830, 832, 835, 836, 844, 846, 852, 857, 860, 868, 877, 878, 881, 888, 891,
894, 899, 906, 907, 921, 922, 923, 934, 935, 936, 938, 940, 945, 952, 954, 958, 965, 967, 970, 978, 980, 987, 1005, 1008, 1018,
1019, 1024, 1037, 1043, 1046, 1062, 1064, 1066, 1067, 1070, 1074, 1075, 1078, 1079, 1081, 1083, 1084, 1091, 1095, 1098, 1107, 1
110, 1113, 1116, 1119, 1120, 1127, 1131, 1139, 1140, 1145, 1147, 1160, 1164, 1166, 1168, 1171, 1179, 1180, 1182, 1186, 1192, 11
96, 1209, 1212, 1213, 1216, 1218, 1220, 1228, 1232, 1237, 1243, 1247, 1248, 1253, 1259, 1262, 1265, 1267, 1274, 1276, 1277, 128
0, 1292, 1293, 1299, 1304, 1312, 1323, 1324, 1327, 1328, 1330, 1334, 1335, 1336, 1349, 1351, 1353, 1358, 1371, 1372, 1374, 137
8]
```

ภาพประกอบ 38 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับ Spambase

โค้ด Multi-view ด้วยเทคนิค K-NNสำหรับ Spambase

Multi view KNN Spambase

In [28]:
```
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า tatal_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า คำตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(p2) #ให้นำค่าใน total_dat ไปเช็คในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.8007246376811594
```

ภาพประกอบ 39 โค้ด Multi-view ด้วยเทคนิค K-NN สำหรับ Spambase

โค้ด Multiple Model สำหรับ Spambase

Multiple vote Model Spambase

In [29]:
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier

clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',random_state=5)
clf2 = tree.DecisionTreeClassifier(random_state = 5)
clf3 = KNeighborsClassifier(n_neighbors = 5)

eclf = VotingClassifier(estimators=[('lr', clf1), ('dt', clf2), ('KNN', clf3)], voting='hard')

for clf, label in zip([clf1, clf2, clf3, eclf], ['Logistic Regression', 'DecisionTree', 'KNeighbors', 'Ensemble']):
    scores = cross_val_score(clf, X_spam, Y_spam, cv=5, scoring='accuracy')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))
```

```
Accuracy: 0.90 (+/- 0.04) [Logistic Regression]
Accuracy: 0.88 (+/- 0.06) [DecisionTree]
Accuracy: 0.77 (+/- 0.04) [KNeighbors]
Accuracy: 0.91 (+/- 0.04) [Ensemble]
```

ภาพประกอบ 40 โค้ด Multiple Model สำหรับ Spambase

โค้ด Import Data Ionosphese และดู จำนวน Data ทั้งหมด

Ionosphese

```
In [30]: X_iono = pd.read_csv('dataiono.csv') # feature หรือตัวแปรที่เค้าให้มา
         Y_iono = pd.read_csv('targetiono.csv') # คำตอบ

In [31]: X_iono.tail()# 34 feature 349 data
```

Out[31]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.0376	...	0.56811	-0.51171	0.41078	-0.46168	0.21266	-0.3409	0.42267	-0.5
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	0.95378	-0.04202	0.83479	0.00123	1.00000	0.12815	0.86660	-0.1
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.94520	0.01361	0.93522	0.04925	0.93159	0.08168	0.94066	-0.0
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.93988	0.03193	0.92489	0.02542	0.92120	0.02242	0.92459	0.0
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	0.91050	-0.02099	0.89147	-0.07760	0.82983	-0.17238	0.96022	-0.0
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	0.86467	-0.15114	0.81147	-0.04822	0.78207	-0.00703	0.75747	-0.0

5 rows × 34 columns

ภาพประกอบ 41 โค้ด Import Data Ionosphese และดู จำนวน Data ทั้งหมด

โค้ดดูประเภทของข้อมูล Ionosphese

```
In [32]: X_iono.dtypes

Out[32]: 1             int64
         0             int64
         0.99539     float64
         -0.05889    float64
         0.85243     float64
         0.02306     float64
         0.83398     float64
         -0.37708    float64
         1.1         float64
         0.0376      float64
         0.85243.1   float64
         -0.17755    float64
         0.59755     float64
         -0.44945    float64
         0.60536     float64
         -0.38223    float64
         0.84356     float64
         -0.38542    float64
         0.58212     float64
         -0.32192    float64
         0.56971     float64
         -0.29674    float64
         0.36946     float64
         -0.47357    float64
         0.56811     float64
         -0.51171    float64
         0.41078     float64
         -0.46168    float64
         0.21266     float64
         -0.3409     float64
         0.42267     float64
         -0.54487    float64
         0.18641     float64
         -0.453      float64
         dtype: object
```

