การพัฒนาแบบจำลองการเรียนรู้ของเครื่องเพื่อจำแนกการสูญเสียลูกค้าในธุรกิจการจัดการทรัพยากรมนุษย์

DEVELOPING MACHINE LEARNING MODELS FOR CLASSIFYING CUSTOMER CHURN IN THE HUMAN RESOURCE MANAGEMENT INDUSTRY

ญาตา ชัยยารังกิจรัตน์

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2567

การพัฒนาแบบจำลองการเรียนรู้ของเครื่องเพื่อจำแนกการสูญเสียลูกค้าในธุรกิจการจัดการทรัพยากรมนุษย์

ญาตา ชัยยารังกิจรัตน์

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2567

DEVELOPING MACHINE LEARNING MODELS FOR CLASSIFYING CUSTOMER CHURN IN THE HUMAN RESOURCE MANAGEMENT INDUSTRY

YATA CHAIYARUNGKIJRAT

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

for the Degree of MASTER OF SCIENCE

(Data Science)

Faculty of Science, Srinakharinwirot University

2024

สารนิพนธ์

เรื่อง

การพัฒนาแบบจำลองการเรียนรู้ของเครื่องเพื่อจำแนกการสูญเสียลูกค้าในธุรกิจการจัดการทรัพยากรมนุษย์

ของ

ญาตา ชัยยารังกิจรัตน์

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

.....

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณบดีบัณฑิตวิทยาลัย

.....

คณะกรรมการสอบปากเปล่าสารนิพนธ์

..... ที่ปรึกษาหลัก

(ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์)

..... ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.จันตรี ผลประเสริฐ)

..... กรรมการ

(อาจารย์ ดร.เรืองศักดิ์ ตระกูลพุทธิรักษ์)

ชื่อเรื่อง	การพัฒนาแบบจำลองการเรียนรู้ของเครื่องเพื่อจำแนกการสูญเสียลูกค้าในธุรกิจการจัดการทรัพยากรมนุษย์
ผู้วิจัย	ญาตา ชัยยารังกิจรัตน์
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2567
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. วราภรณ์ วิยานนท์

การศึกษานี้มีวัตถุประสงค์เพื่อพัฒนาแบบจำลองการเรียนรู้ของเครื่อง (Machine Learning) สำหรับการพยากรณ์การสูญเสียลูกค้า (Customer Churn) ในบริบทของบริษัทผู้ให้บริการแพลตฟอร์มการจัดการทรัพยากรบุคคลแบบ B2B โดยใช้ข้อมูลลูกค้าจริงจำนวน 1,597 รายการ ซึ่งประกอบด้วยคุณลักษณะด้านพฤติกรรมการใช้งาน การชำระเงิน ความพึงพอใจ และข้อมูลทั่วไป งานวิจัยเริ่มจากการเตรียมข้อมูลและตรวจสอบความสมดุลของกลุ่มเป้าหมาย ก่อนเปรียบเทียบประสิทธิภาพของแบบจำลอง 7 แบบ ได้แก่ Decision Tree, Random Forest, Support Vector Machine (SVM), Gradient Boosting, Logistic Regression, K-Nearest Neighbors (KNN) และ AdaBoost จากผลการทดลอง พบว่า AdaBoost, Decision Tree และ Gradient Boosting มีประสิทธิภาพสูงสุด จึงถูกนำมาปรับแต่งค่าพารามิเตอร์ด้วยเทคนิค Grid Search Cross-Validation และสร้างแบบจำลองรวม (Ensemble Learning) ด้วยวิธี Soft Voting ผลการประเมินด้วย Accuracy, Precision, Recall, F1-score และ Cross-Validation Score แสดงให้เห็นว่าแบบจำลองที่ผ่านการปรับแต่งและรวมกันมีประสิทธิภาพสูงในการจำแนก โดยเฉพาะกลุ่มลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการ การวิเคราะห์ความสำคัญของฟีเจอร์โดยใช้ค่าเฉลี่ยจากโมเดลทั้งสามชี้ว่า พฤติกรรมด้านการชำระเงิน ได้แก่ ระยะเวลาหลังการชำระเงินล่าสุด และจำนวนใบแจ้งหนี้ที่ค้างชำระ เป็นตัวแปรที่มีอิทธิพลต่อการพยากรณ์มากที่สุดตามลำดับ ผลการศึกษาสนับสนุนว่าการรวมเทคนิคการเรียนรู้ของเครื่องกับการปรับจูนและรวมแบบจำลองสามารถเพิ่มประสิทธิภาพของการพยากรณ์การสูญเสียลูกค้า และมีศักยภาพในการนำไปประยุกต์ใช้เชิงกลยุทธ์ในธุรกิจ B2B

คำสำคัญ : การยกเลิกการใช้บริการของลูกค้า, การเรียนรู้ของเครื่อง, การจำแนกข้อมูล, การปรับแต่งพารามิเตอร์, การคาดการณ์พฤติกรรมลูกค้า, แพลตฟอร์มการจัดการทรัพยากรบุคคล, แบบจำลองรวม

Title	DEVELOPING MACHINE LEARNING MODELS FOR CLASSIFYING CUSTOMER CHURN IN THE HUMAN RESOURCE MANAGEMENT INDUSTRY
Author	YATA CHAIYARUNGKIJRAT
Degree	MASTER OF SCIENCE
Academic Year	2024
Thesis Advisor	Assistant Professor Dr. Waraporn Viyanon

This study aims to develop a machine learning model for predicting customer churn in the context of a B2B human resource management platform. The research is based on a real-world dataset comprising 1,597 customer records, which include behavioral data, payment history, customer satisfaction, and general demographic information. The analytical process began with data preprocessing and class imbalance assessment, followed by a performance comparison of seven machine learning algorithms: Decision Tree, Random Forest, Support Vector Machine (SVM), Gradient Boosting, Logistic Regression, K-Nearest Neighbors (KNN), and AdaBoost. The experimental results identified AdaBoost, Decision Tree, and Gradient Boosting as the top-performing models. These models were subsequently fine-tuned using Grid Search with Cross-Validation and then integrated into an ensemble model using the Soft Voting technique to enhance classification performance. Evaluation metrics—including Accuracy, Precision, Recall, F1-score, and Cross-Validation Score—demonstrated that the tuned ensemble model achieved superior performance, particularly in identifying customers at high risk of churn. Feature importance analysis, based on the average contributions from the three selected models, indicated that payment behavior features—specifically the time since the last payment and the number of overdue invoices—were the most influential predictors. The findings support the conclusion that combining machine learning with hyperparameter tuning and ensemble methods significantly improves the accuracy of customer churn prediction and offers promising potential for strategic applications in B2B business environments.

Keyword : Customer Churn, Machine Learning, Classification, Ensemble Model, Hyperparameter Tuning, Customer Behavior Prediction, Human Resource Management Platform

กิตติกรรมประกาศ

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์ อาจารย์ที่ปรึกษาโครงการ สำหรับความช่วยเหลือในทุก ๆ ด้านของการทำโครงการศึกษา และยังคอยให้คำปรึกษา คำแนะนำ และตรวจสอบแก้ไขเนื้อหาโครงการมาตลอด ทำให้โครงการศึกษานี้สำเร็จลุล่วง ข้าพเจ้าขอกราบขอบพระคุณอย่างสูงไว้ ณ ที่นี้

ขอขอบพระคุณ อาจารย์สาขาวิชาวิทยาการข้อมูล (วท.ม.) ทุกท่านที่ให้คำแนะนำ และช่วยเหลือข้าพเจ้าที่กำลังศึกษาอยู่ให้มีองค์ความรู้ทางด้านวิทยาศาสตร์จนทำให้สามารถทำโครงการนี้ได้สำเร็จ

ขอขอบพระคุณเจ้าของงานวิจัย บทความและข้อมูลต่างๆที่ใช้ในการศึกษาครั้งนี้ ที่ทำให้สามารถรวมกลายเป็นโครงการฉบับนี้ซึ่งอาจจะเป็นประโยชน์ต่อผู้ค้นคว้าต่อๆไปในอนาคต

ขอขอบคุณเพื่อนๆ ในภาควิชา สำหรับการแบ่งปันข้อมูลคำแนะนำ และช่วยกันแก้ปัญหาที่พบ ทำให้สามารถโครงการนี้สำเร็จลุล่วงได้

สุดท้ายนี้ข้าพเจ้าหวังว่าโครงงานนี้จะเป็นประโยชน์ต่อผู้ที่สนใจจะศึกษาต่อไป

ญาตา ชัยยารังกิจรัตน์

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

ปัจจุบันการแข่งขันทางธุรกิจสูงขึ้นอย่างต่อเนื่องและความคาดหวังของลูกค้าเปลี่ยนแปลงอย่างรวดเร็ว ความสามารถในการรักษาฐานลูกค้าอย่างยั่งยืนกลายเป็นหัวใจสำคัญของความสำเร็จ งานวิจัยนี้นำเสนอการประยุกต์ใช้ Machine Learning เพื่อคาดการณ์การสูญเสียลูกค้า (Customer Churn) โดยใช้ข้อมูลจากธุรกิจแพลตฟอร์มการจัดการทรัพยากรบุคคลในรูปแบบ B2B ช่วยให้สามารถเข้าใจพฤติกรรมลูกค้าเชิงลึก วางกลยุทธ์เชิงรุก และเพิ่มประสิทธิภาพในการรักษาลูกค้า

1.1 ความสำคัญและความเป็นมาของงานวิจัย

Customer Churn หรือการยกเลิกการใช้บริการ สะท้อนถึงความไม่พึงพอใจ ความต้องการที่ไม่ได้รับการตอบสนอง หรือการมีทางเลือกที่ดีกว่าในตลาด ปรากฏการณ์นี้ส่งผลโดยตรงต่อรายได้และความมั่นคงขององค์กร โดยเฉพาะในธุรกิจที่ต้องพึ่งพารายได้จากลูกค้าระยะยาว หากไม่สามารถรักษาฐานลูกค้าเดิมไว้ได้ ธุรกิจอาจต้องใช้ต้นทุนสูงในการหาลูกค้าใหม่มาทดแทน ซึ่งไม่เพียงเพิ่มค่าใช้จ่ายแต่ยังส่งผลต่อการเติบโตในระยะยาว

จากข้อมูลของบริษัทผู้ให้บริการแพลตฟอร์มการจัดการทรัพยากรบุคคล (HR Management Platform Provider) ซึ่งดำเนินธุรกิจในรูปแบบระหว่างองค์กร (B2B) พบว่าอัตราการยกเลิกการใช้บริการของลูกค้า (Customer Churn Rate) อยู่ที่ 6.18 % ต่อเดือน แต่การรักษาลูกค้ารายบุคคลเป็นเรื่องท้าทาย เนื่องจากบริษัทส่วนใหญ่มักมีลูกค้าจำนวนมาก ทำให้ไม่สามารถทุ่มเวลาและทรัพยากรเพื่อดูแลลูกค้าแต่ละรายได้โดยตรง เพราะอาจทำให้ต้นทุนสูงเกินไปจนไม่คุ้มกับรายได้ที่เพิ่มขึ้น

หากบริษัทสามารถคาดการณ์ได้อย่างแม่นยำว่าลูกค้ารายใดจะยกเลิกการใช้บริการก่อนกำหนด ก็จะสามารถมุ่งเน้นความพยายามในการรักษาลูกค้าเหล่านี้ได้โดยเฉพาะ การศึกษานี้มุ่งเน้นการสร้างโมเดล Machine Learning เพื่อวิเคราะห์และทำความเข้าใจพฤติกรรมการใช้งานรูปแบบการชำระเงิน และปัจจัยที่ส่งผลต่อการสูญเสียลูกค้า (Customer Churn) โดยเฉพาะ เป้าหมายหลักของโมเดลนี้คือการสามารถจำแนกการสูญเสียลูกค้าจากการยกเลิกการใช้บริการ ซึ่งจะช่วยให้บริษัทสามารถระบุและวางแผนกลยุทธ์การรักษาลูกค้าได้อย่างมีประสิทธิภาพยิ่งขึ้น

เป้าหมายสูงสุดของการสร้างโมเดล Machine Learning นี้คือการรักษาฐานลูกค้าในปัจจุบัน การใช้โมเดลช่วยให้ธุรกิจสามารถคาดการณ์พฤติกรรมของลูกค้าได้ล่วงหน้า ทำให้มี

โอกาสปรับตัวและตอบสนองได้อย่างรวดเร็วและตรงจุด ซึ่งนับเป็นปัจจัยสำคัญในการแข่งขันในตลาดที่มีการเปลี่ยนแปลงอย่างรวดเร็ว ความสำเร็จในตลาดที่แข่งขันสูงนั้นขึ้นอยู่กับการเข้าใจลูกค้าในเชิงลึก และการนำข้อมูลเชิงพฤติกรรมมาใช้ประโยชน์อย่างเต็มที่ เพื่อให้สามารถตอบโจทย์ความต้องการของลูกค้าได้อย่างแท้จริง

1.2 วัตถุประสงค์ของงานวิจัย

พัฒนาโมเดล Machine Learning ที่สามารถจำแนกการสูญเสียลูกค้าที่ยกเลิกการใช้บริการได้อย่างแม่นยำ

1.3 ขอบเขตของงานวิจัย

1.3.1กลุ่มตัวอย่าง

งานวิจัยจะมุ่งเน้นไปที่การศึกษาในกลุ่มลูกค้าของบริษัทผู้ให้บริการด้านการจัดการทรัพยากรบุคคล (HR management provider) เท่านั้น

1.3.2 ข้อมูลที่ใช้

ข้อมูลที่ใช้ในการศึกษานี้มีที่มาจากข้อมูลของบริษัทผู้ให้บริการแพลตฟอร์มการจัดการทรัพยากรบุคคล (HR Management Platform Provider) ซึ่งดำเนินธุรกิจในรูปแบบระหว่างองค์กร (B2B) ซึ่งมีเนื้อหาเกี่ยวกับด้านการใช้บริการ การชำระเงินและความพึงพอใจของลูกค้าจากการใช้บริการและจัดเก็บในฐานข้อมูลประเภท Relational Database Management System (RDBMS) ชุดข้อมูลประกอบด้วยข้อมูลจำนวนประมาณ 1,597 แถว ข้อมูลดังกล่าวครอบคลุมช่วงเวลาตั้งแต่ปี 2022 ถึงปี 2024 ซึ่งบันทึกข้อมูลของลูกค้าแต่ละราย รายละเอียดของข้อมูลประกอบด้วยทั้งหมด 18 คอลัมน์ ซึ่งแต่ละคอลัมน์มีความหมายและตัวแปรที่เกี่ยวข้อง ซึ่งจะอธิบายไว้ในตาราง 1

ตาราง 1 คำอธิบายชุดข้อมูล

ชื่อคอลัมน์	ชนิดของข้อมูล	คำอธิบาย	ตัวอย่างข้อมูล
ClientId	Integer	รหัสประจำตัวลูกค้าที่ถูกบันทึกในฐานข้อมูล	1003
CustAccount	Varchar	รหัสประจำตัวลูกค้าสำหรับแสดงส่วนลูกค้า	C1800445
Tenure	Integer	จำนวนปีที่ลูกค้าใช้บริการ	5
HasTaxId	Varchar	ลูกค้าได้กรอกเลขผู้เสียภาษีหรือไม่ (Yes/No)	Yes
HasEmail	Varchar	ลูกค้าได้กรอกอีเมลหรือไม่ (Yes/No)	Yes
HasPhoneNumber	Varchar	ลูกค้าได้กรอกเบอร์ติดต่อหรือไม่ (Yes/No)	Yes
PaidInvoiceCount	Integer	จำนวนใบแจ้งหนี้ที่มีสถานะชำระเงินแล้ว	108
OverdueInvoiceCount	Integer	จำนวนใบแจ้งหนี้ที่มีสถานะเลยกำหนดการชำระ	0
PendingInvoiceCount	Integer	จำนวนใบแจ้งหนี้ที่รอการชำระเงิน	3
DayFromLastPayment	Integer	จำนวนวันตั้งแต่การชำระเงินครั้งล่าสุด	45
PlanName	Varchar	แผนบริการที่ลูกค้าใช้ปัจจุบันหรือล่าสุด	Basic (Monthly)
IsChurn	Varchar	ลูกค้ายกเลิกการใช้บริการหรือไม่ (Yes/No)	No
Qty	Integer	จำนวนผู้ใช้งานของบริษัทลูกค้า	100
Contract	Varchar	ประเภทสัญญา	1 Month
TotalIncome	Float	จำนวนเงินทั้งหมดจากลูกค้า	1,264,105.16
AverageRevenuePerMonth	Float	จำนวนเงินเฉลี่ยที่ลูกค้าชำระต่อเดือน	20,388.79
PaymentMethod	Varchar	ประเภทการชำระเงิน	Bank transfer
PercentSatisfactionScore	Integer	ร้อยละคะแนนความพึงพอใจของลูกค้า	80

1.3.3 โมเดลที่ศึกษา

งานวิจัยจะพัฒนาและทดสอบเฉพาะ 7 โมเดล Machine Learning ได้แก่ 1. Decision Tree, 2. Random Forest, 3. Support Vector Machine, 4. Gradient Boosting Classifier, 5. Logistic Regression, 6. K-Nearest Neighbor, และ 7. AdaBoost Classifier

1.3.4 ตัวชี้วัดการประเมิน

งานวิจัยนี้ใช้ค่าตัวชี้วัด ได้แก่ Accuracy, Precision, Recall และ F1-score ในการประเมินประสิทธิภาพของแบบจำลองที่พัฒนา

1.3.5 ขอบเขตการนำไปใช้

ผลลัพธ์จากงานวิจัยนี้จะใช้เพื่อจำแนกการสูญเสียลูกค้าที่ยกเลิกการใช้ในธุรกิจการจัดการทรัพยากรบุคคล และไม่รวมถึงอุตสาหกรรมหรือประเภทบริการอื่น ๆ

1.4 สมมติฐานการวิจัย

สมมติฐานที่ 1: โมเดล Decision tree จะมีประสิทธิภาพสูงสุดในการทำนายจำแนกการสูญเสียลูกค้าจากการยกเลิกการใช้บริการ เมื่อเปรียบเทียบกับโมเดลอื่น ๆ เช่น Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbors, และ AdaBoost Classifier

สมมติฐานที่ 2: ลูกค้าที่มีรูปแบบการชำระเงินที่ไม่สม่ำเสมอจะมีแนวโน้มที่จะเลิกใช้บริการมากกว่าลูกค้าที่มีรูปแบบการชำระเงินที่สม่ำเสมอ

1.5 ประโยชน์ที่คาดว่าจะได้รับ

สามารถเลือกแบบจำลองการเรียนรู้ของเครื่องที่มีความเหมาะสมกับลักษณะข้อมูลของธุรกิจการจัดการทรัพยากรบุคคลได้อย่างมีประสิทธิภาพ รวมถึงสามารถดำเนินการสร้างแบบจำลองแบบรวม (Ensemble Learning) เพื่อเพิ่มประสิทธิผลในการจำแนกกลุ่มลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการ ทั้งนี้ผลลัพธ์จากการวิจัยจะสามารถนำไปประยุกต์ใช้เป็นเครื่องมือในการวิเคราะห์ความเสี่ยงของลูกค้าสำหรับภาคธุรกิจที่ให้บริการในลักษณะ B2B ได้อย่างแม่นยำยิ่งขึ้น และสนับสนุนการวางแผนกลยุทธ์เชิงรุกในการรักษาฐานลูกค้าในอนาคต

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

บทนี้นำเสนอการทบทวนวรรณกรรมที่เกี่ยวข้องกับหัวข้อสำคัญต่าง ๆ ที่สนับสนุนการดำเนินงานวิจัย ได้แก่ แนวคิดเกี่ยวกับพาณิชย์อิเล็กทรอนิกส์ (E-commerce) การยกเลิกการใช้บริการของลูกค้า (Customer Churn) และหลักการของการเรียนรู้ของเครื่อง (Machine Learning) รวมถึงอัลกอริทึมที่ใช้ในการจัดประเภท (Classification Algorithms) ปัญหาความไม่สมดุลของข้อมูล (Imbalance Data) และแนวทางในการจัดการ ตลอดจนวิธีการประเมินแบบจำลอง (Model Evaluation) เพื่อให้เข้าใจถึงความเหมาะสมและประสิทธิภาพของแบบจำลองที่พัฒนา

งานวิจัยนี้จะจำแนกการสูญเสียลูกค้าโดยอาศัยแบบจำลองการเรียนรู้ของเครื่องและวิธีการประเมินผล ซึ่งได้นำเสนอรายละเอียดของงานวิจัยดังกล่าวไว้ในตาราง 2 และ 3

ตาราง 2 คำอธิบายการเรียนรู้ของเครื่อง

ลำดับ	การเรียนรู้ของเครื่อง (ตัวย่อ)	คำอธิบาย
1	DT	Decision Tree
2	RF	Random Forest
3	SVM	Support Vector Machine
4	GB	Gradient Boosting Classifier
5	LR	Logistic Regression
6	K-NN	K-Nearest Neighbors
7	ABC	AdaBoost Classifier

ตาราง 3 คำอธิบายตัวชี้วัดประเมินประสิทธิภาพของโมเดลการเรียนรู้ของเครื่อง

ลำดับ	ตัวชี้วัด (ตัวย่อ)	คำอธิบาย
1	Ac	Accuracy Score
2	Re	Recall Score
3	Pc	Precision Score
4	F1	F1 Score
5	ROC	Receiver Operating Characteristic

2.1 **พาณิชย์อิเล็กทรอนิกส์** (E-commerce)

เพื่อสามารถเข้าใจข้อมูลที่นำมาใช้ได้อย่างถูกต้อง ผู้วิจัยจึงได้นำเสนอความรู้พื้นฐานเกี่ยวกับรูปแบบธุรกิจอีคอมเมิร์ซ (E-Commerce) ไว้ในบทนี้ ทั้งนี้เนื่องจากข้อมูลที่ใช้ในการศึกษาครั้งนี้เป็นข้อมูลจากบริษัทผู้ให้บริการแพลตฟอร์มการจัดการทรัพยากรบุคคล (HR Management Platform Provider) ซึ่งดำเนินธุรกิจในลักษณะระหว่างองค์กร (Business-to-Business: B2B) ดังนั้น การทำความเข้าใจลักษณะของธุรกิจที่เกี่ยวข้องจะช่วยให้สามารถตีความผลการศึกษาได้อย่างถูกต้องและครอบคลุมยิ่งขึ้น [1]