ภาพประกอบ 42 โค้ดดูประเภทของข้อมูล Ionosphese

Import Library ที่ใช้ และกำหนดค่าพารามิเตอร์ต่างๆ สำหรับข้อมูล Ionosphese

```
In [34]: from sklearn.decomposition import PCA
         from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
         pca = PCA(n_components=7) #ต้องเขียนว่าเอากี่ class
         X_iono_view1 = pca.fit(X_iono).transform(X_iono)
         lda = LinearDiscriminantAnalysis(n_components=6)  # class-1
         X_iono_view2 = lda.fit(X_iono, Y_iono.iloc[:,0]).transform(X_iono)

In [35]: X_train,X_test,Y_train,Y_test = train_test_split(X_iono,Y_iono,test_size = 0.3, random_state = 5)
         trn_index=Y_train.index.tolist()
         tst_index = Y_test.index.tolist()
         X_trnv1=X_iono_view1[trn_index,:]
         X_tstv1=X_iono_view1[tst_index,:]
         X_trnv2=X_iono_view2[trn_index,:]
         X_tstv2=X_iono_view2[tst_index,:]
```

ภาพประกอบ 43 Import Library ที่ใช้ และกำหนดค่าพารามิเตอร์ต่างๆ สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับข้อมูล Ionosphese

PCA DT Ionosphese

```
In [36]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
         estimator.fit(X_trnv1,Y_train) #ทำการ train Model
         pred1 = estimator.predict(X_tstv1) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred1)
         print(Acc)
         diff_v1=list()
         for ind in range(len(pred1)):
             if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
         print(diff_v1)

         0.8666666666666667
         [30, 34, 35, 37, 41, 42, 44, 47, 49, 50, 61, 64, 66, 101]
```

ภาพประกอบ 44 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับข้อมูล Ionosphese

LDA DT Ionosphese ¶

```
In [37]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
         estimator.fit(X_trnv2,Y_train) #ทำการ train Model
         pred2 = estimator.predict(X_tstv2) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred2)
         print(Acc)
         diff_v2=list()
         for ind in range(len(pred2)):
             if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
         print(diff_v2)

         0.8666666666666667
         [10, 32, 33, 41, 44, 48, 55, 63, 66, 68, 84, 85, 91, 98]
```

ภาพประกอบ 45 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับข้อมูล Ionosphese

FULL Feature DT Ionosphese

```
In [38]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
         estimator.fit(X_train,Y_train) #ทำการ train Model
         predf = estimator.predict(X_test) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,predf)
         print(Acc)
         diff_vf=list()
         for ind in range(len(predf)):
             if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
         print(diff_vf)

         0.8666666666666667
         [0, 11, 29, 35, 42, 47, 49, 63, 69, 83, 86, 95, 98, 100]
```

ภาพประกอบ 46 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับข้อมูล Ionosphese

โค้ด Multi-viewด้วยเทคนิค Decision Tree สำหรับข้อมูล Ionosphese

Multi-view DT Ionosphese

```
In [39]: pred_agg = list() # สร้างlist ว่างๆ ชื่นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
         for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
             total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
             if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า คำตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
                 pred_agg.append(pf) #ให้นำค่าใน total_dat ไปเซ็ตในตัวแปร pf เพราะว่าเป็น Full feature
             else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
                 count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
                 for ttl in total_dat:
                     count_result[ttl]=[pf,p1,p2].count(ttl)
                 pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาค่าตอบที่เป็น majority ไปใส่เป็น ค่าตอบของ

         Acc = accuracy_score(Y_test,pred_agg)
         print(Acc)

         0.9142857142857143
```

ภาพประกอบ 47 โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับข้อมูล Ionosphese

KNN Ionosphese

PCA KNN Ionosphese

In [40]:
```
from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier(n_neighbors = 5)
KNN.fit(X_trnv1,Y_train) #ทำการ train Model
pred1 = KNN.predict(X_tstv1) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred1)
print(Acc)
diff_v1=list()
for ind in range(len(pred1)):
    if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
print(diff_v1)
```

```
0.8761904761904762
[23, 41, 42, 47, 61, 62, 63, 66, 69, 84, 85, 98, 101]
```

ภาพประกอบ 48 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับข้อมูล Ionosphese

LDA KNN Ionosphese

In [41]:
```
KNN = KNeighborsClassifier(n_neighbors = 5)
KNN.fit(X_trnv2,Y_train) #ทำการ train Model
pred2 = KNN.predict(X_tstv2) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred2)
print(Acc)
diff_v2=list()
for ind in range(len(pred2)):
    if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
print(diff_v2)
```

```
0.8952380952380953
[14, 32, 41, 44, 55, 63, 66, 68, 84, 91, 98]
```

ภาพประกอบ 49 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับข้อมูล Ionosphese

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับข้อมูล Ionosphese

Full feature KNN Ionosphese

```
In [42]: KNN = KNeighborsClassifier(n_neighbors = 5)
KNN.fit(X_train,Y_train) #ทำการ train Model
predf = KNN.predict(X_test) #ทำการทำนายผล
Acc = accuracy_score(Y_test,predf)
print(Acc)
diff_vf=list()
for ind in range(len(predf)):
    if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
print(diff_vf)
```

```
0.8380952380952381
[0, 3, 23, 30, 41, 42, 43, 47, 61, 62, 63, 66, 69, 84, 91, 98, 101]
```

ภาพประกอบ 50 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับข้อมูล Ionosphese

โค้ด Multi-view ด้วยเทคนิค K-NN สำหรับข้อมูล Ionosphese

Multi-view KNN Ionosphese

```
In [43]: pred_agg = list() # สร้างList ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_aggไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า คำตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(pf) #ให้นำค่าใน total_datไปเช็คในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.8761904761904762
```

ภาพประกอบ 51 โค้ด Multi-view ด้วยเทคนิค K-NN สำหรับข้อมูล Ionosphese

โค้ด Multiple Model สำหรับข้อมูล Ionosphese

Multiple Model Ionosphese

```
In [44]: from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier

clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',random_state=5)
clf2 = tree.DecisionTreeClassifier(random_state = 5)
clf3 = KNeighborsClassifier(n_neighbors = 5)

eclf = VotingClassifier(estimators=[('lr', clf1), ('dt', clf2), ('KNN', clf3)], voting='hard')

for clf, label in zip([clf1, clf2, clf3, eclf], ['Logistic Regression', 'DecisionTree', 'KNeighbors', 'Ensemble']):
    scores = cross_val_score(clf, X_iono, Y_iono, cv=5, scoring='accuracy')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))
```

```
Accuracy: 0.86 (+/- 0.04) [Logistic Regression]
Accuracy: 0.86 (+/- 0.07) [DecisionTree]
Accuracy: 0.83 (+/- 0.04) [KNeighbors]
Accuracy: 0.87 (+/- 0.05) [Ensemble]
```

ภาพประกอบ 52 โค้ด Multiple Model สำหรับข้อมูล Ionosphese

โค้ด Import Dataset Import library ต่างๆ และกำหนดพารามิเตอร์ สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Breast Cancer Wisconsin (Original) Data Set

```
In [45]: X_wdbc = pd.read_csv('datawdbc.csv') # feature หรือตัวแปรที่เค้าให้มา
         Y_wdbc = pd.read_csv('targetwdbc.csv') # คำตอบ

In [46]: X_wdbc.tail()# 10 feature 697 data
```

Out[46]:

	1000025	5	1	1.1	1.2	2	1.3	3	1.4	1.5
677	776715	3	1	1	1	3	2	1	1	1
678	841769	2	1	1	1	2	1	1	1	1
679	888820	5	10	10	3	7	3	8	10	2
680	897471	4	8	6	4	3	4	10	6	1
681	897471	4	8	8	5	4	5	10	4	1

```
In [47]: from sklearn.decomposition import PCA
         from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
         pca = PCA(n_components=7) #ต้องเขียนว่าเอากี่ class
         X_wdbc_view1 = pca.fit(X_wdbc).transform(X_wdbc)
         lda = LinearDiscriminantAnalysis(n_components=6)  # class-1
         X_wdbc_view2 = lda.fit(X_wdbc, Y_wdbc.iloc[:,0]).transform(X_wdbc)

In [48]: X_train,X_test,Y_train,Y_test = train_test_split(X_wdbc,Y_wdbc,test_size = 0.3, random_state = 5)
         trn_index=Y_train.index.tolist()
         tst_index = Y_test.index.tolist()
         X_trnv1=X_wdbc_view1[trn_index,:]
         X_tstv1=X_wdbc_view1[tst_index,:]
         X_trnv2=X_wdbc_view2[trn_index,:]
         X_tstv2=X_wdbc_view2[tst_index,:]
```