พาณิชย์อิเล็กทรอนิกส์ (E-Commerce) คือ กระบวนการการทำธุรกิจที่เกี่ยวข้องกับการซื้อขายสินค้าและบริการผ่านช่องทางดิจิทัล โดยเฉพาะการใช้อินเทอร์เน็ตเป็นเครื่องมือหลักในการดำเนินธุรกรรมทางการค้า ซึ่งรวมถึงการแลกเปลี่ยนข้อมูล การชำระเงิน การขนส่ง และการบริการหลังการขาย การพาณิชย์อิเล็กทรอนิกส์ทำให้ธุรกิจสามารถดำเนินการได้ในลักษณะออนไลน์ที่ครอบคลุมตลาดในระดับท้องถิ่นและระดับโลก โดยใช้เทคโนโลยีดิจิทัลในการสนับสนุนกระบวนการเหล่านี้เพื่อเพิ่มประสิทธิภาพ ลดต้นทุน และขยายขอบเขตทางการตลาด [2]

2.1.1 **ส่วนประกอบหลักของพาณิชย์อิเล็กทรอนิกส์**

1. การซื้อและการขายสินค้า/บริการ (Online Buying and Selling)

พาณิชย์อิเล็กทรอนิกส์ช่วยให้ผู้ซื้อและผู้ขายสามารถทำธุรกรรมโดยไม่ต้องพบกันตัวต่อตัว ผ่านช่องทางออนไลน์ เช่น เว็บไซต์ หรือแอปพลิเคชันบนมือถือ ซึ่งลูกค้าสามารถเลือกสินค้า สั่งซื้อ และชำระเงินได้

2. การชำระเงินออนไลน์ (Online Payment Systems)

ระบบชำระเงินออนไลน์เป็นส่วนสำคัญในการพาณิชย์อิเล็กทรอนิกส์ ซึ่งช่วยให้ผู้ซื้อสามารถชำระค่าสินค้าหรือบริการได้ง่ายและปลอดภัย โดยใช้เครื่องมือและระบบต่างๆ เช่น บัตรเครดิต/เดบิต, e-Wallet, การโอนเงินผ่านธนาคารออนไลน์, หรือการจ่ายเงินปลายทาง (COD - Cash on Delivery)

3. การขนส่งและการจัดส่งสินค้า (Logistics and Delivery)

พาณิชย์อิเล็กทรอนิกส์ต้องพึ่งพาระบบโลจิสติกส์ในการส่งสินค้าจากผู้ขายไปยังผู้ซื้อ โดยสามารถใช้บริษัทขนส่งหรือบริการจัดส่ง หรือบริการขนส่งของแพลตฟอร์มอีคอมเมิร์ซเองและมีการติดตามสถานะการจัดส่ง

4. การตลาดออนไลน์ (Online Marketing)

การตลาดออนไลน์เพื่อดึงดูดลูกค้าผ่านเครื่องมือดิจิทัล เช่น การโฆษณาผ่าน Google Ads, Facebook Ads, การทำ SEO (Search Engine Optimization) และการใช้ข้อมูลจากการวิเคราะห์พฤติกรรมผู้ใช้ในการปรับกลยุทธ์การตลาดให้มีประสิทธิภาพ

5. การบริการลูกค้าออนไลน์ (Online Customer Service)

การให้บริการลูกค้าออนไลน์ช่วยให้ธุรกิจสามารถตอบสนองความต้องการของลูกค้าได้ตลอดเวลา เช่น การใช้ระบบ Live Chat, การตอบคำถามผ่านโซเชียลมีเดีย, รวมถึงการใช้ระบบ AI หรือ Chatbots เพื่อให้บริการลูกค้าแบบอัตโนมัติ

2.1.2 รูปแบบของธุรกรรมของพาณิชย์อิเล็กทรอนิกส์

1. B2C (Business-to-Consumer)

เป็นรูปแบบที่ธุรกิจขายสินค้าและบริการให้กับลูกค้าทั่วไป โดยลูกค้าสามารถเข้าถึงและซื้อสินค้าผ่านเว็บไซต์หรือแอปพลิเคชัน ตัวอย่างเช่น Lazada, Shopee, Amazon, eBay

2. B2B (Business-to-Business)

รูปแบบธุรกิจที่มุ่งเน้นการซื้อขายสินค้าและบริการระหว่างธุรกิจด้วยกันเอง โดยไม่เกี่ยวข้องกับผู้บริโภคปลายทางโดยตรง ตัวอย่างของธุรกิจประเภทนี้ ได้แก่ การจัดหาวัตถุดิบ อุปกรณ์ หรือบริการที่จำเป็นสำหรับการดำเนินงานขององค์กร เช่น การให้บริการแพลตฟอร์มการจัดการทรัพยากรบุคคล (HR Management Platform) สำหรับองค์กรต่าง ๆ ซึ่งเป็นลักษณะเดียวกับบริษัทผู้ให้บริการข้อมูลในงานวิจัยฉบับนี้

3. C2C (Consumer-to-Consumer)

เป็นธุรกิจที่ผู้บริโภคขายสินค้าให้กับผู้บริโภคอื่นๆ โดยผ่านแพลตฟอร์มที่สนับสนุนการแลกเปลี่ยนสินค้า เช่น Facebook Marketplace, Carousell, Kaidee

4. C2B (Consumer-to-Business)

เป็นธุรกิจที่ผู้บริโภคเป็นผู้ที่เสนอสินค้า/บริการให้กับธุรกิจ เช่น ฟรีแลนซ์ที่ให้บริการด้านต่างๆ เช่น การออกแบบกราฟิกหรือการพัฒนาเว็บไซต์ในแพลตฟอร์ม Upwork หรือ Fiverr

5. B2G (Business-to-Government)

เป็นการค้าระหว่างธุรกิจกับหน่วยงานภาครัฐ ซึ่งอาจรวมถึงการจัดหาอุปกรณ์ซอฟต์แวร์ หรือบริการให้กับภาครัฐ

6. G2C (Government-to-Consumer)

เป็นการให้บริการหรือขายสินค้าโดยภาครัฐให้แก่ประชาชน เช่น การชำระภาษีออนไลน์ การจองคิวทำบัตรประชาชน หรือการสมัครรับบริการต่างๆ ที่ภาครัฐจัดให้

2.2 การยกเลิกการใช้บริการของลูกค้า (Customer Churn)

การยกเลิกการใช้บริการของลูกค้า (Customer Churn) หมายถึงการยุติการสมัครสมาชิก การยกเลิกการใช้งานผลิตภัณฑ์ หรือการไม่ทำธุรกรรมอีกต่อไป [3] การศึกษาเรื่องนี้มีความสำคัญอย่างยิ่งสำหรับธุรกิจ เนื่องจากการสูญเสียลูกค้าอาจมีผลกระทบโดยตรงต่อรายได้และการเติบโตของธุรกิจ

2.2.1 ประเภทของการยกเลิกการใช้บริการของลูกค้า

1. การยกเลิกการใช้บริการของลูกค้าแบบธรรมชาติ (Voluntary Churn)

เกิดจากการที่ลูกค้าตัดสินใจยกเลิกบริการหรือผลิตภัณฑ์ด้วยเหตุผลส่วนตัว เช่น ความไม่พึงพอใจต่อบริการ หรือการเลือกใช้บริการของคู่แข่ง

2. การยกเลิกการใช้บริการของลูกค้าแบบไม่ตั้งใจ (Involuntary Churn)

เกิดจากเหตุผลที่อยู่นอกเหนือจากการตัดสินใจของลูกค้า เช่น การชำระเงินล่าช้า หรือปัญหาทางเทคนิคที่ทำให้ลูกค้าไม่สามารถใช้บริการได้

2.2.2 สาเหตุที่อาจทำให้เกิดการยกเลิกการใช้บริการของลูกค้า

1. ราคาสูงหรือไม่มีความคุ้มค่า

หากราคาของบริการหรือผลิตภัณฑ์สูงเกินไป หรือไม่สามารถเทียบกับคุณค่าที่ได้รับ ลูกค้าอาจตัดสินใจยกเลิกบริการเพื่อหาทางเลือกที่ดีกว่า

2. การบริการลูกค้าที่ไม่ดี

ลูกค้าจะรู้สึกไม่พอใจหากได้รับบริการที่ไม่ดีหรือไม่ตรงกับที่คาดหวัง เช่น การตอบสนองที่ช้า หรือไม่มีการสนับสนุนที่มีประสิทธิภาพ

3. การแข่งขันจากคู่แข่ง

การที่มีคู่แข่งที่ให้บริการที่ดีกว่า หรือเสนอโปรโมชั่นที่น่าสนใจมากกว่า อาจดึงดูดลูกค้าไปยังคู่แข่ง

4. ประสบการณ์การใช้งานไม่ดี

หากลูกค้าไม่พอใจกับประสบการณ์การใช้งานผลิตภัณฑ์ เช่น ระบบซับซ้อนเกินไปหรือไม่ใช้งานง่าย ก็อาจทำให้ลูกค้าตัดสินใจหยุดใช้บริการ

5. ความไม่พึงพอใจจากบริการหรือผลิตภัณฑ์

ลูกค้าอาจรู้สึกว่าผลิตภัณฑ์หรือบริการที่ได้รับไม่ตรงตามความคาดหวัง หรือไม่สามารถตอบโจทย์ความต้องการ

2.2.3 การคำนวณอัตราการยกเลิกการใช้บริการของลูกค้า

Churn Rate หรืออัตราการยกเลิกการใช้บริการ คือ การวัดระดับการสูญเสียลูกค้าของธุรกิจ ซึ่งคำนวณได้จากการนำจำนวนลูกค้าที่ยกเลิกบริการในช่วงเวลาหนึ่งมาหารด้วยจำนวนลูกค้าทั้งหมดในช่วงเวลาเดียวกันดังสมการ (1)

$$churn\ rate = \frac{\text{จำนวนลูกค้าที่ยกเลิกใช้บริการ}}{\text{จำนวนลูกค้าทั้งหมดในช่วงเวลาเดียวกัน}} \times 100 \quad (1)$$

2.2.4 ผลกระทบจากการยกเลิกการใช้บริการของลูกค้า

1. รายได้ลดลง

การสูญเสียลูกค้าจะทำให้รายได้ของธุรกิจลดลง เนื่องจากการเสียฐานลูกค้าเดิม

2. เพิ่มค่าใช้จ่ายในการหาลูกค้าใหม่

การรักษาลูกค้าเก่ามักจะมีต้นทุนต่ำกว่าการหาลูกค้าใหม่ การสูญเสียลูกค้าหมายถึงการเพิ่มค่าใช้จ่ายในการหาลูกค้าใหม่เพื่อทดแทน

3. ผลกระทบต่อแบรนด์และชื่อเสียง

หากลูกค้าหยุดใช้บริการจำนวนมากอาจส่งผลกระทบต่อภาพลักษณ์และความน่าเชื่อถือของแบรนด์

4. การสูญเสียข้อมูลเชิงลึก

นอกการการเสียรายได้ที่ได้จากการใช้บริการของลูกค้าแต่ละรายแล้วยังทำให้สูญเสียข้อมูลและความคิดเห็นที่สามารถนำไปใช้ปรับปรุงผลิตภัณฑ์หรือบริการ

การยกเลิกการใช้บริการของลูกค้าจึงเป็นปัญหาสำคัญที่ธุรกิจต้องรับมือและหาทางป้องกัน เนื่องจากการสูญเสียลูกค้าอาจส่งผลกระทบต่อการเติบโตและความยั่งยืนของธุรกิจ การเข้าใจสาเหตุของการยกเลิกบริการและการนำกลยุทธ์ต่างๆ มาปรับใช้ในการรักษาลูกค้าจะช่วยให้ธุรกิจลดอัตราการยกเลิกการใช้บริการของลูกค้าและสร้างความภักดีจากลูกค้าในระยะยาว

2.3 การเรียนรู้ของเครื่อง (Machine Learning)

การเรียนรู้ของเครื่อง (Machine Learning) เป็นส่วนสำคัญในสาขาวิทยาการข้อมูลที่กำลังเติบโตอย่างรวดเร็ว แบบจำลองจะถูกฝึกเพื่อทำการจำแนกประเภทหรือทำนายผลและเผย

ข้อมูลที่มีคุณค่าในการขุดข้อมูล (Data Mining) [4] โดยใช้ระเบียบวิธีทางสถิติ กลไกการเรียนรู้ของแบบจำลอง ในการเรียนรู้ของเครื่องสามารถแบ่งออกเป็น 3 ส่วนหลักดังนี้

1. กระบวนการตัดสินใจ (Decision process) เป็นกระบวนการที่ทำการทำนายหรือการจำแนกประเภทโดยการเรียนรู้ของเครื่องจะทำการประเมินรูปแบบในข้อมูลนำเข้า ซึ่งอาจจะเป็นข้อมูลที่มีการติดป้ายกำกับ (labeled) หรือไม่ติดป้ายกำกับ (unlabeled)

2. ฟังก์ชันข้อผิดพลาด (Error function) จะทำการประเมินการทำนายของโมเดล หากมีกรณีที่ทราบล่วงหน้า ฟังก์ชันข้อผิดพลาดจะทำการเปรียบเทียบเพื่อตรวจสอบความถูกต้องของโมเดล

3. กระบวนการปรับแต่งโมเดล (Model optimization process) น้ำหนักของโมเดลจะถูกปรับเปลี่ยนเพื่อลดความแตกต่างระหว่างตัวอย่างที่ทราบและการทำนายของโมเดล หากโมเดลสามารถพอดีกับจุดข้อมูลในชุดข้อมูลฝึกฝนได้ดีขึ้น อัลกอริธึมจะทำการดำเนินกระบวนการประเมินและปรับแต่งซ้ำไปเรื่อยๆ โดยการอัปเดตน้ำหนักจนกว่าจะถึงเกณฑ์ความแม่นยำที่กำหนด

2.3.1 โมเดลการเรียนรู้ของเครื่อง 4 ประเภทหลัก

1. Supervised Machine Learning

การใช้ชุดข้อมูลที่มีการติดป้ายกำกับ (Labeled Dataset) เพื่อฝึกอัลกอริธึมในการจำแนกประเภทข้อมูลหรือทำนายผลลัพธ์อย่างแม่นยำ เมื่อข้อมูลถูกนำเข้าสู่โมเดล โมเดลจะทำการปรับน้ำหนักจนกระทั่งเหมาะสม จากนั้นในกระบวนการข้ามการตรวจสอบ (Cross-Validation) จะถูกใช้เพื่อป้องกัน overfitted หรือ underfit เทคนิคที่ใช้ในกระบวนการเรียนรู้แบบมีผู้ควบคุม ได้แก่ Neural Networks, Naïve Bayes, Linear Regression, Logistic Regression , Random Forest, และ Support Vector Machines [5]

2. Unsupervised Machine Learning

การเรียนรู้แบบไม่มีการติดป้ายกำกับ (Unlabeled Dataset) อัลกอริธึมเหล่านี้สามารถระบุรูปแบบที่ซ่อนอยู่หรือกลุ่มของข้อมูล [6] วิธีนี้เหมาะสำหรับการวิเคราะห์ข้อมูลเชิงสำรวจ (Exploratory Data Analysis), การแบ่งกลุ่มลูกค้า (Consumer Segmentation), และการระบุรูปภาพและลวดลาย เนื่องจากสามารถค้นหาความคล้ายคลึงและความแตกต่างในข้อมูลได้ นอกจากนี้ การลดมิติ (Dimensionality Reduction) ช่วยลดจำนวนคุณสมบัติในโมเดล เทคนิคมาตรฐานที่ใช้ในการลดมิติได้แก่ การแยกค่าคุณสมบัติที่สำคัญ (Singular Value Decomposition, SVD) และการวิเคราะห์องค์ประกอบหลัก (Principal Component Analysis, PCA) เทคนิคที่ใช้ในการเรียนรู้แบบไม่มีผู้ควบคุม ได้แก่ Neural Networks, KMeans Clustering

3. Semi-Supervised Machine Learning

วิธีการเรียนรู้นี้ใช้ในการฝึกฝนชุดข้อมูลที่มีทั้งข้อมูลที่ติดป้ายกำกับและข้อมูลที่ไม่มีการติดป้ายกำกับ เหมาะกับการสกัดคุณสมบัติที่สำคัญจากข้อมูล และเมื่อปริมาณข้อมูลมีมาก ซึ่งจะช่วยในการจัดกลุ่มและการสกัดคุณสมบัติในระหว่างการฝึกฝนจากชุดข้อมูลที่มีขนาดใหญ่และไม่มีการติดป้ายกำกับ โดยใช้ชุดข้อมูลที่มีการติดป้ายกำกับในขนาดที่เล็กลง นอกจากนี้ยังสามารถแก้ปัญหาขาดแคลนข้อมูลที่มีการติดป้ายกำกับสำหรับอัลกอริธึมการเรียนรู้แบบมีผู้ควบคุม เป็นวิธีที่เหมาะสมในการวิเคราะห์ภาพทางการแพทย์ เนื่องจากการมีข้อมูลการฝึกฝนในปริมาณน้อยสามารถเพิ่มความแม่นยำได้อย่างมีนัยสำคัญ

4. การเรียนรู้ของเครื่องแบบเสริมแรง (Reinforcement Machine Learning)

วิธีการเรียนรู้นี้จะไม่ได้รับการฝึกฝนจากข้อมูลตัวอย่าง แต่โมเดลนี้เรียนรู้จากกระบวนการลองผิดลองถูก (trial and error) ผลลัพธ์ที่สำเร็จจะช่วยเสริมการสร้างคำแนะนำหรือกลยุทธ์ที่เหมาะสมที่สุดสำหรับสถานการณ์ที่เฉพาะเจาะจง

2.4 **อัลกอริทึมการจำแนกประเภท** (Classification Algorithms)

การจำแนกประเภท (Classification) เป็นเทคนิคการเรียนรู้แบบ supervised learning ใช้กับข้อมูลที่มีโครงสร้างและข้อมูลที่ไม่มีโครงสร้าง [7] เพื่อระบุประเภทของข้อมูลใหม่ที่ได้รับโดยอ้างอิงจากข้อมูลที่ได้รับการฝึกฝนไว้ และคำนวณความเป็นไปได้ที่ข้อมูลใหม่จะถูกจัดให้อยู่ในประเภทที่ได้มีการจัดหมวดหมู่ไว้ก่อนหน้านี้ ตัวจำแนกประเภท (Classifier) คือ อัลกอริธึมที่ทำหน้าที่ในการจำแนกข้อมูลในชุดข้อมูลหนึ่งๆ การจำแนกประเภทสามารถแบ่งออกได้เป็นสองประเภทหลักดังนี้

1. ตัวจำแนกประเภทแบบไบนารี (Binary Classifier) ใช้ในการจำแนกประเภทมีผลลัพธ์เพียงสองตัวเลือก เช่น ใช่หรือไม่ ซึ่งในการศึกษานี้จะมีการแบ่งประเภทเป็น ลูกค้าที่ยกเลิกใช้บริการ และลูกค้าที่ยังคงใช้บริการอยู่

2. ตัวจำแนกประเภทหลายคลาส (Multi-class Classifier) ใช้ในการจำแนกประเภทมีผลลัพธ์มากกว่าสองตัวเลือก เช่น การจำแนกประเภทของพืชชนิดต่างๆ หรือประเภทของดนตรี

2.4.1 Decision Tree

เป็นอัลกอริธึมที่ใช้โครงสร้างรูปต้นไม้ในการทำการจำแนกประเภท (Classification) หรือการทำนายผล โดยการแยกข้อมูลตามคุณสมบัติต่างๆดังภาพประกอบ 1 ซึ่งทำให้สามารถสร้างแบบจำลองที่มีความเข้าใจง่าย โมเดลนี้เริ่มจากการเลือกคุณสมบัติที่ดีที่สุดในการแบ่งข้อมูล

ในแต่ละขั้นตอนและดำเนินการแยกข้อมูลจนถึงจุดสิ้นสุด (leaf node) ที่เป็นผลลัพธ์การทำนาย ผลลัพธ์ของการทำนายจะอยู่ในรูปของคำตอบที่แยกแยะตามประเภทของข้อมูลที่มีการฝึกฝน [8]

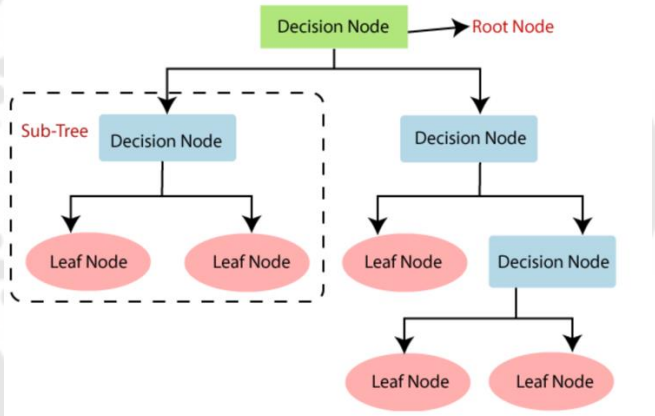
2.4.1.1 โครงสร้างของ Decision tree

1. Root Node หรือโหนดราก เป็นจุดเริ่มต้นของการแบ่งข้อมูล โดยใช้เงื่อนไขหรือเกณฑ์ในการแยกข้อมูลออกเป็นกลุ่มย่อย

2. Internal Nodes หรือโหนดภายใน เกิดจากการแบ่งข้อมูลตามเงื่อนไขที่กำหนด เพื่อจัดกลุ่มข้อมูลที่มีลักษณะคล้ายกัน

3. Leaf Nodes หรือโหนดใบ เป็นจุดสิ้นสุดของกระบวนการจำแนก โดยจะแสดงผลลัพธ์ของการทำนาย และไม่สามารถแยกย่อยเพิ่มเติมได้อีก

4. Branches หรือกิ่งก้าน คือเส้นเชื่อมระหว่างโหนดต่าง ๆ ซึ่งแสดงถึงทิศทางของการแบ่งตามเงื่อนไขที่กำหนดในแต่ละขั้นของต้นไม้การตัดสินใจ



ภาพประกอบ 1 องค์ประกอบของ Decision tree

ที่มาของภาพ: https://www.mdpi.com/2076-3417/11/15/6728

2.4.1.2 หลักการทำงาน (Algorithm) ของ Decision Tree

ขั้นตอนที่ 1: เลือก Feature ในการแยกข้อมูล (Feature Selection)

อัลกอริทึมจะเลือกคุณลักษณะ (Feature) ที่เหมาะสมที่สุดในแต่ละขั้นตอนในการแบ่งข้อมูล โดยการเลือก Feature อาศัยเกณฑ์ที่ช่วยลดความไม่แน่นอนในการแบ่งข้อมูล ซึ่งมีวิธีที่นิยมใช้ ได้แก่:

- Information Gain (Entropy)

ใช้ค่าความไม่แน่นอน (Entropy) ของข้อมูลมาช่วยวัดว่าการเลือกแยก Feature ไหนจะช่วยลดความไม่แน่นอนได้มากที่สุด Feature ที่ลด Entropy ได้มากที่สุดจะถูกเลือก Entropy คำนวณจากสูตรดังสมการ (2)

$$Entropy(S) = -\sum_{i=1}^{c} p_i log_2(p_i) \quad (2)$$

โดยที่ S คือ ชุดข้อมูลที่ใช้พิจารณา

C คือ จำนวนคลาสทั้งหมดในชุดข้อมูล

p_i คือสัดส่วนของข้อมูลที่อยู่ในคลาส i

- Gini Impurity

ค่านี้ใช้วัดความบริสุทธิ์ของกลุ่มข้อมูลหลังแบ่งกลุ่ม โดยการเลือก Feature ที่ทำให้เกิดความบริสุทธิ์สูงที่สุด (Gini Impurity ต่ำที่สุด) จะถูกเลือกในการแบ่งข้อมูล Gini Impurity คำนวณจากสูตรดังสมการ (3)

$$Gini(S) = 1 - \sum_{i=1}^{c} p_i^2) \quad (3)$$

โดยที่ S คือ ชุดข้อมูลที่ใช้พิจารณา

C คือ จำนวนคลาสทั้งหมดในชุดข้อมูล

p_i คือสัดส่วนของข้อมูลที่อยู่ในคลาส i

- Variance Reduction (สำหรับ Regression Trees)

ใช้วัดความแตกต่างของข้อมูลต่อเนื่อง โดยเลือก Feature ที่ลดความแปรปรวนของข้อมูลให้มากที่สุด สำหรับงาน Regression ซึ่งข้อมูลเป็นค่าต่อเนื่อง สามารถคำนวณ Variance Reduction จากสูตรดังสมการ (4)

$$Variance\ Reduction(S, A) = Var(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Var(S_v) \quad (4)$$

โดยที่ Var(S) คือ ค่าความแปรปรวนของข้อมูลชุด

$Var(S_v)$ คือ ค่าความแปรปรวนของชุดข้อมูลย่อยหลังแบ่งตามค่า v ของ Feature A

S_v คือ จำนวนข้อมูลของกลุ่มย่อยหลังแบ่ง

|S| คือจำนวนข้อมูลทั้งหมดก่อนแบ่ง

$|S_v|$ คือจำนวนข้อมูลในแต่ละกลุ่มหลังแบ่ง

ขั้นตอนที่ 2: แบ่งข้อมูลตามเงื่อนไขของ Feature ที่ถูกเลือก (Data Splitting)

เมื่อเลือก Feature แล้ว ข้อมูลจะถูกแบ่งออกเป็นกลุ่มย่อยตามเงื่อนไขที่เหมาะสม เช่น ถ้าเป็นลูกค้าที่ยังใช้บริการอยู่ ให้ไปกลุ่มที่ 1 หรือถ้าเป็นลูกค้าที่ยกเลิกการใช้บริการให้ไปกลุ่มที่ 2 จากนั้นทำซ้ำขั้นตอนนี้ต่อไปเรื่อยๆ กับกลุ่มข้อมูลใหม่ที่ได้มา จนกว่าจะถึงเกณฑ์ที่หยุดการแบ่งข้อมูล (Stopping Criteria)

ขั้นตอนที่ 3: หยุดการแบ่งข้อมูล (Stopping Criteria)

การหยุดแบ่งข้อมูลจะเกิดขึ้นเมื่อมีเงื่อนไขได้แก่ กลุ่มข้อมูลมีความบริสุทธิ์ (ข้อมูลทั้งหมดเป็นกลุ่มเดียวกันแล้ว) หรือถึงระดับความลึกสูงสุด (Maximum Depth) หรือจำนวนข้อมูลในแต่ละโหนดต่ำกว่าที่กำหนด (Minimum samples per leaf)

2.4.2 Random Forest

เป็นอัลกอริธึมที่ใช้การรวมหลายๆ ต้นไม้การตัดสินใจเข้าด้วยกัน โดยการสร้างต้นไม้หลายๆ ต้นจากข้อมูลที่ถูกสุ่มเลือก (bagging) และใช้การโหวตจากต้นไม้ทั้งหมดเพื่อให้ผลลัพธ์ที่แม่นยำขึ้น วิธีนี้ช่วยลดปัญหาการ overfitting ซึ่งเป็นปัญหาที่อาจเกิดขึ้นจากการใช้ต้นไม้การตัดสินใจเพียงต้นเดียว โดย Random Forest สามารถใช้ได้ทั้งงาน Classification (การจำแนกข้อมูล) และ Regression (การพยากรณ์ค่าต่อเนื่อง) [9]

2.4.2.1 หลักการทำงาน (Algorithm) ของ Random Forest

ขั้นตอนที่ 1: Bootstrap Sampling (สุ่มข้อมูลด้วยการแทนที่)

สุ่มข้อมูลจาก Training Set เพื่อสร้าง Subset สำหรับการฝึกแต่ละต้นไม้ โดยจำนวนข้อมูลที่สุ่มมา = ขนาดของ Training Set และข้อมูลสามารถซ้ำกันได้ (sampling with replacement)

ขั้นตอนที่ 2: Random Feature Selection (สุ่ม Feature)

ในการแยกแต่ละโหนดของต้นไม้ จะสุ่มเลือกเพียงบางส่วนของ Features แทนการใช้ทั้งหมด ถ้ามี m Features ให้สุ่มเท่ากับ $\sqrt{\mathrm{m}}$ สำหรับ Classification และ $\frac{m}{3}$ สำหรับ Regression

ขั้นตอนที่ 3: การแยกโหนด (Splitting Nodes)

ใช้เกณฑ์เดียวกับ Decision Tree ได้แก่ Gini Impurity, Information Gain (Entropy), Variance Reduction (สำหรับ Regression)

2.4.2.1 ตัวแปรที่สำคัญต่อการทำงานของ Random Forest

ตัวแปรที่สำคัญต่อการทำงานของ Random Forest ได้แก่ n_estimators, max_features, max_depth, min_samples_split, min_samples_split, criterion ซึ่งความหมายของแต่ละตัวแปรจะถูกอธิบายในตาราง 4

ตาราง 4 ตัวแปรที่สำคัญต่อการทำงานของ Random Forest

ตัวแปร	ความหมาย
n_estimators	จำนวนต้นไม้ (Trees)
max_features	จำนวน Feature ที่ใช้สุ่มในแต่ละโหนด
max_depth	ความลึกสูงสุดของต้นไม้แต่ละต้น
min_samples_split	จำนวนข้อมูลขั้นต่ำที่ต้องมีเพื่อแยกโหนด
criterion	เกณฑ์ที่ใช้แยกโหนด

2.4.3 Support Vector Machine

เป็นอัลกอริธึมที่ใช้สำหรับการจำแนกประเภทข้อมูลโดยการหาขอบเขต (hyperplane) ที่สามารถแยกข้อมูลได้อย่างดีที่สุดระหว่างสองคลาส ขอบเขตนี้จะต้องมีระยะห่างมากที่สุดจากข้อมูลทั้งสองคลาส เทคนิคนี้มีประสิทธิภาพสูงในการจำแนกข้อมูลที่ไม่เป็นเชิงเส้นและสามารถใช้ได้ดีในพื้นที่มิติสูง [10]

2.4.3.1 หลักการทำงาน (Algorithm) ของ Support Vector Machine (SVM)

ขั้นตอนที่ 1 การหาเส้น Hyperplane ที่ดีที่สุด

หาเส้น Hyperplane ที่ดีที่สุด โดย Hyperplane คือเส้นตรง (2D), ระนาบ (3D), หรือระนาบความมิติสูง (High-dimensional Plane) ที่ใช้ในการแบ่งข้อมูลระหว่างคลาส การหาเส้น Hyperplane ดังสมการ (5)

$$w \cdot x + b = 0 \tag{5}$$

โดยที่ w คือ เวกเตอร์ของน้ำหนัก (weight vector)

x คือ เวกเตอร์ของข้อมูล (input vector)

b คือ ค่า bias หรือ offset

ขั้นตอนที่ 2 การหาค่า Margin

Margin คือระยะห่างระหว่างเส้น Hyperplane กับจุดข้อมูลที่ใกล้ที่สุดของแต่ละคลาส ซึ่งจุดเหล่านี้เรียกว่า Support Vectors จุดประสงค์ของ SVM คือการหาค่า w และ b ที่ทำให้ Margin มากที่สุด และสามารถหาค่า Margin ได้จากสูตรดังสมการ (6)

$$Margin = \frac{2}{||w||} \tag{6}$$

โดย w คือ เวกเตอร์ของน้ำหนัก (weight vector)

2.4.3.2 กรณีที่ข้อมูลไม่สามารถแบ่งได้ด้วยเส้นตรง

ใช้เทคนิค Kernel Trick เพื่อเปลี่ยนข้อมูลจากมิติเดิมไปยังมิติที่สูงขึ้น เพื่อให้สามารถแบ่งด้วยเส้นตรงได้ โดยฟังก์ชัน Kernel ที่นิยมใช้ได้แก่

1. Linear Kernel ดังสมการ (7)

$$K\left(x_i, x_j\right) = x_i \cdot x_j \tag{7}$$

2. Polynomial Kernel ดังสมการ (8)

$$K\left(x_i, x_j\right) = (x_i \cdot x_j + r)^d \tag{8}$$

3. Radial Basis Function (RBF) / Gaussian Kernel ดังสมการ (9)

$$K\left(x_i, x_j\right) = exp(-\Upsilon||x_i \cdot x_j||^d) \tag{9}$$

โดยที่ γ คือ ตัวควบคุมความโค้งของเส้นแบ่ง (ค่าน้อย = โค้งกว้าง, ค่ามาก = โค้งแคบ)

2.4.3.3 ตัวแปรที่สำคัญต่อการทำงานของ Support Vector Machine (SVM)

ตัวแปรที่สำคัญต่อการทำงานของ Support Vector Machine ประกอบด้วย C, Kernel, Gamma, และ Degree จะกล่าวถึงความหมายและคำอธิบายในตาราง 5

ตาราง 5 ตัวแปรที่สำคัญต่อการทำงานของ Support Vector Machine (SVM)

ตัวแปร	ความหมาย	คำอธิบาย
C	Regularization parameter	ควบคุมการอนุญาตให้มีการผิดพลาดในการจำแนก
Kernel	Kernel function	เลือกรูปร่างของเส้นแบ่ง เช่น "linear", "rbf", "poly"
Gamma	Parameter สำหรับ RBF Kernel	กำหนดความซับซ้อนของเส้นแบ่ง
Degree	ใช้เมื่อเลือก Polynomial Kernel	กำหนดจำนวนพจน์ของ Polynomial

2.4.4 Gradient Boosting Classifier

เป็นเทคนิคการเรียนรู้ที่ใช้วิธีการเสริมกำลัง (boosting) ซึ่งสร้างโมเดลหลายๆ ตัวโดยการเรียนรู้จากข้อผิดพลาดของโมเดลก่อนหน้า อัลกอริธึมนี้จะปรับน้ำหนักของตัวอย่างที่ถูกจำแนกผิดในแต่ละรอบเพื่อให้โมเดลมีความแม่นยำมากขึ้น Gradient Boosting Classifier เป็นเวอร์ชันที่มีการปรับปรุงจากการเสริมกำลังแบบธรรมดาและมีความเร็วในการคำนวณสูง [11]

2.4.4.1 หลักการทำงาน (Algorithm) ของ Gradient Boosting Classifier

ทำนายค่า $\widehat{y_i}$ จากข้อมูล x_i โดยใช้ Tree Ensemble ซึ่งจะเป็นผลรวมของต้นไม้ K ต้น สามารถคำนวณได้จากสูตรดังสมการ (10)

$$\widehat{y_i} = \sum_{k=1}^{K} f_k(x_i),\ f_k \epsilon\, \mathcal{F} \tag{10}$$

โดยที่ F คือ เซตของฟังก์ชันตัดสินใจ

$F_k(x_i)$ คือ โมเดลต้นไม้ต้นที่ k

2.4.4.2 ตัวแปรที่สำคัญต่อการทำงานของ Gradient Boosting Classifier

ตัวแปรที่สำคัญต่อการทำงานของ Gradient Boosting Classifier ประกอบด้วย Eta, max_depth, subsample, colsample_bytree, lambda, alpha, gamma, n_estimators จะกล่าวถึงความหมายและคำอธิบายในตาราง 6

ตาราง 6 ตัวแปรที่สำคัญต่อการทำงานของ Gradient Boosting Classifier

ตัวแปร	ความหมาย	คำอธิบาย
Eta	อัตราการเรียนรู้	ควบคุมการอัปเดตในแต่ละรอบ
max_depth	ความลึกสูงสุดของต้นไม้	ช่วยควบคุมความซับซ้อน
subsample	สัดส่วนข้อมูลที่ใช้ในแต่ละรอบ	ป้องกัน overfitting
colsample_bytree	สัดส่วนของ Features ที่ใช้สร้างแต่ละต้น	เพิ่มความหลากหลาย
lambda	L2 regularization	ป้องกัน overfitting
alpha	L1 regularization	ช่วยทำให้ model บางลง
gamma	ค่าปรับโทษในการแยกโหนด	เพิ่มความรัดกุมในการแยก
n_estimators	จำนวนรอบของ boosting	จำนวนต้นไม้ทั้งหมด

2.4.5 Logistic Regression

เป็นเทคนิคการทำนายผลลัพธ์ที่เป็นแบบสองตัวเลือก (binary outcome) โดยการคำนวณความน่าจะเป็นของผลลัพธ์ผ่านฟังก์ชันโลจิสติกส์ ซึ่งทำให้ผลลัพธ์อยู่ในช่วงระหว่าง 0 และ 1 เทคนิคนี้สามารถใช้ในการจำแนกประเภท เช่น การทำนายว่าจะซื้อหรือไม่ซื้อสินค้าของลูกค้า [12]

2.4.5.1 หลักการทำงาน (Algorithm) ของ Logistic Regression

ขั้นตอนที่ 1: Linear Combination of Inputs

คำนวณค่าเชิงเส้นของข้อมูลจากสูตรดังสมการ (11)

$$z = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n \quad (11)$$

โดยที่ x คือ เวกเตอร์ของ Feature input

w คือ เวกเตอร์ของน้ำหนัก (Weights)

z คือ ค่าผลรวมเชิงเส้น

ขั้นตอนที่ 2: Sigmoid Function (หรือ Logistic Function)

แทนที่การใช้ z โดยตรง ด้วย Sigmoid function เพื่อให้ค่าที่ได้อยู่ในช่วง [0, 1] สามารถตีความเป็นความน่าจะเป็นได้ ซึ่งค่าที่ได้คือความน่าจะเป็นที่ตัวอย่างจะอยู่ในคลาส 1 จากสูตรดังสมการ (12)

$$\sigma(z) = \frac{1}{1+ e^{-z}} \tag{12}$$

ขั้นตอนที่ 3: การตัดสินใจ (Prediction)

เมื่อได้ค่าจาก Sigmoid Function โมเดลจะตัดสินใจว่าอยู่ในคลาสใด โดยปกติใช้เกณฑ์ หาก $\boldsymbol{\sigma}$(z) มากกว่าหรือเท่ากับ 0.5 แล้วผลลัพธ์จะเท่ากับ 1 แต่หาก $\boldsymbol{\sigma}$(z) น้อยกว่า 0.5 แล้วผลลัพธ์จะเท่ากับ 0

2.4.6 K-Nearest Neighbors

เป็นเทคนิคการจำแนกประเภทที่ใช้ข้อมูลที่ใกล้เคียงที่สุด (nearest neighbors) ในการทำนายผลลัพธ์ โดยจะคำนวณจาก K ตัวอย่างที่ใกล้เคียงที่สุดในชุดข้อมูลที่ฝึกฝน จากนั้นผลลัพธ์จะถูกกำหนดจากการโหวตของ K ตัวอย่างที่ใกล้ที่สุด เทคนิคนี้ไม่ต้องการการฝึกฝนโมเดลล่วงหน้าและง่ายต่อการเข้าใจ [13]

2.4.6.1 หลักการทำงาน (Algorithm) ของ K-Nearest Neighbors

ขั้นตอนที่ 1: เลือกค่าของ K

K คือ Hyperparameter ซึ่งเป็นจำนวนเพื่อนบ้านใกล้สุดที่ใช้พิจารณา

ขั้นตอนที่ 2: วัดระยะห่าง (Distance Measurement)

มี 3วิธีหลักในการวัดระยะห่างได้แก่

1. Euclidean distance คำนวณได้จากสูตรดังสมการ (13)

$$d(x, x_i) = \sqrt{\sum_{j=1}^{n}(x_j - x_{ij})^2} \tag{13}$$

2. Manhattan distance คำนวณได้จากสูตรดังสมการ (14)

$$d = \sum |x_j - x_{ij}| \tag{14}$$

3. Minkowski distance (generalized distance) คำนวณได้จากสูตรดังสมการ (15)

$$d(x, x_i) = (\sum_{j=1}^{n} |x_j - x_{ij}|^p)^{1/p} \tag{15}$$

โดยที่ x คือ จุดข้อมูลใหม่

x_i คือ จุดข้อมูลใน Training set

x_j และ x_{ij} คือ ค่าของ feature ตัวที่ j

2.4.7 AdaBoost Classifier

เป็นอัลกอริธึมที่ใช้วิธีการเสริมกำลังแบบปรับตัว (adaptive) เพื่อเพิ่มประสิทธิภาพของโมเดลโดยให้ความสำคัญกับตัวอย่างข้อมูลที่ถูกจำแนกผิด ในแต่ละรอบของการฝึกฝน โมเดลจะปรับน้ำหนักให้กับตัวอย่างที่ผิดพลาดเพื่อให้โมเดลมีความสามารถในการจำแนกข้อมูลได้ดีขึ้น เทคนิคนี้มักจะใช้ร่วมกับอัลกอริธึมที่เป็นตัวแยกประเภทที่ไม่ซับซ้อน เช่น Decision tree [14]

2.4.7.1 หลักการทำงาน (Algorithm) ของ AdaBoost Classifier

ขั้นตอนที่ 1 การตั้งน้ำหนักข้อมูล

ตั้งน้ำหนักให้ข้อมูลแต่ละตัวอย่างเท่ากัน จากสูตรดังสมการ (16) (สำหรับ i = 1,2, … , n)

$$w_i^{(1)} = \frac{1}{n} \tag{16}$$

ขั้นตอนที่ 2 การฝึก Weak Leaner

ฝึก Weak Learner ด้วยน้ำหนัก w_i และคำนวณ Error rate ของ weak classifier ตัวที่ t คำนวณจากสูตรดังสมการ (17)

$$\varepsilon_t = \sum_{i=1}^{n} w_i^{(t)} \cdot I(h_t(x_i) \neq y_i) \tag{17}$$

โดยที่ $h_t(x_i)$ คือ ผลการทำนายของ weak classifier

y_i คือ ค่าจริง

I คือ ฟังก์ชันบ่งชี้ (ให้ค่า 1 เมื่อทำนายผิด, 0 เมื่อถูก)

ขั้นตอนที่ 3 การคำนวณน้ำหนัก Weak Learner

คำนวณ น้ำหนักของ weak learner จากสูตรดังสมการ (18)

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{18}$$

จากสูตรข้างต้น ยิ่ง weak classifier แม่นยำมาก ε_t น้อย จะส่งผลให้ α_t ยิ่งสูง (ให้ความสำคัญมาก)

ขั้นตอนที่ 4 การอัปเดตน้ำหนักของข้อมูล

อัปเดตน้ำหนักของข้อมูลดังสมการ (19)

$$w_i^{(t+1)} = w_i^t \cdot e^{-\alpha_t y_i h_t(x_i)} \tag{19}$$

จากนั้น normalize ให้รวมกันได้ 1 ดังสมการ (20)

$$w_i^{(t+1)} = \frac{w_i^{(t+1)}}{\sum_{j=1}^{n} w_i^{(t+1)}} \tag{20}$$

โดยที่ w_i^t คือ น้ำหนักของตัวอย่างที่ i ในรอบที่ t

α_t คือ น้ำหนักของ weak learner ที่ t

h_t คือ weak classifier ในรอบที่ t

จากสูตรข้างต้น หากทำนายถูก $y_i = h_t(x_i)$ จะทำให้ exponent เป็นลบ และน้ำหนักลดลงแต่หากทำนายผิด น้ำหนักเพิ่มขึ้น

ขั้นตอนที่ 5 การสร้างโมเดลสุดท้าย

สร้างโมเดลสุดท้ายเมื่อฝึกครบ T รอบ จะได้ค่าที่เป็นการรวมผลของ weak learners หลายตัวแบบถ่วงน้ำหนักตามความแม่นยำ จากสูตรดังสมการ (21)

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \quad (21)$$

โดยที่ α_t คือ น้ำหนักของ weak learner ที่ t

h_t คือ weak classifier ในรอบที่ t

2.4.8 Gradient Boosting Classifier

เป็นอัลกอริธึมการเสริมกำลังที่สร้างโมเดลโดยการเรียนรู้จากข้อผิดพลาดที่เกิดขึ้นจากโมเดลก่อนหน้า โดยการปรับน้ำหนักให้กับข้อมูลที่ผิดพลาดในแต่ละรอบ เพื่อให้โมเดลสามารถทำการทำนายที่แม่นยำมากขึ้น เทคนิคนี้เป็นที่นิยมในงานที่ต้องการความแม่นยำสูง เช่น การคาดการณ์พฤติกรรมของลูกค้า Gradient Boosting ทำงานโดยเพิ่มโมเดลย่อย $\mathbf{f_t(x)}$ ทีละตัวเข้ามาเพื่อปรับปรุงผลลัพธ์ที่โมเดลก่อนหน้าทำผิด โดยในแต่ละรอบจะเรียนรู้จาก ค่าความคลาดเคลื่อน (Residuals) หรือ Negative Gradient ของ Loss function [11]

2.4.8.1 หลักการทำงาน (Algorithm) ของ Gradient Boosting Classifier

ขั้นตอนที่ 1 การทำนายค่าเริ่มต้น

การทำนาย ใช้ค่าเดียวกันสำหรับทุก x_i คำนวณจากสูตรดังสมการ (22)

$$F_0(x) = \arg min_\gamma \sum_{i=1}^{n} L(y_i, \gamma) \quad (22)$$

โดยที่ $L(y_i)$ คือ ฟังก์ชันความสูญเสีย (Loss Function)

γ คือ ค่าสเกลของ weak learner

ขั้นตอนที่ 2 การคำนวณ Residuals

คำนวณ Residuals สำหรับข้อมูลทุกจุดดังสมการ (23)

$$r_i^{(t)} = -[\frac{\partial L(y_i, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)}] \quad (23)$$

โดยที่ L คือ Loss function (เช่น Log-loss สำหรับ classification)

$F_{t-1}(x_i)$ คือ ค่าทำนายรอบก่อนหน้า

$r_i^{(t)}$ คือ ค่าที่บอกว่าโมเดลควรปรับขึ้น/ลงเท่าใด

ขั้นตอนที่ 3 การสร้าง Weak Learner

สร้าง Weak Learner ใหม่ $h_t(x)$โดยฝึกให้พยากรณ์ค่า$r_i^{(t)}$ จากนั้นคำนวณค่าขั้นตอน (step size) หรือ learning rate η และค่าสเกล γ_t จากสูตรดังสมการ (24)

$$\gamma_t = \arg min_\gamma \sum_{i=1}^{n} L(y_i, F_{t-1}(x_i) + \gamma h_t(x_i)) \quad (24)$$