ภาพประกอบ 53 โค้ด Import Dataset Import library ต่างๆ และกำหนดพารามิเตอร์ สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

PCA DT Breast Cancer Wisconsin

```
In [49]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
         estimator.fit(X_trnv1,Y_train) #ทำการ train Model
         pred1 = estimator.predict(X_tstv1) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred1)
         print(Acc)
         diff_v1=list()
         for ind in range(len(pred1)):
             if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
         print(diff_v1)

         0.9658536585365853
         [42, 56, 111, 155, 156, 182, 195]
```

ภาพประกอบ 54 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย PCA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

LDA DT Breast Cancer Wisconsin

```
In [50]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียกใช้ต้นไม้
         estimator.fit(X_trnv2,Y_train) #ทำการ train Model
         pred2 = estimator.predict(X_tstv2) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred2)
         print(Acc)
         diff_v2=list()
         for ind in range(len(pred2)):
             if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
         print(diff_v2)

         0.9853658536585366
         [6, 93, 156]
```

ภาพประกอบ 55 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย LDA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Full feature DT Breast Cancer Wisconsin

```
In [51]: estimator = tree.DecisionTreeClassifier(random_state = 5) #ทำการเรียนใช้ต้นไม้
         estimator.fit(X_train,Y_train) #ทำการ train Model
         predf = estimator.predict(X_test) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,predf)
         print(Acc)
         diff_vf=list()
         for ind in range(len(predf)):
             if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
         print(diff_vf)

         0.9365853658536586
         [42, 54, 61, 105, 128, 137, 153, 155, 156, 159, 174, 200, 203]
```

ภาพประกอบ 56 โค้ดการสร้าง Multi-view ด้วยเทคนิค Decision Tree โดย Full Feature สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Multi-view DT Breast Cancer Wisconsin

In [52]:
```
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแค่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า tatal_dat มีค่าเท่ากับ 3 ถ้าเท่ากับ 3 แสดงว่า ค่าตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(pf) #ให้นำค่าใน total_dat ไปเข็ดในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าค่าตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก ค่าตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางค่าตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาค่าตอบที่เป็น majority ไปใส่เป็น ค่าตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.9853658536585366
```

ภาพประกอบ 57 โค้ด Multi-view ด้วยเทคนิค Decision Tree สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

KNN Breast Cancer Wisconsin |

PCA KNN Breast Cancer Wisconsin

In [53]:
```
from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier(n_neighbors = 5) #
KNN.fit(X_trnv1,Y_train) #ทำการ train Model
pred1 = KNN.predict(X_tstv1) #ทำการทำนายผล
Acc = accuracy_score(Y_test,pred1)
print(Acc)
diff_v1=list()
for ind in range(len(pred1)):
    if pred1[ind]!=Y_test.iloc[ind,0]: diff_v1.append(ind)
print(diff_v1)
```

```
0.6195121951219512
[1, 4, 6, 8, 10, 13, 15, 22, 23, 29, 31, 33, 36, 38, 39, 40, 41, 42, 45, 50, 52, 54, 57, 61, 62, 65, 68, 72, 74, 85, 89, 91, 9
2, 93, 95, 99, 101, 102, 103, 104, 108, 111, 114, 118, 119, 120, 121, 124, 127, 129, 134, 138, 139, 142, 146, 150, 151, 152, 15
6, 158, 161, 162, 164, 165, 167, 168, 175, 177, 178, 182, 183, 184, 186, 188, 190, 192, 202, 203]
```

ภาพประกอบ 58 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย PCA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

LDA KNN Breast Cancer Wisconsin

```
In [54]: KNN = KNeighborsClassifier(n_neighbors = 5) #
         KNN.fit(X_trnv2,Y_train) #ทำการ train Model
         pred2 = KNN.predict(X_tstv2) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,pred2)
         print(Acc)
         diff_v2=list()
         for ind in range(len(pred2)):
             if pred2[ind]!=Y_test.iloc[ind,0]: diff_v2.append(ind)
         print(diff_v2)

         0.975609756097561
         [42, 137, 153, 155, 182]
```