ขั้นตอนที่ 4 การอัปเดตโมเดล

อัปเดตโมเดลดังสมการ (25)

$$F_t(x) = F_{t-1}(x) + \eta \cdot \gamma_t h_t(x) \quad (25)$$

โดยที่ η คือ Learning rate (ตัวควบคุมขนาดการอัปเดต เช่น 0.01 – 0.1)

$h_t(x)$ คือ Weak learner ในรอบที่ t

γ_t คือ ค่าสเกลของ weak learner

2.4.9 Ensemble learning

คือ การเรียนรู้ของเครื่อง (Machine Learning) ที่รวมแบบจำลองหลายตัวเข้าด้วยกันเพื่อสร้างแบบจำลองที่มีความแม่นยำและเสถียรมากกว่าแบบจำลองตัวใดตัวหนึ่งเพียงลำพัง แนวคิดหลักของ Ensemble Learning คือ ปัญญารวมหมู่ (wisdom of the crowd) โดยเชื่อว่าแบบจำลองหลายตัวที่มีความแตกต่างกันสามารถรวมกันเพื่อให้ผลลัพธ์ที่ดีกว่าแบบจำลองเดียวอย่างมีนัยสำคัญ [15]

2.4.9.1 หลักการพื้นฐานของ Ensemble Learning

Ensemble Learning ทำงานโดยการฝึกแบบจำลองหลายชุด (ซึ่งอาจเป็นชนิดเดียวกันหรือต่างชนิดกันก็ได้) และรวมผลลัพธ์ของแบบจำลองเหล่านั้นเข้าด้วยกันเพื่อตัดสินผลลัพธ์สุดท้าย วิธีการรวมผลลัพธ์ขึ้นอยู่กับลักษณะของปัญหา เช่น การจำแนกประเภท (classification) หรือการถดถอย (regression)

ในกรณีของ classification การรวมผลอาจใช้การโหวต (majority voting) หรือการรวมความน่าจะเป็น (probability averaging) ขณะที่ regression มักใช้ค่าเฉลี่ยของผลลัพธ์จากแต่ละแบบจำลอง

2.4.9.2 ประเภทของ Ensemble Learning ที่นิยมใช้

2.4.9.2.1 Bagging (Bootstrap Aggregating)

เป็นเทคนิคที่สร้างแบบจำลองหลายตัวจากชุดข้อมูลที่สุ่มซ้ำจากข้อมูลเดิม (sampling with replacement) แล้วนำแบบจำลองที่ได้มาโหวตผลรวมกัน ตัวอย่างที่นิยมมากคือ Random Forest ซึ่งเป็นการรวมของต้นไม้ตัดสินใจ (Decision Trees) หลายต้นที่ถูกฝึกจากชุดข้อมูลที่แตกต่างกัน

2.4.9.2.2 Boosting

เป็นเทคนิคที่สร้างแบบจำลองหลายตัวแบบเรียงลำดับ โดยแต่ละแบบจำลองจะเรียนรู้จากข้อผิดพลาดของแบบจำลองก่อนหน้า เพื่อแก้จุดอ่อนทีละขั้น ทำให้สามารถเพิ่มความแม่นยำได้อย่างมีประสิทธิภาพ ตัวอย่างของ Boosting ได้แก่ AdaBoost, Gradient Boosting, XGBoost, LightGBM, และ CatBoost

2.4.9.2.3 Stacking (Stacked Generalization)

เป็นการรวมแบบจำลองต่างชนิดกัน (heterogeneous models) โดยใช้แบบจำลองอีกตัวหนึ่งซึ่งเรียกว่า meta-model มาทำนายผลจากผลลัพธ์ของแบบจำลองชั้นแรก (base learners) เทคนิคนี้สามารถรวมจุดแข็งของแต่ละโมเดลเข้าด้วยกันได้ดี

2.4.9.2.4 Voting (เฉพาะ classification)

เป็นการนำผลการทำนายจากโมเดลหลายตัวมาโหวตหาคำตอบสุดท้าย แบ่งเป็นสองประเภท ได้แก่

1. Hard Voting: ใช้ผลโหวตส่วนมากจากโมเดล
2. Soft Voting: ใช้ค่า probability เฉลี่ยจากโมเดล แล้วเลือกคลาสที่มีค่าสูงสุด

2.5 ความไม่สมดุลของข้อมูล (Imbalance Data) และการจัดการ

ข้อมูลไม่สมดุล (Imbalanced Data) หมายถึงชุดข้อมูลที่มีการกระจายตัวของคลาสเป้าหมาย (Target Class) อย่างไม่สมดุล [16] ในกรณีการทำนายการยกเลิกใช้บริการของลูกค้า อาจพบว่ามีลูกค้าส่วนใหญ่ที่ยังคงใช้บริการอยู่ (Non-Churn) ขณะที่ลูกค้าที่เลิกใช้บริการ (Churn) มีเพียงส่วนน้อย หากนำข้อมูลในลักษณะนี้ไปฝึกโมเดลโดยตรง อาจทำให้โมเดลเรียนรู้ไปในทิศทางที่ลำเอียง (biased) และมีแนวโน้มคาดการณ์คลาสเสียงข้างน้อยผิดพลาดสูง ซึ่งจะส่งผลต่อประสิทธิภาพของโมเดลโดยรวม โดยเฉพาะในด้าน Recall ของกลุ่มเป้าหมายหลักในการจัดการกับปัญหานี้สามารถใช้เทคนิคได้หลายวิธี โดยทั่วไปแบ่งออกเป็น 3 แนวทางหลัก

2.5.1 การปรับสมดุลข้อมูล (Resampling Techniques)

เช่น การสุ่มเพิ่มข้อมูลของกลุ่มที่มีน้อย (Over-sampling) และการสุ่มลดข้อมูลของกลุ่มที่มีมาก (Under-sampling)

2.5.2 การใช้เทคนิคสังเคราะห์

เป็นวิธีสร้างข้อมูลใหม่ในคลาสเสียงข้างน้อยโดยอาศัยการคำนวณระยะห่างในเชิงเวกเตอร์จากข้อมูลจริง

SMOTE เป็นเทคนิคที่ได้รับความนิยมอย่างแพร่หลายในการแก้ปัญหาข้อมูลไม่สมดุล เนื่องจากสามารถเพิ่มตัวอย่างข้อมูลในคลาสเสียงข้างน้อยได้อย่างสมจริงโดยไม่ทำให้เกิดข้อมูลซ้ำ (duplicated data) มากเกินไป ซึ่งช่วยให้โมเดลสามารถเรียนรู้รูปแบบได้ดียิ่งขึ้น และลดความเอนเอียงของผลลัพธ์ในเชิงสถิติได้อย่างมีประสิทธิภาพ [17]

2.5.3 การปรับปรุงอัลกอริธึมการเรียนรู้

เช่น การใช้ Cost-sensitive Learning ที่ให้ค่าน้ำหนักกับคลาสเสียงข้างน้อยมากขึ้น เพื่อให้โมเดลให้ความสำคัญกับการทำนายคลาสนั้น

2.6 การประเมินแบบจำลอง (Model Evaluation)

การประเมินโมเดล (Model Evaluation) คือกระบวนการวิเคราะห์ประสิทธิภาพ จุดแข็ง และข้อจำกัดของโมเดลการเรียนรู้ของเครื่อง โดยใช้เกณฑ์การประเมินที่หลากหลาย ในระยะเริ่มต้นของการวิจัย การประเมินโมเดลมีความสำคัญอย่างยิ่งเพื่อให้สามารถตรวจสอบประสิทธิผลของโมเดลได้ [18]

2.6.1 Confusion matrix

Confusion Matrix คือเครื่องมือที่ใช้ในการประเมินประสิทธิภาพของโมเดลการจำแนกประเภท (Classification Model) โดยเฉพาะโมเดลที่ทำงานกับปัญหาแบบ Supervised Learning ที่มีผลลัพธ์หลายประเภท เช่น Binary Classification หรือ Multi-class Classification เมทริกซ์นี้จะแสดงในรูปแบบของตาราง 7 เปรียบเทียบระหว่างค่าที่โมเดลทำนาย (Predicted) กับค่าที่เป็นจริง (Actual) เพื่อให้สามารถประเมินว่าโมเดลทำนายได้ถูกต้องหรือไม่ และผิดพลาดอย่างไร [19]

ตาราง 7 โครงสร้างของ Confusion Matrix (กรณี Binary Classification)

Confusion Matrix	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

1. True Positive (TP): โมเดลทำนายว่าเป็นบวก (Positive) และผลจริงก็เป็นบวก ตัวอย่างเช่น โมเดลทำนายว่าลูกค้ายังใช้บริการอยู่ และลูกค้ายังใช้บริการอยู่จริงๆ

2. True Negative (TN): โมเดลทำนายว่าเป็นลบ (Negative) และผลจริงก็เป็นลบ ตัวอย่างเช่น โมเดลทำนายว่าลูกค้ายกเลิกการใช้บริการ และลูกค้ายกเลิกการใช้บริการจริงๆ
3. False Positive (FP) (Type I Error): โมเดลทำนายว่าเป็นบวก แต่ผลจริงเป็นลบ ตัวอย่างเช่น โมเดลทำนายว่าลูกค้ายังใช้บริการอยู่ แต่จริงๆ แล้วลูกค้ายกเลิกการใช้บริการ
4. False Negative (FN) (Type II Error): โมเดลทำนายว่าเป็นลบ แต่ผลจริงเป็นบวก ตัวอย่างเช่น โมเดลทำนายว่าลูกค้ายกเลิกการใช้บริการ แต่จริงๆ แล้วลูกค้ายังใช้บริการอยู่

2.6.2 Accuracy Score

คือ ค่าที่ใช้วัดความถูกต้องโดยรวมของโมเดลในการจำแนกข้อมูลทั้งหมด [20] ไม่ว่าจะเป็นข้อมูลกลุ่มบวกหรือกลุ่มลบ โดยค่านี้คำนวณจากสัดส่วนของจำนวนรายการที่โมเดลทำนายได้ถูกต้องทั้งหมด (ทั้ง True Positive และ True Negative) ต่อจำนวนรายการทั้งหมดในชุดข้อมูล ความหมายของ Accuracy จึงสะท้อนถึงภาพรวมของความสามารถในการทำนายของโมเดลในระดับกว้าง มีช่วงค่าตั้งแต่ 0 ถึง 1 โดยค่า 1 หมายถึงโมเดลสามารถจำแนกข้อมูลได้ถูกต้องทั้งหมดทุกกรณี ส่วนค่า 0 หมายถึงโมเดลไม่สามารถทำนายได้ถูกต้องเลย โดยสามารถคำนวณได้จากสูตรดังสมการ (26)

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (26)$$

2.6.3 Recall Score

Recall Score คือค่าที่ใช้ประเมินความสามารถของโมเดลในการตรวจจับข้อมูลที่อยู่ในกลุ่มบวกได้ครบถ้วนเพียงใด [21] โดยพิจารณาจากอัตราส่วนระหว่างจำนวนข้อมูลที่โมเดลสามารถทำนายได้อย่างถูกต้องว่าอยู่ในกลุ่มบวก (True Positive) กับจำนวนข้อมูลที่เป็นกลุ่มบวกทั้งหมดในความเป็นจริง (True Positive + False Negative) Recall Score มีช่วงค่าระหว่าง 0 ถึง 1 หาก Recall มีค่าที่สูง หรือเข้าใกล้1 แสดงว่าโมเดลสามารถตรวจจับข้อมูลบวกได้อย่างครบถ้วนหรือเกือบทั้งหมด หาก Recall มีค่าต่ำ หมายความว่าโมเดลพลาดการตรวจจับข้อมูลกลุ่มบวกไปจำนวนมาก กล่าวคือยังมีข้อมูลที่ควรถูกจัดให้อยู่ในกลุ่มเป้าหมายแต่กลับไม่ถูกจำแนกอย่างถูกต้อง ดังนั้น Recall จึงสะท้อนให้เห็นถึงความครอบคลุมของโมเดลในการจำแนกข้อมูลบวกทั้งหมด โดยสามารถคำนวณได้จากสูตรดังสมการ (27)

$$Recall = \frac{TP}{TP+FN} \quad (27)$$

2.6.4 Precision Score

Precision Score คือค่าที่ใช้วัดความแม่นยำของโมเดลในการจำแนกข้อมูลกลุ่มบวก โดยเฉพาะอย่างยิ่งในกรณีที่ต้องการทราบว่า เมื่อโมเดลทำนายว่าข้อมูลรายการใดเป็นกลุ่มบวกแล้ว ผลลัพธ์นั้นมีความถูกต้องมากน้อยเพียงใด [22] กล่าวคือ Precision เป็นอัตราส่วนระหว่างจำนวนตัวอย่างที่โมเดลทำนายว่าเป็นบวกและถูกต้องจริง (True Positive) ต่อจำนวนตัวอย่างทั้งหมดที่โมเดลทำนายว่าเป็นบวก ไม่ว่าจะถูกหรือผิด (True Positive + False Positive) ค่า Precision มีช่วงตั้งแต่ 0 ถึง 1 หากค่า Precision มีค่าสูงหรือเข้าใกล้ 1 แสดงว่าโมเดลมีความแม่นยำสูง ในทางตรงกันข้าม หากค่า Precision ต่ำ แสดงว่าโมเดลมักจะทำนายค่าบวกผิดพลาดบ่อย กล่าวคือมีการแจ้งเตือนหรือจำแนกข้อมูลว่าเป็นกลุ่มเป้าหมาย ทั้งที่ในความเป็นจริงไม่ใช่ ดังนั้น Precision Score จึงเป็นค่าที่เหมาะสมในการประเมินโมเดลในสถานการณ์ที่การควบคุมข้อผิดพลาดจากการทำนายว่าข้อมูลเป็นบวกโดยไม่ถูกต้องมีความสำคัญเป็นพิเศษโดยสามารถคำนวณได้จากสูตรดังสมการ (28)

$$Precision = \frac{TP}{TP+FP} \tag{28}$$

2.6.5 F1 Score

F1 Score คือดัชนีที่ใช้ประเมินประสิทธิภาพของโมเดลการจำแนกประเภทโดยเฉพาะในกรณีที่ข้อมูลมีความไม่สมดุลระหว่างคลาส [23] หรือเมื่อทั้งความแม่นยำ (Precision) และความครอบคลุม (Recall) มีความสำคัญอย่างเท่าเทียมกัน การตีความค่า F1 Score จะขึ้นอยู่กับค่าที่ได้จากการคำนวณ ซึ่งมีช่วงค่าตั้งแต่ 0 ถึง 1 โดยที่ค่าที่ใกล้เคียง 1 แสดงว่าโมเดลมีความสามารถในการทำนายได้ดี ทั้งในด้านของการทำนายค่าที่ถูกต้องว่าเป็นกลุ่มบวกและการตรวจจับข้อมูลที่เป็นกลุ่มบวกได้ครบถ้วน หากค่า F1 Score มีค่าต่ำ แสดงว่าโมเดลยังมีข้อบกพร่องด้านใดด้านหนึ่ง หรือทั้งด้านความแม่นยำและความครอบคลุม ดังนั้นค่า F1 Score ที่ต่ำจึงบ่งชี้ว่าโมเดลอาจไม่เหมาะสมสำหรับการใช้งานในสถานการณ์ที่ต้องการความน่าเชื่อถือในการทำนายโดยสามารถคำนวณได้จากสูตรดังสมการ (29)

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{29}$$

2.6.6 Receiver Operating Characteristic

Receiver Operating Characteristic หรือ ROC Curve คือกราฟที่ใช้ประเมินประสิทธิภาพของโมเดลการจำแนกประเภท (Classification Model) [24] โดยเฉพาะโมเดลแบบทวิภาค (Binary Classification) เพื่อแสดงความสามารถของโมเดลในการแยกแยะระหว่างคลาสบวก (Positive) และ คลาสลบ (Negative) โดยไม่ขึ้นอยู่กับเกณฑ์การตัดสิน (Threshold) เช่น ทำนายว่า “จะยกเลิกบริการ” หรือ “ไม่ยกเลิกบริการ”

Area Under the Curve (AUC) คือ พื้นที่ใต้เส้น ROC Curve (AUC-ROC) บอกประสิทธิภาพของโมเดลโดยรวมในการจำแนกคลาส โดยความหมายของAUC ดังตาราง 8

ตาราง 8 ความหมายของ AUC Score

AUC Score	ความหมาย
1.0	ดีเลิศ (แยกคลาสได้สมบูรณ์แบบ)
0.9 - 1.0	ดีมาก
0.8 - 0.9	ดี
0.7 - 0.8	พอใช้
0.6 - 0.7	อ่อน
0.5	แย่ (เท่ากับเดาแบบสุ่ม)
< 0.5	แย่มาก (แยกผิดฝั่ง)

จากตาราง 8 ได้แบ่งช่วงของค่า AUC พร้อมคำอธิบายเชิงคุณภาพอย่างชัดเจน หากแบบจำลองมีค่า AUC เท่ากับ 1.0 หมายความว่าแบบจำลองสามารถแยกแยะคลาสได้อย่างสมบูรณ์แบบ ไม่มีข้อผิดพลาดใดๆในการทำนาย ในขณะที่ช่วงค่า AUC ตั้งแต่ 0.9 ถึง 1.0 แสดงถึงแบบจำลองที่มีประสิทธิภาพดีมาก ส่วนช่วง 0.8 ถึง 0.9 ถือว่าเป็นแบบจำลองที่ดี ในขณะที่ช่วง 0.7 ถึง 0.8 สื่อว่าแบบจำลองนั้นมีประสิทธิภาพในระดับพอใช้ หากค่า AUC อยู่ในช่วง 0.6 ถึง 0.7 จะถือว่าแบบจำลองมีความสามารถในการจำแนกที่ค่อนข้างอ่อน และหากค่าเท่ากับ 0.5 แสดงว่าแบบจำลองไม่มีความสามารถในการจำแนกคลาสได้เลย หรือกล่าวได้ว่ามีความสามารถเทียบเท่ากับการเดาแบบสุ่ม ซึ่งไม่มีประโยชน์เชิงวิเคราะห์ ในกรณีที่ค่า AUC น้อยกว่า 0.5 จะหมายถึงแบบจำลองทำงานได้แย่มาก โดยมีแนวโน้มในการแยกคลาสผิดฝั่ง ซึ่งอาจต้องพิจารณาว่ามีข้อผิดพลาดในการออกแบบโมเดลหรือไม่

2.7 งานวิจัยที่เกี่ยวข้อง

2.7.1 งานวิจัยเรื่อง การวิเคราะห์การทำนายการขอยกเลิกใช้บริการสำหรับลูกค้าบริษัทโทรคมนาคมโดยวิธีการเรียนรู้ของเครื่อง

โดย Ying Wei (2024) มีวัตถุประสงค์เพื่อศึกษาปัจจัยที่ส่งผลต่อการตัดสินใจยกเลิกการใช้บริการของลูกค้าในบริษัทโทรคมนาคม และพัฒนาโมเดลการทำนายที่สามารถระบุลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการได้อย่างแม่นยำ

ในการศึกษานี้ ผู้วิจัยได้รวบรวมข้อมูลจากฐานข้อมูลของบริษัทโทรคมนาคม ซึ่งประกอบด้วยข้อมูลพฤติกรรมการใช้งาน ข้อมูลทางประชากรศาสตร์ และข้อมูลการติดต่อสื่อสาร

กับลูกค้า จากนั้นได้ทำการวิเคราะห์และเตรียมข้อมูลเพื่อให้พร้อมสำหรับการสร้างโมเดลการทำนาย โดยใช้เทคนิคการเรียนรู้ของเครื่องหลายรูปแบบ เช่น Logistic Regression, Decision Tree, และ Random Forest เพื่อเปรียบเทียบประสิทธิภาพของแต่ละโมเดล

ผลการวิจัยพบว่า โมเดล Random Forest มีความแม่นยำสูงสุดในการทำนายการยกเลิกการใช้บริการของลูกค้า โดยสามารถระบุปัจจัยที่มีผลต่อการตัดสินใจยกเลิกบริการได้อย่างชัดเจน เช่น ระยะเวลาการเป็นลูกค้า, จำนวนการร้องเรียน, และรูปแบบการใช้งานบริการ ผลการศึกษานี้สามารถนำไปใช้ในการพัฒนากลยุทธ์การรักษาลูกค้าและปรับปรุงคุณภาพการให้บริการของบริษัทโทรคมนาคม เพื่อลดอัตราการยกเลิกใช้บริการและเพิ่มความพึงพอใจของลูกค้าในระยะยาว

2.7.2 งานวิจัยเรื่อง Customer Churn Data Analysis Using Data Mining

โดย Xingyuan Jiang (2024) มีวัตถุประสงค์เพื่อวิเคราะห์และทำนายการสูญเสียลูกค้าขององค์กร โดยใช้เทคนิคการทำเหมืองข้อมูล เพื่อช่วยให้องค์กรสามารถพัฒนากลยุทธ์ในการรักษาลูกค้าและเพิ่มความพึงพอใจของลูกค้าได้อย่างมีประสิทธิภาพ

ในการศึกษานี้ ผู้วิจัยได้เก็บรวบรวมชุดข้อมูลที่เกี่ยวข้องกับการสูญเสียลูกค้าจากองค์กรจริง จากนั้นทำการเตรียมและทำความสะอาดข้อมูลเพื่อให้มั่นใจในความถูกต้องและความสอดคล้องของข้อมูล ต่อมาได้ใช้เทคนิคการทำเหมืองข้อมูล เช่น การวิเคราะห์เปรียบเทียบ (contrastive analysis) และการวิเคราะห์ปัจจัย (factor analysis) เพื่อวิเคราะห์ข้อมูลดังกล่าว นอกจากนี้ ยังได้ดำเนินการเลือกคุณลักษณะ (feature selection) และฝึกอบรมโมเดลเพื่อสร้างแบบจำลองการทำนายการสูญเสียลูกค้า

ผลการวิจัยพบว่า โมเดลที่พัฒนาขึ้นสามารถทำนายการสูญเสียลูกค้าได้อย่างแม่นยำ ซึ่งช่วยให้องค์กรสามารถระบุลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการได้ล่วงหน้า นอกจากนี้ การวิเคราะห์ยังเผยให้เห็นถึงปัจจัยที่ส่งผลต่อการสูญเสียลูกค้า ซึ่งสามารถนำไปใช้ในการพัฒนากลยุทธ์การจัดการความสัมพันธ์กับลูกค้า การให้บริการที่ปรับแต่งเฉพาะบุคคล และกิจกรรมทางการตลาดที่เหมาะสม เพื่อรักษาลูกค้าและเพิ่มความสามารถในการทำกำไรขององค์กร

2.7.3 งานวิจัยเรื่อง Customer Churn Analysis Prediction Based on Cluster Analysis and Machine Learning Algorithms

โดย Yuchen Jiang (2024) มีวัตถุประสงค์เพื่อศึกษาผลกระทบของการสูญเสียลูกค้า (customer churn) ต่อการเติบโตและความสามารถในการทำกำไรของบริษัทหรือองค์กร และพัฒนารูปแบบการทำนายเพื่อระบุลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการ

ในการศึกษานี้ ผู้วิจัยได้ใช้ชุดข้อมูลของลูกค้าธนาคารที่มีข้อมูลส่วนบุคคล การใช้งานบริการ และข้อมูลการชำระเงิน รวมทั้งหมด 7,043 ตัวอย่าง โดยแต่ละตัวอย่างมี 21 คุณลักษณะ จากนั้นได้ทำการวิเคราะห์กลุ่มลูกค้าโดยใช้เทคนิคการจัดกลุ่มแบบ k-means เพื่อแบ่งกลุ่มลูกค้าตามลักษณะและพฤติกรรมที่คล้ายคลึงกัน หลังจากนั้น ได้สร้างโมเดลการเรียนรู้ของเครื่องเพื่อทำนายความน่าจะเป็นของการสูญเสียลูกค้า โดยใช้โมเดล Decision Tree และ Random Forest

ผลการวิจัยพบว่า โมเดล Decision Tree มีความแม่นยำในการทำนายสูงถึง 99% ในขณะที่ Logistic regression มีความแม่นยำอยู่ที่ 90% ซึ่งแสดงให้เห็นว่า โมเดล Decision Tree มีประสิทธิภาพดีกว่าในการทำนายการสูญเสียลูกค้า อย่างไรก็ตาม ผู้วิจัยได้ชี้ให้เห็นว่าผลการทำนายอาจแตกต่างกันไปตามโมเดลที่ใช้ ดังนั้น ควรพิจารณาและวิเคราะห์หลายโมเดลในการทำนายการสูญเสียลูกค้า เพื่อให้ได้ผลลัพธ์ที่แม่นยำและเหมาะสมที่สุด

2.7.4 **งานวิจัยเรื่อง** Machine Learning Approach to Predict E-commerce Customer Satisfaction Score

โดย พงษ์ธนินท์ วังคีอาจ มีวัตถุประสงค์เพื่อศึกษาประสิทธิภาพของการใช้เทคนิคการเรียนรู้ของเครื่องในการทำนายคะแนนความพึงพอใจของลูกค้าในธุรกิจอีคอมเมิร์ซ โดยใช้ชุดข้อมูลการขายจาก Olist ซึ่งเป็นแพลตฟอร์มอีคอมเมิร์ซในบราซิล

ในการศึกษานี้ ผู้วิจัยได้ใช้โมเดลการเรียนรู้ของเครื่องหลายรูปแบบในการวิเคราะห์ข้อมูลที่มีอยู่ เพื่อทำนายคะแนนความพึงพอใจของลูกค้า โดยพิจารณาปัจจัยต่าง ๆ ที่อาจมีผลต่อความพึงพอใจ เช่น เวลาที่ใช้ในการจัดส่งสินค้า ความถูกต้องของคำสั่งซื้อ และมูลค่าของคำสั่งซื้อ โมเดลที่ใช้ในการวิเคราะห์รวมถึง Random Forest และ Support Vector Machines (SVM) ซึ่งเป็นเทคนิคที่ได้รับความนิยมในการทำนายและจัดหมวดหมู่ข้อมูล

ผลการศึกษาพบว่า โมเดล Random Forest มีความแม่นยำสูงสุดในการทำนายคะแนนความพึงพอใจของลูกค้า โดยสามารถทำนายได้อย่างถูกต้องในระดับที่น่าพอใจ นอกจากนี้การวิเคราะห์ยังชี้ให้เห็นว่า ปัจจัยที่มีผลกระทบมากที่สุดต่อความพึงพอใจของลูกค้า ได้แก่ เวลาที่ใช้ในการจัดส่งสินค้า และความถูกต้องของคำสั่งซื้อ ซึ่งข้อมูลเหล่านี้สามารถนำไปใช้ในการปรับปรุงกระบวนการทางธุรกิจ เพื่อเพิ่มความพึงพอใจและความภักดีของลูกค้าในอนาคต

2.7.5 **งานวิจัยเรื่อง** Customer Experience Analytics for Thailand Hospitals

โดย พนิดา กาศกลางดอน มีวัตถุประสงค์เพื่อวิเคราะห์ประสบการณ์ของผู้ใช้บริการโรงพยาบาลในประเทศไทย โดยศึกษาความคิดเห็นและข้อเสนอแนะจากผู้ป่วยและผู้รับบริการ เพื่อทำความเข้าใจปัจจัยที่ส่งผลต่อความพึงพอใจและประสบการณ์โดยรวมของผู้ใช้บริการ

ในการศึกษานี้ ผู้วิจัยได้เก็บรวบรวมข้อมูลความคิดเห็นของผู้ใช้บริการจากแหล่งข้อมูลออนไลน์ เช่น รีวิวบนเว็บไซต์และสื่อสังคมออนไลน์ จากนั้นนำข้อมูลที่ได้มาวิเคราะห์โดยใช้เทคนิคการประมวลผลภาษาธรรมชาติ (Natural Language Processing) และการวิเคราะห์ความรู้สึก (Sentiment Analysis) เพื่อระบุแนวโน้มและปัจจัยที่มีผลต่อประสบการณ์ของผู้ใช้บริการ

ผลการวิจัยพบว่าปัจจัยที่มีผลต่อประสบการณ์ของผู้ใช้บริการโรงพยาบาลในประเทศไทย ได้แก่ คุณภาพการบริการของบุคลากรทางการแพทย์ ความสะดวกสบายของสถานที่ และระยะเวลารอคอยในการรับบริการ นอกจากนี้ การวิเคราะห์ยังชี้ให้เห็นถึงความสำคัญของการสื่อสารระหว่างผู้ให้บริการและผู้รับบริการในการสร้างประสบการณ์ที่ดี

2.7.6 **งานวิจัยเรื่อง** Support Vector Machine-Based Prediction Model for Healthcare Workforce Transition Success Under Decentralization

โดย Atiya Sarakshetrin (2025) มีวัตถุประสงค์เพื่อพัฒนาแบบจำลองการพยากรณ์ความสำเร็จในการโยกย้ายบุคลากรด้านสาธารณสุขในบริบทของการกระจายอำนาจในประเทศไทย ซึ่งถือเป็นประเด็นสำคัญที่ส่งผลต่อประสิทธิภาพของระบบบริการสุขภาพ งานวิจัยมุ่งเน้นไปที่การคัดเลือกตัวแปรที่มีอิทธิพลต่อความสำเร็จในการเปลี่ยนผ่าน และสร้างแบบจำลองที่สามารถใช้สนับสนุนการตัดสินใจของหน่วยงานที่เกี่ยวข้อง

ผู้วิจัยใช้ข้อมูลจากบุคลากรด้านสาธารณสุขที่ย้ายจากส่วนกลางไปยังองค์กรปกครองส่วนท้องถิ่น โดยมีการรวบรวมข้อมูลในหลากหลายด้าน เช่น ข้อมูลส่วนบุคคล ตำแหน่งงาน ประสบการณ์ และความคิดเห็นต่อระบบการกระจายอำนาจ จากนั้นได้ดำเนินการเลือกตัวแปรสำคัญโดยใช้เทคนิค Information Gain และสร้างโมเดลพยากรณ์ด้วยอัลกอริทึม Machine Learning 5 ประเภท ได้แก่ Support Vector Machine (SVM), Random Forest, Decision Tree, Naive Bayes และ Logistic Regression โดยที่ SVM ถูกเลือกเป็นโมเดลหลักเนื่องจากมีความสามารถในการแยกข้อมูลที่ซับซ้อนได้ดี และเหมาะกับข้อมูลที่มีมิติมาก ทั้งนี้ยังได้ทำการประเมินผลด้วยวิธี Cross-validation และใช้ค่า Accuracy, Precision, Recall และ F1-score เป็นเกณฑ์ในการเปรียบเทียบประสิทธิภาพ

ผลการศึกษาแสดงให้เห็นว่าโมเดล SVM มีประสิทธิภาพดีที่สุดในการพยากรณ์ โดยให้ค่า Accuracy สูงที่สุดเมื่อเทียบกับโมเดลอื่น ๆ นอกจากนี้ยังพบว่าตัวแปรที่มีอิทธิพลต่อความสำเร็จในการโยกย้ายมากที่สุด ได้แก่ ความพึงพอใจในงาน โอกาสในการพัฒนาอาชีพ และ

การสนับสนุนจากองค์กร การวิจัยนี้จึงมีศักยภาพในการนำไปประยุกต์ใช้จริงเพื่อช่วยในการวางแผนนโยบายการบริหารบุคลากรภาครัฐอย่างมีประสิทธิภาพในระยะยาว

2.7.7 **งานวิจัยเรื่อง** Gradient Boosting Based Model for Elderly Heart Failure, Aortic Stenosis, and Dementia Classification

โดย Khomkrit Yongcharoenchaiyasit (2023) มีวัตถุประสงค์เพื่อพัฒนาแบบจำลองการจำแนกโรคหัวใจล้มเหลว (Heart Failure), โรคลิ้นหัวใจตีบ (Aortic Stenosis) และภาวะสมองเสื่อม (Dementia) ในผู้สูงอายุ โดยใช้ข้อมูลทางคลินิกจากโรงพยาบาลในประเทศไทย เพื่อเพิ่มความแม่นยำในการวินิจฉัยโรคที่มักเกิดขึ้นร่วมกันในกลุ่มผู้สูงวัย ซึ่งการวินิจฉัยล่าช้าหรือไม่แม่นยำอาจนำไปสู่ภาวะแทรกซ้อนและคุณภาพชีวิตที่ลดลง

ผู้วิจัยได้รวบรวมข้อมูลจากโรงพยาบาลเชียงรายประชานุเคราะห์ ประกอบด้วยตัวแปร 21 รายการ เช่น อายุ เพศ อัตราการเต้นของหัวใจ ค่าความดัน และค่าทางห้องปฏิบัติการ จากนั้นใช้กระบวนการคัดเลือกคุณลักษณะด้วย Recursive Feature Elimination (RFE) และวิเคราะห์ด้วยโมเดล Machine Learning หลายประเภท ได้แก่ Gradient Boosting Classifier (GBC), Random Forest (RF), Extra Trees (ET), Support Vector Machine (SVM), K-Nearest Neighbors (KNN) และ Decision Tree (DT) โดย Gradient Boosting ถูกเลือกเป็นโมเดลหลัก เนื่องจากมีความสามารถสูงในการจัดการกับข้อมูลไม่สมดุล และมีความแม่นยำในการจำแนกหลายคลาส ทั้งยังใช้เทคนิค Optuna สำหรับการปรับค่าพารามิเตอร์ เพื่อเพิ่มประสิทธิภาพโมเดล

ผลการทดลองแสดงให้เห็นว่าโมเดล Gradient Boosting มีประสิทธิภาพดีที่สุด โดยให้ค่า Accuracy เฉลี่ยสูงกว่าโมเดลอื่น ๆ ในทั้ง 3 กลุ่มโรค โดยเฉพาะในการจำแนกผู้ป่วยโรคหัวใจล้มเหลวและภาวะสมองเสื่อม ซึ่งเป็นกลุ่มที่มักมีข้อมูลซับซ้อน โมเดลที่พัฒนาขึ้นนี้จึงสามารถนำไปประยุกต์ใช้จริงในระบบช่วยตัดสินใจทางคลินิก เพื่อส่งเสริมการวินิจฉัยโรคในผู้สูงอายุได้อย่างมีประสิทธิภาพมากขึ้น และลดความเสี่ยงจากการวินิจฉัยผิดพลาด

จากการศึกษางานวิจัยที่เกี่ยวข้องจำนวน 7 ฉบับ พบว่าแต่ละงานวิจัยเลือกใช้โมเดล Machine Learning ที่หลากหลายเพื่อเปรียบเทียบประสิทธิภาพในการพยากรณ์ ซึ่งเป็นบริบทที่มีลักษณะข้อมูลซับซ้อนและมีหลายมิติ โดยเฉพาะในบริบทของการทำนายการยกเลิกการใช้บริการ พบว่าโมเดล Random Forest, Decision Tree และ Logistic Regression เป็นที่นิยมในการใช้งาน เนื่องจากตีความง่ายและให้ผลลัพธ์ที่แม่นยำในข้อมูลเชิงพฤติกรรมและประชากรศาสตร์ ขณะที่ Support Vector Machine และ Gradient Boosting Classifier มักถูกเลือกใช้ในกรณีที่ต้องการความสามารถในการแยกข้อมูลที่มีความซับซ้อนสูง และมีความแม่นยำในการจำแนก

ข้อมูลหลายคลาส โดยเฉพาะในข้อมูลไม่สมดุลหรือมีความแตกต่างสูงระหว่างกลุ่ม ตัวอย่างเช่น งานวิจัยของ Sarakshetrin (2025) และ Yongcharoenchaiyasit (2023) ที่เลือกใช้ SVM และ Gradient Boosting ตามลำดับ เนื่องจากความสามารถเชิงเทคนิคในการจัดการข้อมูลที่ซับซ้อนและไม่สมดุลได้อย่างดี ส่วน K-Nearest Neighbors และ AdaBoost Classifier ก็ถูกนำมาใช้เพื่อเปรียบเทียบผลลัพธ์กับโมเดลอื่น ๆ ซึ่งงานวิจัยของ Jiang (2024) และงานของ Ying Wei (2024) ต่างชี้ให้เห็นถึงความสำคัญของการวิเคราะห์เปรียบเทียบหลายโมเดล เพื่อให้ได้แบบจำลองที่มีประสิทธิภาพสูงสุด ด้วยเหตุนี้การศึกษานี้จึงเลือกใช้โมเดลทั้ง 7 ได้แก่ Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbor และ AdaBoost Classifier เพื่อให้ครอบคลุมลักษณะโมเดลที่มีทั้งความสามารถเชิงตีความ ความแม่นยำ และความยืดหยุ่นในการประมวลผลข้อมูลที่ซับซ้อน และเพื่อเปรียบเทียบประสิทธิภาพในมิติต่าง ๆอย่างรอบด้านที่สุด

บทที่ 3
วิธีการดำเนินงานวิจัย

บทนี้อธิบายกระบวนการและเครื่องมือที่ใช้ในการวิเคราะห์ข้อมูล โดยครอบคลุมตั้งแต่การจัดเตรียมชุดข้อมูล การทำความสะอาดข้อมูล (Data Cleansing) การวิเคราะห์ข้อมูลเบื้องต้น (Exploratory Data Analysis: EDA) การแสดงภาพข้อมูล การแบ่งชุดข้อมูลสำหรับฝึกสอนและทดสอบ การสร้างแบบจำลอง การประเมินผล และการปรับแต่งแบบจำลอง (Fine-tune) เพื่อใช้ในกระบวนการรวมโมเดล (Ensemble Learning)

การศึกษานี้มีวัตถุประสงค์เพื่อจำแนกการสูญเสียลูกค้าที่เกิดจากการยกเลิกการใช้บริการ และวิเคราะห์ว่าแบบจำลองใดสามารถทำนายการยกเลิกได้อย่างมีประสิทธิภาพมากที่สุด โดยเริ่มต้นจากการทำความเข้าใจชุดข้อมูล เพื่อให้สามารถเห็นภาพรวมและค้นหาข้อมูลเชิงลึก จากนั้นดำเนินการทำความสะอาดข้อมูลและวิเคราะห์ข้อมูลเบื้องต้น เพื่อตรวจสอบรูปแบบและความสัมพันธ์ของข้อมูล ก่อนนำไปแบ่งออกเป็นชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบอย่างเหมาะสม

ในขั้นตอนการสร้างแบบจำลอง จะเลือกใช้อัลกอริธึมต่างๆ และประเมินประสิทธิภาพของแบบจำลองโดยใช้ข้อมูลที่ผ่านการเตรียมมาแล้ว ทั้งนี้ผลการประเมินจะถูกนำมาเปรียบเทียบกับวิธีการพื้นฐาน (Baseline Method) เพื่อพิจารณาความน่าเชื่อถือของแบบจำลอง หากแบบจำลองใดแสดงผลลัพธ์ที่น่าพึงพอใจ จึงจะนำไปปรับเทียบและรวมเข้ากับแบบจำลองอื่นในขั้นตอนของ Ensemble Learning ซึ่งมีเป้าหมายเพื่อเพิ่มความแม่นยำในการทำนาย และสามารถนำไปประยุกต์ใช้จริงในการวิเคราะห์การยกเลิกบริการของลูกค้าต่อไปในอนาคต

3.1 ชุดข้อมูล

ข้อมูลที่นำมาศึกษาในงานวิจัยครั้งนี้ เป็นข้อมูลจริงที่ได้รับจากบริษัทผู้ให้บริการด้านการจัดการทรัพยากรบุคคลแห่งหนึ่ง ซึ่งเป็นผู้ให้บริการแพลตฟอร์มสำหรับการจัดการทรัพยากรบุคคล และมีการเก็บค่าบริการรายเดือนตามแผนการใช้งานและประเภทบริการที่ลูกค้าต้องการ โดยข้อมูลที่เก็บรวบรวมมีรายละเอียดเกี่ยวกับการใช้งานบริการต่าง ๆ ของลูกค้าในระยะเวลาที่ผ่านมาดังตัวอย่างในภาพประกอบ 2

```
[ ] df.head()
```

	ClientId	CustAccount	Tenure	HasTaxId	HasEmail	HasPhoneNumber	PaidInvoiceCount	OverdueInvoiceCount	PendingInvoiceCount	RejectInvoiceCount	DayFromLastPayment	PlanName
0	1003	C1800445	5	Yes	Yes	Yes	108	0	3	2	45	Basic (Monthly)
1	1005	50008860-0	5	Yes	Yes	Yes	24	4	4	1	71	Pro
2	1006	C1800029	5	Yes	Yes	Yes	4857	0	12	2	0	Pro
3	1011	C1800014	5	Yes	Yes	Yes	53	0	2	1	42	Basic (Monthly)
4	1018	C1800463	5	Yes	Yes	Yes	55	0	8	1	46	Standard (Monthly)

ภาพประกอบ 2 ตัวอย่างข้อมูลจากชุดข้อมูล

ชุดข้อมูลประกอบด้วยข้อมูลจำนวนประมาณ 1,597 แถว (ภาพประกอบ 3) โดยข้อมูลครอบคลุมช่วงเวลาตั้งแต่ปี 2022 ถึงปี 2024 ซึ่งบันทึกข้อมูลของลูกค้าแต่ละราย รายละเอียดของข้อมูลประกอบด้วยทั้งหมด 19 คอลัมน์ (ภาพประกอบ 4) ซึ่งแต่ละคอลัมน์มีความหมายและตัวแปรที่เกี่ยวข้อง ซึ่งจะอธิบายไว้ในตาราง 1 โดยข้อมูลนี้มีความสำคัญในการวิเคราะห์การยกเลิกการใช้บริการของลูกค้า (Customer Churn) และสามารถนำไปใช้ในการทำนายพฤติกรรมการยกเลิกบริการของลูกค้าต่อไป

```
[ ] df.shape

(1597, 19)
```

ภาพประกอบ 3 ขนาดของชุดข้อมูล

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1597 entries, 0 to 1596
Data columns (total 19 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   ClientId                   1597 non-null   int64
 1   CustAccount                1597 non-null   object
 2   Tenure                     1597 non-null   int64
 3   HasTaxId                   1597 non-null   object
 4   HasEmail                   1597 non-null   object
 5   HasPhoneNumber             1597 non-null   object
 6   PaidInvoiceCount           1597 non-null   int64
 7   OverdueInvoiceCount        1597 non-null   int64
 8   PendingInvoiceCount        1597 non-null   int64
 9   RejectInvoiceCount         1597 non-null   int64
 10  DayFromLastPayment         1597 non-null   int64
 11  PlanName                   1597 non-null   object
 12  IsChurn                    1597 non-null   object
 13  Qty                        1597 non-null   int64
 14  Contract                   1597 non-null   object
 15  TotalIncome                1597 non-null   float64
 16  AverageRevenuePerMonth     1597 non-null   float64
 17  PaymentMethod              1597 non-null   object
 18  PercentSatisfactionScore   1597 non-null   int64
dtypes: float64(2), int64(9), object(8)
memory usage: 237.2+ KB
```

ภาพประกอบ 4 ชื่อคอลัมน์และชนิดของข้อมูล

3.2 **การทำความสะอาดข้อมูล** (Data Cleansing)

การทำความสะอาดข้อมูล (Data Cleansing) เป็นขั้นตอนสำคัญในกระบวนการวิเคราะห์ข้อมูลและการสร้างแบบจำลองทางการเรียนรู้ของเครื่อง เนื่องจากข้อมูลดิบที่ได้มาในรูปแบบจริงมักมีความไม่สมบูรณ์ เช่น ค่าว่าง ข้อมูลผิดรูปแบบ หรือข้อมูลที่ไม่เหมาะสมต่อการวิเคราะห์ หากไม่จัดการปัญหาเหล่านี้อย่างเหมาะสม อาจส่งผลกระทบต่อความแม่นยำของแบบจำลองและทำให้ผลการวิเคราะห์คลาดเคลื่อนได้ ในงานวิจัยนี้ การทำความสะอาดข้อมูลประกอบด้วย 3 ขั้นตอนหลัก ได้แก่ การจัดการข้อมูลที่มีค่าว่าง (Missing Values) เพื่อป้องกันไม่ให้โมเดลเรียนรู้จากข้อมูลที่ไม่สมบูรณ์ การแปลงข้อมูลตัวอักษรเป็นตัวเลข เพื่อให้สามารถนำไปประมวลผลโดยอัลกอริทึมทางคณิตศาสตร์ได้อย่างถูกต้องและ การแบ่งกลุ่มข้อมูลจำพวกตัวเลข (Numerical Binning) เพื่อช่วยลดความผันแปรของข้อมูลและเพิ่มประสิทธิภาพของโมเดลในการเรียนรู้จากข้อมูลดังกล่าว

3.2.1 **การจัดการข้อมูลที่มีค่าว่าง** (Missing Values)

จากการตรวจสอบข้อมูลทั้งหมดจำนวน 1,597 แถว เนื่องจากเป็นข้อมูลที่ดึงมาจากฐานข้อมูล และมีการกรองข้อมูลที่เป็นค่าว่าง (null) ออกแล้ว จึงทำให้ชุดข้อมูลนี้ไม่มีค่าว่าง ดังแสดงในภาพประกอบ 5