ภาพประกอบ 59 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย LDA สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Full feature DT Breast Cancer Wisconsin

```
In [55]: KNN = KNeighborsClassifier(n_neighbors = 5) #
         KNN.fit(X_train,Y_train) #ทำการ train Model
         predf = KNN.predict(X_test) #ทำการทำนายผล
         Acc = accuracy_score(Y_test,predf)
         print(Acc)
         diff_vf=list()
         for ind in range(len(predf)):
             if predf[ind]!=Y_test.iloc[ind,0]: diff_vf.append(ind)
         print(diff_vf)

         0.6195121951219512
         [1, 4, 6, 8, 10, 13, 15, 22, 23, 29, 31, 33, 36, 38, 39, 40, 41, 42, 45, 50, 52, 54, 57, 61, 62, 65, 68, 72, 74, 85, 89, 91, 9
         2, 93, 95, 99, 101, 102, 103, 104, 108, 111, 114, 118, 119, 120, 121, 124, 127, 129, 134, 138, 139, 142, 146, 150, 151, 152, 15
         6, 158, 161, 162, 164, 165, 167, 168, 175, 177, 178, 182, 183, 184, 186, 188, 190, 192, 202, 203]
```

ภาพประกอบ 60 โค้ดการสร้าง Multi-view ด้วยเทคนิค K-NN โดย Full Feature สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ด Multi-view ด้วยเทคนิค K-NN สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Multi-view KNN Breast Cancer Wisconcin

In [56]:
```
pred_agg = list() # สร้างlist ว่างๆ ขึ้นมา โดยใช้ ตัวแปร pred_agg ไว้เก็บค่า
for pf,p1,p2 in zip(predf,pred1,pred2):# สร้างเงื่อนไขในการทำงานโดยการรวมค่าที่หาได้ 3 ค่ามาอยู่ใน zip
    total_dat = set([pf,p1,p2])  # ใช้ set เพื่อหาว่าค่าไหนซ้ำกันแล้วให้เขียนแต่ค่านั้น
    if len(total_dat)==3:  #สร้างเงื่อนไขถ้า total_dat มีค่าเท่ากับ 3 แสดงว่า ค่าตอบของทั้ง สาม classifiers ไม่เท่ากันเลย
        pred_agg.append(pf) #ให้นำค่าใน total_dat ไปเข้าในตัวแปร pf เพราะว่าเป็น Full feature
    else:#ถ้าคำตอบ ของทั้งสาม ไม่เท่ากัน เราจะเลือก คำตอบของ full feature เพราะมันได้ค่า accuracy สูงสุด นั่นคือ เอาค่า pf ใส่เข้าไปใน pred_agg
        count_result=dict()# หากความยาวน้อยกว่าสาม แสดงว่ามีบางคำตอบ มีคนตอบซ้ำ จะมี majority vote ละ เราจึงหาตัวที่มากสุด
        for ttl in total_dat:
            count_result[ttl]=[pf,p1,p2].count(ttl)
        pred_agg.append(max(count_result,key=count_result.get))#ต่อด้วยคำสั่ง max ได้แล้ว จากนั้นเอาคำตอบที่เป็น majority ไปใส่เป็น คำตอบของ

Acc = accuracy_score(Y_test,pred_agg)
print(Acc)
```

```
0.6195121951219512
```

ภาพประกอบ 61 โค้ด Multi-view ด้วยเทคนิค K-NN สำหรับข้อมูล Breast Cancer Wisconsin (Original)

โค้ด Multiple Model สำหรับข้อมูล Breast Cancer Wisconsin (Original)

Multiple Model Breast Cancer Wisconcin

In [57]:
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier

clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',random_state=5)
clf2 = tree.DecisionTreeClassifier(random_state = 5)
clf3 = KNeighborsClassifier(n_neighbors = 5)

eclf = VotingClassifier(estimators=[('lr', clf1), ('dt', clf2), ('KNN', clf3)], voting='hard')

for clf, label in zip([clf1, clf2, clf3, eclf], ['Logistic Regression', 'DecisionTree', 'KNeighbors', 'Ensemble']):
    scores = cross_val_score(clf, X_wdbc, Y_wdbc, cv=5, scoring='accuracy')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))
```

```
Accuracy: 0.65 (+/- 0.00) [Logistic Regression]
Accuracy: 0.93 (+/- 0.02) [DecisionTree]
Accuracy: 0.53 (+/- 0.06) [KNeighbors]
Accuracy: 0.67 (+/- 0.04) [Ensemble]
```

ภาพประกอบ 62 โค้ด Multiple Model สำหรับข้อมูล Breast Cancer Wisconsin (Original)

ประวัติผู้เขียน

ชื่อ-สกุล	เปมิกา บุญเสริมส่ง
วัน เดือน ปี เกิด	8 พฤศจิกายน 2533
สถานที่เกิด	กรุงเทพฯ
วุฒิการศึกษา	บริหารธุรกิจบัณฑิต สาขาคอมพิวเตอร์ จาก วิทยาลัยเทคโนโลยีสยาม