```
df.isnull().sum()
```

	0
ClientId	0
CustAccount	0
Tenure	0
HasTaxId	0
HasEmail	0
HasPhoneNumber	0
PaidInvoiceCount	0
OverdueInvoiceCount	0
PendingInvoiceCount	0
RejectInvoiceCount	0
DayFromLastPayment	0
PlanName	0
IsChurn	0
Qty	0
Contract	0
TotalIncome	0
AverageRevenuePerMonth	0
PaymentMethod	0
PercentSatisfactionScore	0

ภาพประกอบ 5 จำนวนข้อมูลที่เป็นค่าว่าง (null) ของแต่ละคอลัมน์

3.2.2 การแปลงข้อมูลตัวอักษรเป็นตัวเลข

ในการเตรียมข้อมูลเพื่อใช้ในการฝึกแบบจำลองการเรียนรู้ของเครื่อง (Machine Learning) จำเป็นต้องดำเนินการแปลงข้อมูลที่อยู่ในรูปแบบเชิงหมวดหมู่ (Categorical Data) ให้อยู่ในรูปแบบเชิงตัวเลข (Numerical Data) เพื่อให้สามารถประมวลผลได้โดยอัลกอริธึมต่าง ๆ อย่างมีประสิทธิภาพ ในการศึกษานี้ได้เลือกใช้เทคนิค Label Encoding สำหรับการแปลงค่าหมวดหมู่ โดยการแทนค่าหมวดหมู่แต่ละรายการด้วยตัวเลขจำนวนเต็ม เช่น 0, 1, 2 ตามลำดับที่ปรากฏในข้อมูล ซึ่งช่วยลดจำนวนฟีเจอร์โดยรวม และมีความรวดเร็วในการประมวลผล เหมาะสำหรับการทดลองเชิงเบื้องต้น

ตัวแปรที่แปลงค่าด้วยวิธีดังกล่าวได้แก่ PlanName, HasTaxId, HasEmail, HasPhoneNumber, IsChurn, Contract, และ PaymentMethod อย่างไรก็ตัวแปรบางชนิด เช่น PaymentMethod เป็นตัวแปรเชิงหมวดหมู่แบบไม่มีลำดับ (Nominal Categorical Variable) ดังนั้นตามแนวทางปฏิบัติที่เหมาะสมควรใช้ One-Hot Encoding เพื่อหลีกเลี่ยงการแฝงลำดับที่ไม่มีอยู่จริงลงไปในข้อมูล เช่น การแทนประเภทการชำระเงิน Bank Transfer, Credit Card, QR Code ด้วยตัวเลข 0, 1, 2 อาจทำให้โมเดลเข้าใจผิดว่า QR Code มีอันดับสูงกว่า Credit Card ทั้งที่ในความเป็นจริงไม่เกี่ยวข้องกันแต่ในการศึกษาครั้งนี้ ได้เลือกใช้ Label Encoding กับตัวแปรเชิงหมวดหมู่แบบไม่มีลำดับ เพื่อควบคุมจำนวนฟีเจอร์ไม่ให้เพิ่มมากเกินไป ซึ่งอาจส่งผลให้เกิดการกระจายของข้อมูล (Data Sparsity) หากใช้ One-Hot Encoding โดยเฉพาะในกรณีที่จำนวนข้อมูลไม่มากเพียงพอ หรือมีข้อจำกัดด้านประสิทธิภาพการประมวลผล ทั้งนี้เพื่อความเรียบง่ายของโมเดลในระยะเริ่มต้น และอำนวยความสะดวกในการเปรียบเทียบโมเดลหลายประเภทบนชุดข้อมูลเดียวกัน [25]

ในการศึกษาต่อยอด อาจพิจารณานำตัวแปรเชิงหมวดหมู่แบบไม่มีลำดับไปแปลงเป็นรูปแบบ One-Hot Encoding และเปรียบเทียบผลลัพธ์ของโมเดลทั้งสองกรณี เพื่อประเมินว่าการแปลงข้อมูลในลักษณะที่เหมาะสมกับลักษณะของตัวแปรจริง ๆ จะส่งผลให้แบบจำลองมีประสิทธิภาพเพิ่มขึ้นหรือไม่ ซึ่งจะช่วยเสริมความถูกต้องในการตัดสินใจเลือกวิธีการแปลงค่าที่เหมาะสมที่สุด

3.2.2.1 แผนการใช้บริการ (Plan Name)

Plan Name คือ แผนบริการที่ใช้ในการศึกษานี้ประกอบด้วยทั้งหมด 25 รูปแบบ โดยจะแสดงดังตาราง 9 และภาพประกอบ 6 แต่ละแผนยังมีการแบ่งย่อยตามระยะเวลาการสมัคร

ใช้งาน เช่น รายเดือน (Monthly), รายไตรมาส (Quarterly) หรือรูปแบบอื่น ๆ ที่บริษัทจัดทำขึ้น โดยในกระบวนการ

นอกจากนี้ ยังได้ดำเนินการจัดลำดับความสำคัญของแผนบริการ (Plan Priority) โดยเรียงจากแผนที่มีค่าใช้จ่ายต่ำที่สุดไปยังแผนที่มีค่าใช้จ่ายสูงที่สุด ซึ่งกำหนดลำดับใหม่เริ่มต้นตั้งแต่ลำดับที่ 1 เพื่อให้สอดคล้องกับการวิเคราะห์ข้อมูลเชิงลำดับ (Ordinal Analysis) และสามารถนำไปใช้ในขั้นตอนการวิเคราะห์เชิงเปรียบเทียบหรือการเรียนรู้ของโมเดล

ตาราง 9 ลำดับแผนการใช้บริการ

ลำดับ	แผนการใช้บริการ
1	FREE 10
2	FREE (Monthly)
3	FREE (Quarterly)
4	FREE (Yearly)
5	Lite (Monthly)
6	Lite (Quarterly)
7	Lite (Yearly)
8	Basic (Monthly)
9	Basic (Quarterly)
10	Basic (Yearly)
11	Standard (Monthly)
12	Standard (Quarterly)
13	Standard (Yearly)
14	Pro (Monthly)
15	Pro (Quarterly)
16	Pro (Yearly)
17	Pro+ Classic (Monthly)
18	Pro+ Classic (Quarterly)
19	Pro+ Classic (Yearly)
20	Pro+ (Monthly)
21	Pro+ (Quarterly)

ลำดับ	แผนการใช้บริการ
22	Pro+ (Yearly)
23	Enterprise (Monthly)
24	Enterprise (Quarterly)
25	Enterprise (Yearly)

```
[ ] from pandas.api.types import CategoricalDtype

    package_order = [
        'FREE 10','FREE (Monthly)','FREE (Quarterly)','FREE (Yearly)','Lite (Monthly)','Lite (Quarterly) Lite (Yearly)',
        'Basic (Monthly)','Basic (Quarterly)','Basic (Yearly)','Standard (Monthly)','Standard (Quarterly)','Standard (Yearly)',
        'Pro (Monthly)','Pro (Quarterly)','Pro (Yearly)','Pro+ Classic (Monthly)','Pro+ Classic (Quarterly)','Pro+ Classic (Yearly)',
        'Pro+ (Monthly)','Pro+ (Quarterly)','Pro+ (Yearly)','Enterprise (Monthly)','Enterprise (Quarterly)','Enterprise (Yearly)'
    ]

    package_mapping = {package: i + 1 for i, package in enumerate(package_order)}

    print(package_mapping)

    df["PlanName"] = df["PlanName"].map(package_mapping)

    df[["PlanName"]].head()
```

ภาพประกอบ 6 โค้ดสำหรับแปลง PlanName เป็นตัวเลข

3.2.2.2 HasTaxId, HasEmail, HasPhoneNumber, IsChurn

ข้อมูลจำนวน 3 คอลัมน์ในชุดข้อมูลนี้มีลักษณะเป็นข้อมูลประเภทตรรกะ (Boolean) โดยประกอบด้วยค่าที่เป็น Yes และ No เพื่อให้สามารถนำข้อมูลดังกล่าวไปใช้ในการประมวลผลด้วยอัลกอริธึมการเรียนรู้ของเครื่อง (Machine Learning) ได้อย่างเหมาะสม จึงได้มีการแปลงค่าดังกล่าวให้อยู่ในรูปแบบตัวเลข โดยกำหนดให้ Yes แทนด้วยค่า 1 และ No แทนด้วยค่า 0 ดังภาพประกอบ 7

```
[ ] binary_cols = ['HasTaxId', 'HasEmail', 'HasPhoneNumber']

    df[binary_cols] = df[binary_cols].apply(lambda x: x.map({'Yes': 1, 'No': 0}))

    df[binary_cols].head()
```

ภาพประกอบ 7 โค้ดสำหรับแปลง HasTaxId, HasEmail, HasPhoneNumber, IsChurn เป็นตัวเลข

3.2.2.3 Contract

คอลัมน์ Contract แสดงประเภทของระยะเวลาการเรียกเก็บเงินตามใบแจ้งหนี้ของลูกค้า โดยมีรูปแบบข้อมูลเป็นข้อความ เช่น 1 month, 3 months, 6 months, 12 months และ 1 Year ซึ่งทั้งหมดสื่อถึงระยะเวลาการชำระค่าบริการตามรอบบิล เพื่อให้ข้อมูลดังกล่าวมีความเป็นระเบียบและสามารถนำไปประมวลผลร่วมกับโมเดลการเรียนรู้ของเครื่องได้อย่างมีประสิทธิภาพ จึงได้ดำเนินการแปลงค่าข้อความให้อยู่ในรูปแบบตัวเลขแทนจำนวนเดือน ดังแสดง

ในตาราง 10 และภาพประกอบ 8 แทนด้วย 12 เพื่อให้ข้อมูลทั้งหมดอยู่ในหน่วยเดียวกันและสามารถนำไปวิเคราะห์เชิงปริมาณต่อไปได้อย่างเหมาะสม

ตาราง 10 จำนวนเดือนของแต่ละประเภทและประเภทสัญญา

จำนวนเดือน	ประเภทสัญญา
1	1 month
3	3 months
6	6 months
12	12 months
12	1 Year

```
[ ]  contract_map = {
         '1 Month': 1,
         '3 Month': 3,
         '6 Month': 6,
         '1 Year': 12
     }

     df['Contract'] = df['Contract'].map(contract_map)

     df['Contract'].head()
```

ภาพประกอบ 8 โค้ดสำหรับแปลงประเภทสัญญาเป็นตัวเลข

3.2.2.4 Payment Method

คอลัมน์ Payment Method แสดงถึงวิธีการชำระเงินของลูกค้า ซึ่งเดิมมีรูปแบบข้อมูลเป็นข้อความ ได้แก่ "Bank transfer", "Credit card", "Bill payment" และ "QR code" เพื่อให้สามารถนำข้อมูลดังกล่าวไปใช้ในการวิเคราะห์และประมวลผลด้วยโมเดลการเรียนรู้ของเครื่อง (Machine Learning) ได้อย่างเหมาะสม จึงได้ดำเนินการแปลงข้อมูลให้อยู่ในรูปแบบตัวเลข โดยกำหนดรหัสแทนแต่ละวิธีการชำระเงินดังตาราง 11 และภาพประกอบ 9

ตาราง 11 ลำดับและวิธีการชำระเงิน

ลำดับ	วิธีการชำระเงิน
1	Bank transfer
2	Credit card
3	Bill payment
4	QR code

```
[ ]  payment_method_map = {
         'Bank transfer': 1,
         'Credit card': 2,
         'Bill payment': 3,
         'QR code': 4
     }
     df['PaymentMethod'] = df['PaymentMethod'].map(payment_method_map)
     df['PaymentMethod'].fillna(0, inplace=True)
     df['PaymentMethod'].head()
```

ภาพประกอบ 9 โค้ดสำหรับแปลงวิธีการชำระเงินเป็นตัวเลข

3.2.3 แบ่งกลุ่มข้อมูลจำพวกตัวเลข

ข้อมูลที่ใช้ในการศึกษาครั้งนี้ประกอบด้วยทั้งหมด 18 คอลัมน์ โดยในจำนวนนั้นมีข้อมูลในรูปแบบตัวเลขจำนวนทั้งสิ้น 9 คอลัมน์ ได้แก่ Tenure, Paid Invoice Count, Overdue Invoice Count, Pending Invoice Count, Reject Invoice Count, Day From Last Payment, Qty, Total Income และ Average Revenue Per Month ดังตาราง 12 ซึ่งคอลัมน์เหล่านี้ได้ถูกจัดเก็บแยกไว้ในกลุ่มตัวแปรที่เรียกว่า numerical_cols เพื่อความสะดวกในการประมวลผลและวิเคราะห์ข้อมูลเชิงปริมาณในลำดับถัดไป

ตาราง 12 ค่าเฉลี่ยของข้อมูลจำพวกตัวเลขแบ่งตามสถานการณ์ยกเลิกใช้บริการ

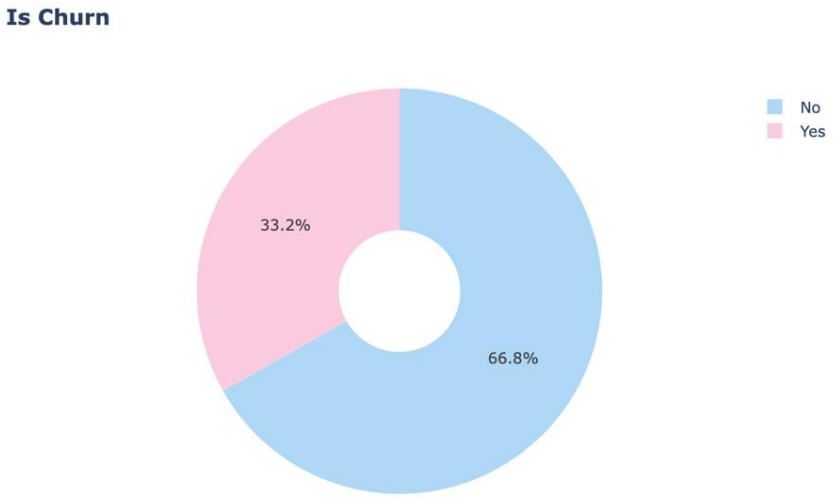
ตัวแปร	**สถานะการยกเลิก**	
	ลูกค้ายังใช้บริการ (No)	**ลูกค้ายกเลิกบริการ (Yes)**
Tenure	2	3
PaidInvoiceCount	13	5
OverdueInvoiceCount	0.26	1.31
PendingInvoiceCount	1.06	0.58
RejectInvoiceCount	0.17	0.006
DayFromLastPayment	110	781
Qty	118	35
TotalIncome	527,419	192,190
AverageRevenuePerMonth	19,947	4,964
PercentSatisfactionScore	81	78

จากการจำแนกกลุ่มของข้อมูลเชิงตัวเลข (Numerical Data) ได้มีการจัดกลุ่มย่อย โดยเปรียบเทียบระหว่างกลุ่มลูกค้าที่ยกเลิกการใช้บริการกับกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ ซึ่งในแต่ละกลุ่มได้มีการวิเคราะห์และเปรียบเทียบค่าทางสถิติต่าง ๆ เช่น ค่าเฉลี่ย ค่าสูงสุด ค่าต่ำสุด และค่ามัธยฐาน ทั้งนี้รายละเอียดของข้อมูลทางสถิติดังกล่าวแสดงไว้ใน ตาราง 12 เพื่อใช้ในการวิเคราะห์แนวโน้มของลูกค้า

3.3 การวิเคราะห์ข้อมูลเชิงสำรวจ และการแสดงภาพข้อมูล

การวิเคราะห์ข้อมูลเบื้องต้น (Exploratory Data Analysis: EDA) เป็นกระบวนการที่ใช้ในการสำรวจและทำความเข้าใจชุดข้อมูลที่ใช้ในการศึกษา โดยมีวัตถุประสงค์เพื่อวิเคราะห์ลักษณะสำคัญของข้อมูล เช่น การแจกแจงของตัวแปร ความสัมพันธ์ระหว่างตัวแปร รูปแบบที่ซ่อนอยู่ ตลอดจนการตรวจสอบข้อผิดพลาดหรือค่าผิดปกติ (Outliers) ที่อาจส่งผลกระทบต่อการวิเคราะห์ในขั้นตอนต่อไป

สำหรับการศึกษาในครั้งนี้ ตัวแปรเป้าหมายที่ใช้ในการพยากรณ์ผลลัพธ์ คือ ตัวแปร Is Churn ดังภาพประกอบ 10 และ 11 ซึ่งแสดงสถานะของลูกค้าว่ามีการยกเลิกใช้บริการหรือไม่ ดังนั้นในการวิเคราะห์ข้อมูลเบื้องต้น จึงมุ่งเน้นการสำรวจและเปรียบเทียบลักษณะของแต่ละคอลัมน์ในชุดข้อมูลกับตัวแปร Is Churn เพื่อให้เข้าใจถึงลักษณะของลูกค้าที่มีแนวโน้มยกเลิกบริการ และปัจจัยที่อาจมีความสัมพันธ์กับการยกเลิกบริการดังกล่าว ซึ่งข้อมูลที่ได้จากขั้นตอนนี้จะเป็นประโยชน์ต่อการคัดเลือกตัวแปรและออกแบบแบบจำลองในลำดับถัดไป



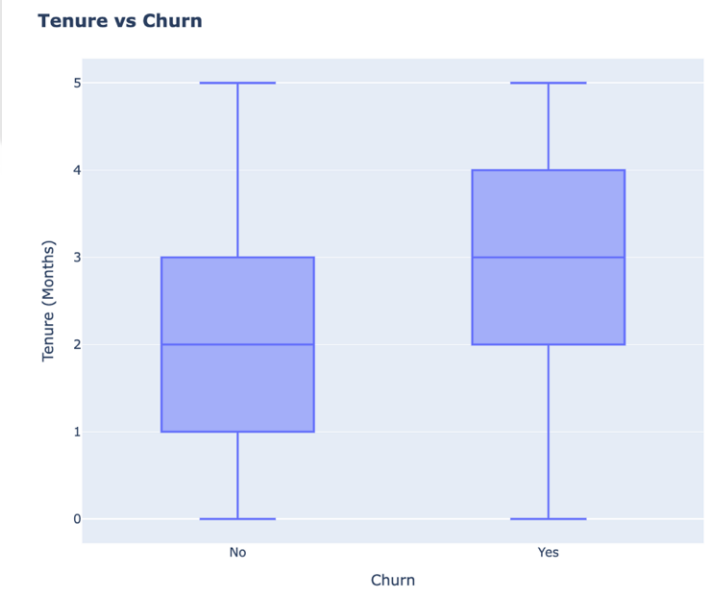
ภาพประกอบ 10 สัดส่วนสถานการณ์ยกเลิกใช้บริการของชุดข้อมูล

```
color_map = {"Yes": "#fbcce0", "No": "#afd7f6"}
fig = px.pie(df,
             names="IsChurn",
             color="IsChurn",
             title="<b>Is Churn</b>",
             hole=0.3,
             color_discrete_map=color_map)
fig.update_layout(width=700, height=500, bargap=0.1)
fig.show()
```

ภาพประกอบ 11 โค้ดสำหรับสัดส่วนสถานการณ์ยกเลิกใช้บริการของชุดข้อมูล

3.3.1 ระยะเวลาการใช้บริการ (ปี) กับสถานะการยกเลิกใช้บริการ

อ้างอิงจากภาพประกอบ 12 และ 13 ลูกค้าที่ยกเลิกการใช้บริการ (Churn = Yes) มีแนวโน้มที่จะใช้บริการในระยะเวลาที่สั้นกว่าลูกค้าที่ยังคงใช้บริการอยู่ (Churn = No) โดยค่ามัธยฐานของกลุ่มที่ยกเลิกใช้บริการอยู่ที่ประมาณ 3 เดือน และมีช่วงระหว่างควอไทล์ที่กว้างกว่า คือตั้งแต่ประมาณ 2 ถึง 4 เดือน ในทางกลับกันกลุ่มลูกค้าที่ยังคงใช้บริการอยู่มีค่ามัธยฐานอยู่ที่ประมาณ 2 เดือน แต่ช่วงระหว่างควอไทล์แคบกว่า คืออยู่ในช่วงประมาณ 1 ถึง 3 เดือน แสดงถึงการมีพฤติกรรมการใช้งานที่คงที่และกระจุกตัวมากกว่า



ภาพประกอบ 12 เปรียบเทียบระยะเวลาการใช้บริการ (ปี) กับสถานะการยกเลิกใช้บริการ

```
fig = px.box(df, x='IsChurn', y = 'Tenure')

fig.update_yaxes(title_text='Tenure (Months)', row=1, col=1)
fig.update_xaxes(title_text='Churn', row=1, col=1)

fig.update_layout(autosize=True, width=750, height=600,
    title_font=dict(size=25, family='Courier'),
    title='<b>Tenure vs Churn</b>',)

fig.show()
```

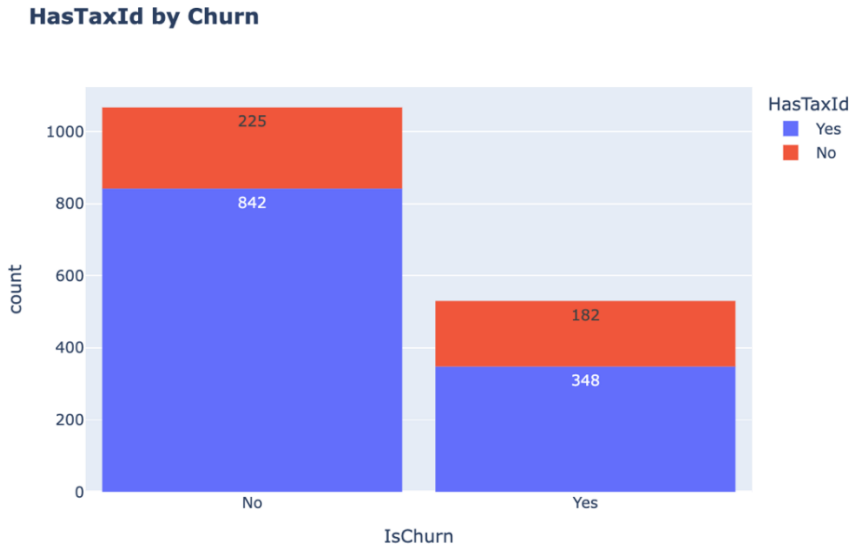
ภาพประกอบ 13 โค้ดสำหรับเปรียบเทียบระยะเวลาการใช้บริการ (ปี) กับสถานการณ์ยกเลิกใช้บริการ

3.3.2 ลูกค้ามีการกรอกข้อมูลเลขผู้เสียภาษีกับสถานะการยกเลิกใช้บริการ

จากภาพประกอบ 14 และ 15 การวิเคราะห์ข้อมูลด้วยกราฟแบบ Stacked Bar Chart แสดงให้เห็นถึงความสัมพันธ์ระหว่างการมีหรือไม่มีข้อมูลเลขผู้เสียภาษี (HasTaxId) กับสถานะการยกเลิกใช้บริการ (IsChurn) โดยข้อมูลแบ่งเป็นสองกลุ่มคือ ลูกค้าที่ยังคงใช้บริการอยู่ (IsChurn = No) และลูกค้าที่ยกเลิกการใช้บริการ (IsChurn = Yes)

ในกลุ่มลูกค้าที่ยังคงใช้บริการอยู่มีจำนวนทั้งหมด 1,067 คน แบ่งเป็นลูกค้าที่ให้ข้อมูลเลขผู้เสียภาษีจำนวน 842 คน และไม่ได้ให้ข้อมูลจำนวน 225 คน คิดเป็นสัดส่วนของผู้ที่ไม่มีข้อมูลเลขผู้เสียภาษีประมาณ 21.1% ของกลุ่มนี้ ขณะที่ในกลุ่มลูกค้าที่ ยกเลิกการใช้บริการ ซึ่งมีจำนวนทั้งหมด 530 คน มีผู้ให้ข้อมูลเลขผู้เสียภาษี 348 คน และไม่ได้ให้ข้อมูล 182 คน คิดเป็นสัดส่วนประมาณ 34.3% ของกลุ่มนี้

เมื่อเปรียบเทียบทั้งสองกลุ่ม จะพบว่าลูกค้าที่ไม่มีข้อมูลเลขผู้เสียภาษีมีสัดส่วนสูงกว่าในกลุ่มที่ยกเลิกบริการอย่างชัดเจน ซึ่งอาจสะท้อนถึงแนวโน้มว่าลูกค้าที่ไม่ให้ข้อมูลเลขผู้เสียภาษีมีโอกาสเลิกใช้บริการมากกว่า โดยอาจมีความเป็นไปได้ว่า ลูกค้าในกลุ่มนี้เป็นลูกค้ารายย่อยใช้บริการชั่วคราว หรือไม่ได้ผูกพันเชิงธุรกิจมากนัก จึงมีแนวโน้มที่จะยกเลิกบริการได้ง่ายกว่ากลุ่มที่ให้ข้อมูลครบถ้วน



ภาพประกอบ 14 เปรียบเทียบลูกค้ามีการกรอกข้อมูลเลขผู้เสียภาษีกับสถานการณ์ยกเลิกใช้บริการ

```
fig = px.histogram(df,
                   x="IsChurn",
                   color="HasTaxId",
                   title="<b>HasTaxId by Churn</b>",
                   text_auto="both")
fig.update_layout(width=700, height=500, bargap=0.1)
fig.show()
```

ภาพประกอบ 15 โค้ดสำหรับเปรียบเทียบลูกค้ามีการกรอกข้อมูลเลขผู้เสียภาษีกับสถานการณ์ยกเลิกใช้บริการ

3.3.3 ลูกค้ามีการกรอกเบอร์ติดต่อกับสถานะการยกเลิกใช้บริการ

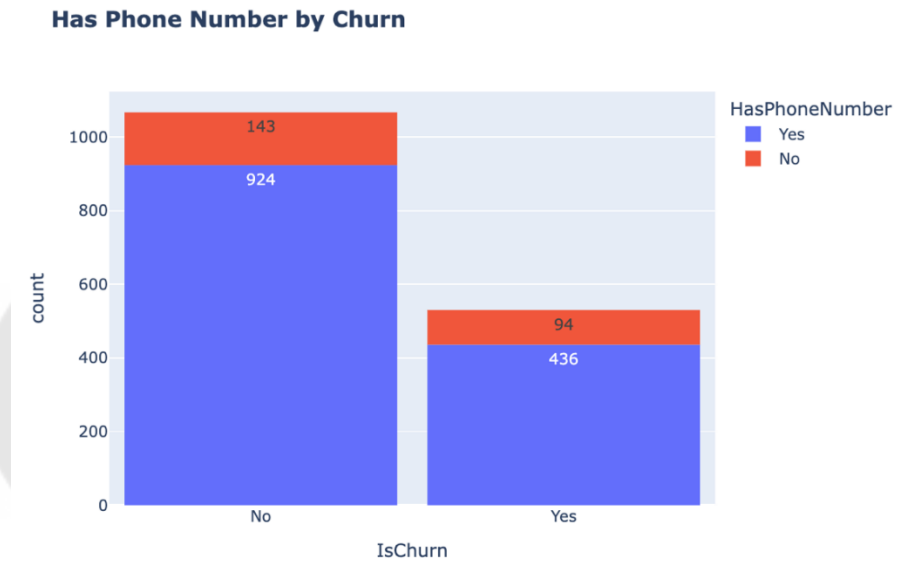
จากภาพประกอบ 16 และ 17 การวิเคราะห์ข้อมูลด้วยภาพ Stacked Bar Chart สามารถสรุปได้ว่าลูกค้าส่วนใหญ่ทั้งที่ยังคงใช้บริการและที่ยกเลิกบริการ มีการระบุหมายเลขโทรศัพท์ไว้ โดยในกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ (IsChurn = No) ซึ่งมีจำนวนรวม 1,067 คน พบว่ามีลูกค้า 924 คนที่ให้ข้อมูลหมายเลขโทรศัพท์ และ 143 คนที่ไม่มีข้อมูลคิดเป็นสัดส่วนของผู้ที่ไม่มีหมายเลขโทรศัพท์อยู่ที่ประมาณ 13.4%

ในขณะที่กลุ่มลูกค้าที่ยกเลิกการใช้บริการ (IsChurn = Yes) ซึ่งมีจำนวนรวม 530 คน มีลูกค้า 436 คนที่มีข้อมูลหมายเลขโทรศัพท์ และ 94 คนที่ไม่มี คิดเป็นสัดส่วนของลูกค้าที่ไม่มีหมายเลขโทรศัพท์อยู่ที่ประมาณ 17.7% ซึ่งแม้ว่าสัดส่วนนี้จะสูงกว่ากลุ่มที่ยังคงใช้บริการอยู่

เล็กน้อย แต่ความแตกต่างไม่ถือว่ามีนัยสำคัญมากเมื่อเปรียบเทียบกับข้อมูลอื่น เช่น ข้อมูลเลขผู้เสียภาษี

ช่วงจำนวนลูกค้าที่ไม่มีหมายเลขโทรศัพท์ อยู่ที่ 94–143 คน ในขณะที่ลูกค้าที่มีหมายเลขโทรศัพท์ อยู่ในช่วง 436–924 คน โดยกลุ่มที่ยังใช้บริการมีจำนวนมากกว่าในทั้งสองกรณี

จากข้อมูลนี้สามารถตีความได้ว่า การมีหรือไม่มีหมายเลขโทรศัพท์อาจไม่ได้เป็นปัจจัยที่ส่งผลอย่างชัดเจนต่อการตัดสินใจยกเลิกใช้บริการของลูกค้า เพราะแม้สัดส่วนของผู้ไม่มีหมายเลขโทรศัพท์จะสูงกว่าเล็กน้อยในกลุ่มที่เลิกใช้บริการ แต่ยังถือว่าอยู่ในระดับใกล้เคียงกัน



ภาพประกอบ 16 เปรียบเทียบการกรอกเบอร์ติดต่อกับสถานการณ์ยกเลิกใช้บริการ

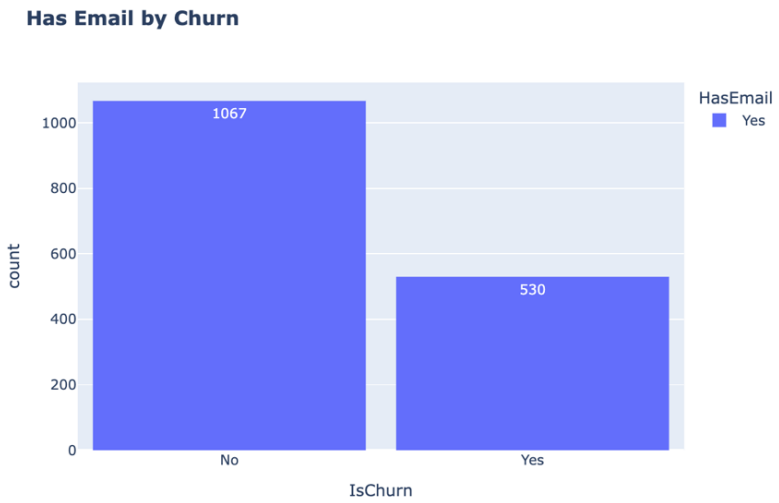
```
[ ]  fig = px.histogram(df,
                        x="IsChurn",
                        color="HasPhoneNumber",
                        title="<b>Has Phone Number by Churn</b>",
                        text_auto="both")
     fig.update_layout(width=700, height=500, bargap=0.1)
     fig.show()
```

ภาพประกอบ 17 โค้ดสำหรับเปรียบเทียบการกรอกเบอร์ติดต่อกับสถานการณ์ยกเลิกใช้บริการ

3.3.4 ลูกค้ามีการกรอกอีเมลติดต่อกับสถานะการยกเลิกใช้บริการ

จากภาพประกอบ 18 และ 19 วิเคราห์ข้อมูลที่แสดงใน bar chart สามารถสรุปได้ว่าลูกค้าทั้งหมดที่ปรากฏในกราฟเป็นลูกค้าที่มีข้อมูลอีเมล โดยในกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ (IsChurn = No) มีจำนวนรวม 1,067 คน และในกลุ่มลูกค้าที่ยกเลิกการใช้บริการ (IsChurn =

Yes) มีจำนวนรวม 530 คน ทั้งสองกลุ่มมีจำนวนลูกค้าที่ให้ข้อมูลอีเมลครบถ้วน ซึ่งหมายความว่าข้อมูลในกราฟไม่สามารถสะท้อนถึงลูกค้าที่ไม่มีอีเมลได้



ภาพประกอบ 18 เปรียบเทียบการกรอกอีเมลติดต่อกับสถานะการยกเลิกใช้บริการ

```
fig = px.histogram(df,
                   x="IsChurn",
                   color="HasEmail",
                   title="<b>Has Email by Churn</b>",
                   text_auto="both")
fig.update_layout(width=700, height=500, bargap=0.1)
fig.show()
```

ภาพประกอบ 19 โค้ดสำหรับเปรียบเทียบการกรอกอีเมลติดต่อกับสถานการณ์ยกเลิกใช้บริการ

3.3.5 ประเภทสัญญากับสถานะการยกเลิกใช้บริการ

จากภาพประกอบ 20 และ 21 แสดงการกระจายตัวของลูกค้าตามประเภทสัญญาแยกตามสถานะการยกเลิกบริการ พบว่าในกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ มีจำนวนตั้งแต่ 1 คนถึง 783 คน โดยลูกค้าที่ทำสัญญา 1 ปีมีจำนวนมากที่สุดคือ 783 คน รองลงมาคือสัญญา 1 เดือนที่มี 185 คน และสัญญา 3 เดือนที่มี 98 คน ขณะที่สัญญา 6 เดือนพบเพียง 1 คน ซึ่งถือว่ามีน้อยมาก

ในกลุ่มลูกค้าที่ตัดสินใจยกเลิกบริการ มีจำนวนตั้งแต่ 0 คนถึง 300 คน โดยลูกค้าสัญญา 1 เดือนมีจำนวนมากที่สุดคือ 300 คน ขณะที่ลูกค้าสัญญา 1 ปีอยู่ที่ 181 คน สัญญา 3 เดือนอยู่ที่ 49 คน และไม่มีลูกค้าสัญญา 6 เดือนเลยในกลุ่มนี้

จากข้อมูลดังกล่าวสามารถสังเกตได้ว่าลูกค้าที่ทำสัญญา 1 เดือนมีแนวโน้มที่จะยกเลิกบริการสูงกว่ากลุ่มอื่นอย่างชัดเจน โดยจำนวนในกลุ่มที่ยกเลิก (300 คน) มากกว่ากลุ่มที่ยัง

ใช้บริการอยู่ (185 คน) ในทางกลับกัน ลูกค้าที่ทำสัญญา 1 ปีมีแนวโน้มที่จะใช้บริการต่อมากที่สุด เนื่องจากจำนวนในกลุ่มที่ยังใช้บริการอยู่ (783 คน) สูงกว่ากลุ่มที่ยกเลิก (181 คน) อย่างมีนัยสำคัญ สำหรับลูกค้าสัญญา 3 เดือน แม้ว่าจะมีจำนวนไม่มากนัก แต่มีแนวโน้มที่จะยังใช้บริการอยู่มากกว่ายกเลิก ขณะที่สัญญา 6 เดือนพบว่ามีเพียง 1 คนที่ยังใช้บริการ และไม่มีลูกค้าในกลุ่มที่ยกเลิก

ภาพประกอบ 20 เปรียบเทียบประเภทสัญญาและสถานการณ์ยกเลิกการใช้บริการ

```
import plotly.express as px

fig = px.histogram(df, x="IsChurn", color="Contract", barmode="group",
                   title="<b>Customer contract distribution<b>", text_auto=True)

fig.update_layout(width=700, height=500, bargap=0.1)
fig.show()
```

ภาพประกอบ 21 โค้ดสำหรับเปรียบเทียบประเภทสัญญาและสถานการณ์ยกเลิกการใช้บริการ

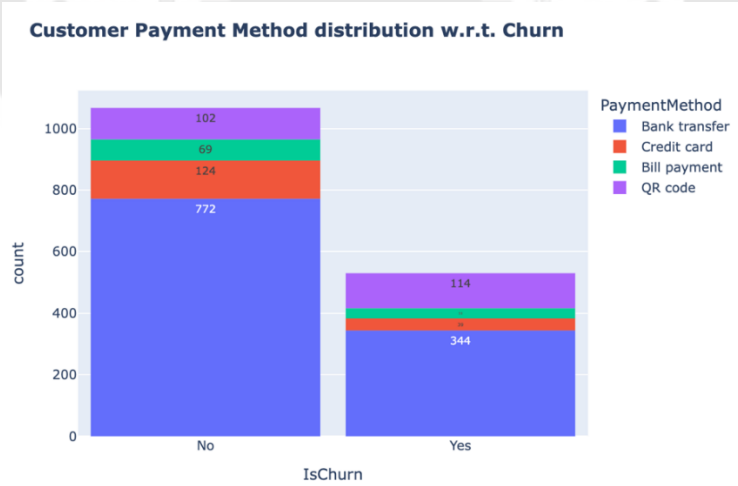
3.3.6 ประเภทการชำระเงินกับสถานะการยกเลิกใช้บริการ

จากภาพประกอบ 22 และ 23 แสดง stacked bar chart ที่แสดงการกระจายตัวของลูกค้าตามประเภทการชำระเงิน (Payment Method) แยกตามการยกเลิกใช้บริการ (IsChurn) พบว่าลูกค้าที่ชำระเงินด้วย Bank transfer เป็นกลุ่มที่มีจำนวนสูงที่สุดในทั้งสองกลุ่มการยกเลิกบริการ ในกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ (IsChurn = No) จำนวนลูกค้าที่ชำระเงินด้วย Bank transfer อยู่ที่ 772 คน ซึ่งคิดเป็น 72.4% ของกลุ่มนี้ ขณะที่ในกลุ่มลูกค้าที่ยกเลิกการใช้บริการ

(IsChurn = Yes) จำนวนลูกค้าที่ชำระเงินด้วย Bank transfer ลดลงเหลือ 344 คน หรือประมาณ 64.9% ของกลุ่มนี้

เมื่อพิจารณาประเภทการชำระเงินอื่น ๆ พบว่าลูกค้าที่ใช้ Credit card มีจำนวนค่อนข้างใกล้เคียงกันระหว่างทั้งสองกลุ่ม โดยในกลุ่มที่ยังคงใช้บริการอยู่ (IsChurn = No) มีจำนวน 124 คน ในขณะที่ในกลุ่มที่ยกเลิกการใช้บริการ (IsChurn = Yes) มีจำนวน 114 คน ลูกค้าที่ใช้ Bill payment ในกลุ่มที่ยังคงใช้บริการอยู่มีจำนวน 69 คน ขณะที่ในกลุ่มที่ยกเลิกการใช้บริการมีจำนวนเพียง 10 คน ลูกค้าที่ใช้ QR code ในกลุ่มที่ยังคงใช้บริการอยู่มีจำนวน 102 คน และในกลุ่มที่ยกเลิกการใช้บริการมีจำนวน 62 คน ซึ่งแสดงให้เห็นว่าลูกค้าที่ใช้ QR code มีแนวโน้มที่จะยกเลิกบริการสูงขึ้นเล็กน้อย

ในแง่ของการกระจายตัวและความสัมพันธ์ระหว่างประเภทการชำระเงินกับการยกเลิกบริการ สามารถสรุปได้ว่าลูกค้าที่เลือกใช้ Bank transfer มีแนวโน้มที่จะคงใช้บริการมากที่สุด ขณะที่ลูกค้าที่เลือกใช้ QR code มีแนวโน้มที่จะยกเลิกบริการมากที่สุด ลูกค้าที่ชำระเงินด้วย Bill payment มีจำนวนที่น้อยมากในกลุ่มที่ยกเลิกบริการ, ซึ่งอาจบ่งชี้ว่าเป็นกลุ่มที่มีความพึงพอใจสูงในการใช้บริการ



ภาพประกอบ 22 เปรียบเทียบประเภทการชำระเงินและสถานการณ์ยกเลิกการใช้บริการ

```
[ ]  fig = px.histogram(df, x="IsChurn"
                        , color="PaymentMethod"
                        , title="<b>Customer Payment Method distribution w.r.t. Churn</b>"
                        , text_auto=True)
     fig.update_layout(width=700, height=500, bargap=0.1)
     fig.update_traces(textfont_size=10)
     fig.show()
```

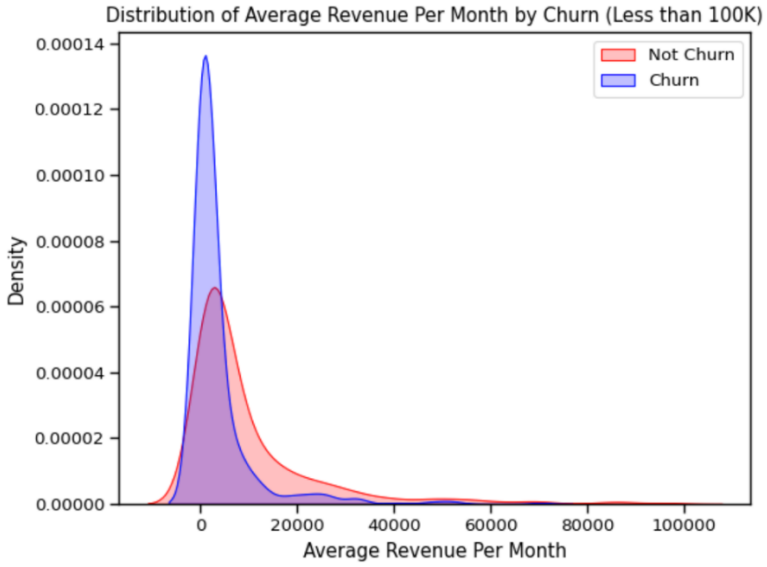
ภาพประกอบ 23 โค้ดสำหรับเปรียบเทียบประเภทการชำระเงินและสถานการณ์ยกเลิกการใช้บริการ

3.3.7 ค่าเฉลี่ยยอดค่าบริการรายเดือนกับสถานะการยกเลิกใช้บริการ

จากภาพประกอบ 24 และ 25 วิเคราะห์ข้อมูลการกระจายตัวของค่าเฉลี่ยยอดค่าบริการรายเดือน (Average Revenue Per Month) ของลูกค้าที่ยกเลิกบริการ (Churn) และลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) โดยจำกัดข้อมูลให้อยู่ในช่วงที่มีค่าเฉลี่ยยอดค่าบริการไม่เกิน 100,000 หน่วย พบว่าสำหรับลูกค้าที่ยกเลิกการใช้บริการ (Churn) จุดสูงสุดของความหนาแน่นจะอยู่ในช่วงค่าประมาณ 0 ถึง 10,000 หน่วย โดยมีลักษณะการกระจายตัวที่แคบและจุดสูงสุดชัดเจน ซึ่งแสดงให้เห็นว่าลูกค้าส่วนใหญ่ที่ยกเลิกบริการมีค่าเฉลี่ยยอดค่าบริการรายเดือนที่ต่ำ การกระจายตัวของข้อมูลนี้มีหางที่ทอดตัวไปทางขวาเล็กน้อย บ่งชี้ถึงการมีลูกค้าบางกลุ่มที่มีค่าเฉลี่ยยอดค่าบริการที่สูงกว่า แต่มีจำนวนที่น้อยกว่าลูกค้าที่มียอดใช้จ่ายต่ำ ในขณะที่ลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่า จุดสูงสุดของความหนาแน่นอยู่ในช่วงประมาณ 5,000 ถึง 20,000 หน่วย โดยมีลักษณะของการกระจายตัวที่กว้างกว่ากลุ่มที่ยกเลิกบริการ และมีลักษณะเป็นสองยอดเล็กน้อย โดยยอดแรกจะใกล้เคียงกับจุดสูงสุดของกลุ่มที่ยกเลิกบริการ ส่วนยอดที่สองจะอยู่ในช่วงที่สูงกว่า ซึ่งบ่งชี้ว่าในกลุ่มนี้มีลูกค้าบางกลุ่มที่มีค่าเฉลี่ยยอดค่าบริการรายเดือนที่สูงกว่าอย่างเห็นได้ชัด และหางของกราฟสีแดงทอดตัวไปทางขวาแสดงให้เห็นถึงการมีลูกค้าจำนวนมากที่มียอดใช้จ่ายสูงและยังคงใช้บริการอยู่

จากการเปรียบเทียบข้อมูลระหว่างสองกลุ่มนี้พบว่า ลูกค้าที่ยกเลิกบริการมักจะมีค่าเฉลี่ยยอดค่าบริการที่ต่ำกว่า โดยจุดสูงสุดของการแจกแจงความหนาแน่นของกลุ่มนี้อยู่ในช่วง 0 ถึง 10,000 หน่วย ขณะที่กลุ่มลูกค้าที่ยังคงใช้บริการจะมีจุดสูงสุดในช่วงที่สูงขึ้นประมาณ 5,000 ถึง 20,000 หน่วย นอกจากนี้ยังมีการกระจายตัวของลูกค้าที่มีค่าใช้จ่ายต่ำแต่ยังคงใช้บริการอยู่ ซึ่งแสดงถึงการมีความทับซ้อนในช่วงค่าที่ต่ำกว่า และการมีลูกค้าที่มียอดใช้จ่ายสูงในกลุ่มที่ไม่ยกเลิกบริการ

ดังนั้นข้อมูลนี้แสดงให้เห็นถึงความแตกต่างระหว่างลูกค้าที่ยกเลิกบริการและไม่ยกเลิกบริการในเรื่องของค่าเฉลี่ยยอดค่าบริการรายเดือน โดยลูกค้าที่มียอดใช้จ่ายต่ำมีแนวโน้มที่จะยกเลิกบริการมากกว่า ในขณะที่ลูกค้าที่มียอดใช้จ่ายสูงมักจะยังคงใช้บริการอยู่



ภาพประกอบ 24 เปรียบเทียบประเภทสัญญาและสถานการณ์ยกเลิกการใช้บริการ

```
[ ]  import seaborn as sns
     import matplotlib.pyplot as plt

     filtered_df = df[df["AverageRevenuePerMonth"] < 100000]

     sns.set_context("paper", font_scale=1.1)

     ax = sns.kdeplot(filtered_df.AverageRevenuePerMonth[(filtered_df["IsChurn"] == 'No')],
                      color="Red", shade=True)

     ax = sns.kdeplot(filtered_df.AverageRevenuePerMonth[(filtered_df["IsChurn"] == 'Yes')],
                      ax=ax, color="Blue", shade=True)

     ax.legend(["Not Churn", "Churn"], loc='upper right')
     ax.set_ylabel('Density')
     ax.set_xlabel('Average Revenue Per Month')
     ax.set_title('Distribution of Average Revenue Per Month by Churn (Less than 100K)')

     plt.show()
```

ภาพประกอบ 25 โค้ดสำหรับเปรียบเทียบประเภทสัญญาและสถานการณ์ยกเลิกการใช้บริการ

3.3.8 **จำนวนใบแจ้งหนี้ที่ชำระแล้วกับสถานะการยกเลิกใช้บริการ**

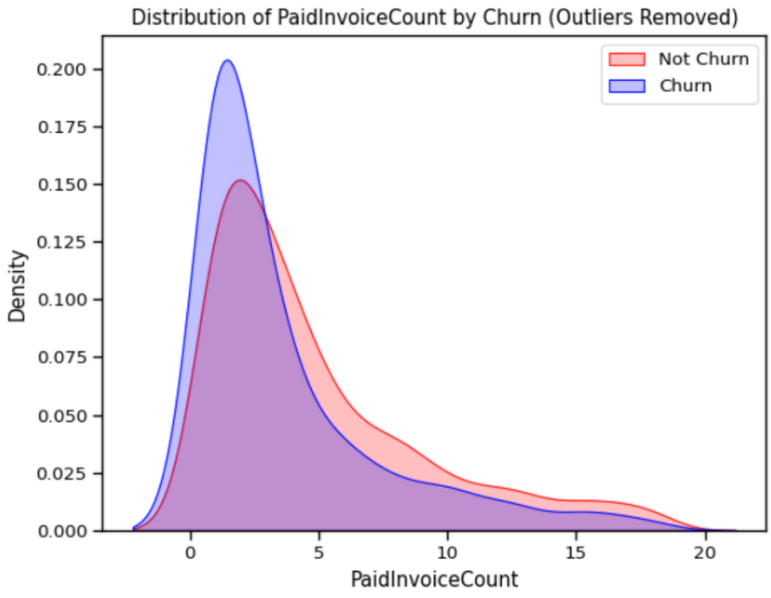
จากภาพประกอบ 26 และ 27 วิเคราะห์ข้อมูลการกระจายตัวของจำนวนใบแจ้งหนี้ที่ชำระแล้ว (Paid Invoice Count) สำหรับลูกค้าที่ยกเลิกบริการ (Churn) และลูกค้าที่ยังคงใช้บริการ

อยู่ (Not Churn) โดยมีการนำค่าผิดปกติ (Outliers) ออกไปแล้ว พบว่าความแตกต่างในการกระจายตัวของข้อมูลในแต่ละกลุ่มมีลักษณะที่ชัดเจน

สำหรับลูกค้าที่ยกเลิกการใช้บริการ (Churn) จุดสูงสุดของความหนาแน่นของข้อมูลจะอยู่ในช่วงประมาณ 1-2 ใบ โดยลูกค้าส่วนใหญ่ที่ยกเลิกบริการมีจำนวนใบแจ้งหนี้ที่ชำระแล้วในช่วงเริ่มต้นของการใช้บริการ หางของกราฟจะทอดตัวไปทางขวา แต่อย่างไรก็ตาม ความหนาแน่นจะลดลงอย่างรวดเร็ว แสดงว่ามีลูกค้าบางกลุ่มที่มีจำนวนใบแจ้งหนี้ที่ชำระแล้วมากขึ้นแต่มีจำนวนน้อย

ในขณะที่ลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่าจุดสูงสุดของความหนาแน่นจะอยู่ในช่วงประมาณ 2-3 ใบ โดยการกระจายตัวของข้อมูลในกลุ่มนี้กว้างกว่ากลุ่มที่ยกเลิกบริการและมีความหนาแน่นที่สูงขึ้นในช่วงจำนวนใบแจ้งหนี้ที่ชำระแล้วมากกว่า (ประมาณ 3-10 ใบ) หางของกราฟสีแดงทอดตัวไปทางขวาอย่างชัดเจนและยาวกว่ากลุ่มที่ยกเลิกบริการ ซึ่งบ่งชี้ว่ามีลูกค้าจำนวนมากที่ยังคงใช้บริการและมีจำนวนใบแจ้งหนี้ที่ชำระแล้วมากกว่า

ดังนั้นลูกค้าที่ยกเลิกบริการส่วนใหญ่จะมีจำนวนใบแจ้งหนี้ที่ชำระแล้วน้อย โดยจุดสูงสุดของความหนาแน่นอยู่ในช่วง 1-2 ใบ ขณะที่ลูกค้าที่ยังคงใช้บริการอยู่จะมีจำนวนใบแจ้งหนี้ที่ชำระแล้วมากกว่า โดยจุดสูงสุดของความหนาแน่นอยู่ในช่วง 2-3 ใบ และการกระจายตัวของข้อมูลในกลุ่มนี้จะกว้างกว่ากลุ่มที่ยกเลิกบริการ ซึ่งแสดงถึงพฤติกรรมการชำระเงินที่ต่อเนื่องของลูกค้าที่ยังคงใช้บริการอยู่



ภาพประกอบ 26 เปรียบเทียบจำนวนใบแจ้งหนี้ที่ชำระแล้วและสถานการณ์ยกเลิกการใช้บริการ

```
[29] import seaborn as sns
     import numpy as np
     import matplotlib.pyplot as plt

     Q1 = df["PaidInvoiceCount"].quantile(0.25)
     Q3 = df["PaidInvoiceCount"].quantile(0.75)
     IQR = Q3 - Q1

     lower_bound = Q1 - 1.5 * IQR
     upper_bound = Q3 + 1.5 * IQR

     filtered_df = df[(df["PaidInvoiceCount"] >= lower_bound) & (df["PaidInvoiceCount"] <= upper_bound)]

     sns.set_context("paper", font_scale=1.1)
     ax = sns.kdeplot(filtered_df.PaidInvoiceCount[(filtered_df["IsChurn"] == 'No')],
                      color="Red", shade=True)
     ax = sns.kdeplot(filtered_df.PaidInvoiceCount[(filtered_df["IsChurn"] == 'Yes')],
                      ax=ax, color="Blue", shade=True)

     ax.legend(["Not Churn", "Churn"], loc='upper right')
     ax.set_ylabel('Density')
     ax.set_xlabel('PaidInvoiceCount')
     ax.set_title('Distribution of PaidInvoiceCount by Churn (Outliers Removed)')

     plt.show()
```

ภาพประกอบ 27 โค้ดสำหรับเปรียบเทียบจำนวนใบแจ้งหนี้ที่ชำระแล้วและสถานการณ์ยกเลิกการใช้บริการ

3.3.9 **จำนวนใบแจ้งหนี้ที่เลยกำหนดการชำระกับสถานะการยกเลิกใช้บริการ**

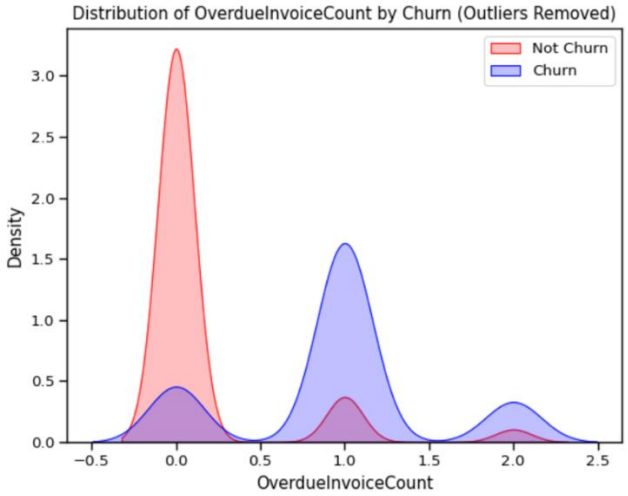
จากภาพประกอบ 28 และ 29 วิเคราะห์การกระจายตัวของจำนวนใบแจ้งหนี้ที่ค้างชำระ (Overdue Invoice Count) สำหรับลูกค้าที่ยกเลิกบริการ (Churn) และลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่าในทั้งสองกลุ่มมีลักษณะการกระจายตัวที่แตกต่างกันอย่างชัดเจน

สำหรับกลุ่มลูกค้าที่ยกเลิกบริการ (Churn) จุดสูงสุดของความหนาแน่นจะอยู่ที่ประมาณ 1 ใบ และอีกจุดหนึ่งจะอยู่ที่จำนวนใบแจ้งหนี้ค้างชำระที่ใกล้เคียงกับ 0 ใบ ซึ่งบ่งชี้ว่ามีลูกค้าที่ยกเลิกบริการจำนวนมากที่ไม่มียอดค้างชำระเลย หรือมีเพียงเล็กน้อย โดยความหนาแน่นจะลดลงอย่างรวดเร็วเมื่อจำนวนใบแจ้งหนี้ค้างชำระเพิ่มขึ้น การกระจายตัวในกลุ่มนี้เป็นแบบสองยอด (Bimodal) โดยลูกค้าส่วนใหญ่มีจำนวนใบแจ้งหนี้ค้างชำระที่ 0-1 ใบ

ในขณะที่กลุ่มลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่าจุดสูงสุดของความหนาแน่นจะอยู่ที่จำนวนใบแจ้งหนี้ค้างชำระ 0 ใบ ซึ่งบ่งชี้ว่าลูกค้าส่วนใหญ่ที่ยังคงใช้บริการไม่มีใบแจ้งหนี้ค้างชำระเลย และเมื่อจำนวนใบแจ้งหนี้ค้างชำระเพิ่มขึ้น ความหนาแน่นจะลดลงอย่างรวดเร็ว โดยเฉพาะเมื่อมียอดค้างชำระ 1 ใบขึ้นไป ความหนาแน่นในกลุ่มนี้จะต่ำกว่ากลุ่มที่ยกเลิกบริการอย่างชัดเจน

ดังนั้นลูกค้าที่ยังไม่มียอดค้างชำระมีความเสี่ยงต่ำที่จะยกเลิกบริการ ขณะที่ลูกค้าที่มียอดค้างชำระ 1 ใบมีแนวโน้มที่จะยกเลิกบริการมากกว่าลูกค้าที่ไม่มียอดค้างชำระเลย โดยเฉพาะ

ในกลุ่มลูกค้าที่ยกเลิกบริการ ซึ่งมีการกระจายตัวในช่วง 0-1 ใบ ขณะที่ลูกค้าที่ยังคงใช้บริการอยู่จะมีแนวโน้มที่จะไม่มียอดค้างชำระเลย



ภาพประกอบ 28 เปรียบเทียบจำนวนใบแจ้งหนี้ที่เลยกำหนดการชำระและสถานการณ์ยกเลิกการใช้บริการ

```python
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

Q1 = df["OverdueInvoiceCount"].quantile(0.25)
Q3 = df["OverdueInvoiceCount"].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

filtered_df = df[(df["OverdueInvoiceCount"] >= lower_bound) & (df["OverdueInvoiceCount"] <= upper_bound)]

sns.set_context("paper", font_scale=1.1)
ax = sns.kdeplot(filtered_df.OverdueInvoiceCount[(filtered_df["IsChurn"] == 'No')],
                 color="Red", shade=True)
ax = sns.kdeplot(filtered_df.OverdueInvoiceCount[(filtered_df["IsChurn"] == 'Yes')],
                 ax=ax, color="Blue", shade=True)

ax.legend(["Not Churn", "Churn"], loc='upper right')
ax.set_ylabel('Density')
ax.set_xlabel('OverdueInvoiceCount')
ax.set_title('Distribution of OverdueInvoiceCount by Churn (Outliers Removed)')

plt.show()
```

ภาพประกอบ 29 โค้ดสำหรับเปรียบเทียบจำนวนใบแจ้งหนี้ที่เลยกำหนดการชำระและสถานการณ์ยกเลิกการใช้บริการ

3.3.10 จำนวนใบแจ้งหนี้ที่รอการชำระกับสถานะการยกเลิกใช้บริการ

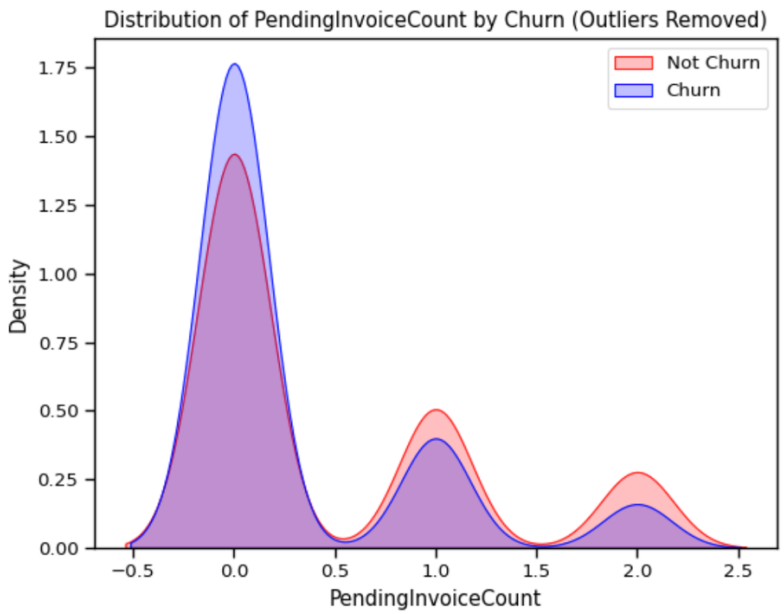
จากภาพประกอบ 30 และ 31 วิเคราะห์การกระจายตัวของจำนวนใบแจ้งหนี้ที่รอการชำระ (Pending Invoice Count) สำหรับลูกค้าที่ยกเลิกบริการ (Churn) และลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่ามีลักษณะการกระจายตัวที่คล้ายคลึงกันในทั้งสองกลุ่ม โดยมีจุดสูงสุดของ

ความหนาแน่นในทั้งสองกลุ่มที่จำนวนใบแจ้งหนี้ที่รอการชำระเป็น 0 ใบ ซึ่งบ่งชี้ว่าลูกค้าส่วนใหญ่ทั้งที่ยกเลิกและยังคงใช้บริการอยู่ไม่มีใบแจ้งหนี้ที่รอการชำระอยู่เลย

ในกลุ่มลูกค้าที่ยกเลิกบริการ (Churn) พบว่ามีจุดสูงสุดที่ประมาณ 1 ใบ ซึ่งหมายความว่ามีลูกค้าจำนวนหนึ่งที่มีใบแจ้งหนี้รอการชำระเพียง 1 ใบ โดยมีความหนาแน่นที่สูงกว่ากลุ่มลูกค้าที่ยังคงใช้บริการในช่วงดังกล่าว ความหนาแน่นของกราฟจะลดลงอย่างรวดเร็วเมื่อจำนวนใบแจ้งหนี้ที่รอการชำระเพิ่มขึ้น

สำหรับกลุ่มลูกค้าที่ยังคงใช้บริการอยู่ (Not Churn) พบว่ามีจุดสูงสุดที่ประมาณ 0 ใบ เช่นเดียวกัน โดยความหนาแน่นที่จุดนี้สูงกว่ากลุ่มลูกค้าที่ยกเลิกบริการอย่างชัดเจน ในช่วงที่มีใบแจ้งหนี้รอการชำระ 1 ใบ ความหนาแน่นจะลดลงอย่างรวดเร็ว และยังพบว่ามีลูกค้าบางส่วนที่มีใบแจ้งหนี้รอการชำระมากกว่า 1 ใบ ซึ่งกลุ่มนี้มีความหนาแน่นสูงกว่าเล็กน้อยเมื่อเทียบกับกลุ่มที่ยกเลิกบริการ

ดังนั้นลูกค้าที่ยังคงใช้บริการส่วนใหญ่ไม่มีใบแจ้งหนี้ที่รอการชำระ ในขณะที่ลูกค้าที่ยกเลิกบริการมีสัดส่วนสูงขึ้นในช่วงที่มีใบแจ้งหนี้รอการชำระประมาณ 1 ใบ แต่ทั้งสองกลุ่มยังคงมีการกระจายตัวที่ค่อนข้างทับซ้อนกันในช่วง 0-1 ใบ จึงไม่สามารถใช้จำนวนใบแจ้งหนี้รอการชำระเป็นตัวบ่งชี้การยกเลิกบริการที่ชัดเจน



ภาพประกอบ 30 เปรียบเทียบจำนวนใบแจ้งหนี้ที่รอการชำระและสถานการณ์ยกเลิกการใช้บริการ

```
[31] import seaborn as sns
     import numpy as np
     import matplotlib.pyplot as plt

     Q1 = df["PendingInvoiceCount"].quantile(0.25)
     Q3 = df["PendingInvoiceCount"].quantile(0.75)
     IQR = Q3 - Q1

     lower_bound = Q1 - 1.5 * IQR
     upper_bound = Q3 + 1.5 * IQR

     filtered_df = df[(df["PendingInvoiceCount"] >= lower_bound) & (df["PendingInvoiceCount"] <= upper_bound)]

     sns.set_context("paper", font_scale=1.1)
     ax = sns.kdeplot(filtered_df.PendingInvoiceCount[(filtered_df["IsChurn"] == 'No')],
                      color="Red", shade=True)
     ax = sns.kdeplot(filtered_df.PendingInvoiceCount[(filtered_df["IsChurn"] == 'Yes')],
                      ax=ax, color="Blue", shade=True)

     ax.legend(["Not Churn", "Churn"], loc='upper right')
     ax.set_ylabel('Density')
     ax.set_xlabel('PendingInvoiceCount')
     ax.set_title('Distribution of PendingInvoiceCount by Churn (Outliers Removed)')

     plt.show()
```

ภาพประกอบ 31 โค้ดสำหรับเปรียบเทียบจำนวนใบแจ้งหนี้ที่รอการชำระและสถานการณ์ยกเลิกการใช้บริการ

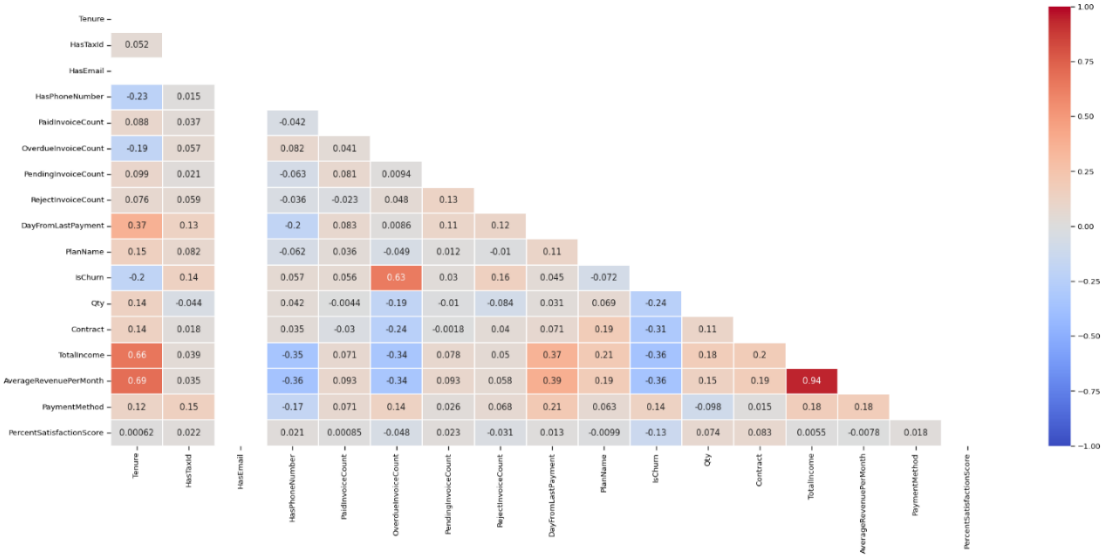
3.3.11 ความสัมพันธ์ระหว่างตัวแปรต่าง ๆ ในชุดข้อมูล

จากภาพ Heatmap (ภาพประกอบ 32 และ 33) ที่แสดงความสัมพันธ์ระหว่างตัวแปรต่างๆในชุดข้อมูล โดยเฉพาะการยกเลิกบริการ (Is Churn) พบว่าตัวแปรบางตัวมีความสัมพันธ์ที่เห็นได้ชัดกับการยกเลิกบริการ ขณะที่บางตัวแปรมีความสัมพันธ์ที่อ่อนหรือแทบไม่มีความสัมพันธ์เชิงเส้นเลย

การวิเคราะห์พบว่าลูกค้าที่มีระยะเวลาการใช้บริการ (Tenure) สั้นจะมีแนวโน้มที่จะยกเลิกบริการมากกว่าลูกค้าที่มีระยะเวลาการใช้บริการยาวนาน ค่าสหสัมพันธ์ระหว่าง Tenure และ Is Churn คือ -0.2 ซึ่งบ่งบอกถึงความสัมพันธ์เชิงลบที่แข็งแกร่ง นอกจากนี้การที่ลูกค้ามีค่าใช้จ่ายเฉลี่ยต่อเดือน (Average Revenue Per Month) สูงขึ้นก็สัมพันธ์กับการยกเลิกบริการที่มากขึ้น ค่าสหสัมพันธ์ในกรณีนี้คือ 0.69 ซึ่งแสดงถึงความสัมพันธ์ที่แข็งแกร่งในทิศทางบวก

อีกตัวแปรที่มีความสัมพันธ์กับการยกเลิกบริการคือ จำนวนใบแจ้งหนี้ค้างชำระ (Overdue Invoice Count) โดยลูกค้าที่มีใบแจ้งหนี้ค้างชำระมากจะมีแนวโน้มที่จะยกเลิกบริการมากขึ้น ค่าสหสัมพันธ์ระหว่าง Overdue Invoice Count และ Is Churn คือ 0.63 ซึ่งเป็นความสัมพันธ์เชิงบวกในระดับค่อนข้างสูง

ในส่วนของตัวแปรอื่นๆ เช่น การมีข้อมูลอีเมล (Has Email), และจำนวนใบแจ้งหนี้ที่ชำระแล้ว (Paid Invoice Count) มีความสัมพันธ์เชิงบวกหรือลบในระดับอ่อนกับการยกเลิกบริการ และอาจไม่สามารถใช้เป็นตัวบ่งชี้ที่แข็งแกร่งในการพยากรณ์การยกเลิกบริการได้



ภาพประกอบ 32 ความสัมพันธ์ระหว่างตัวแปรต่างๆ ในชุดข้อมูล

```
[128] df_corr = df.drop(['ClientId', 'CustAccount'], axis=1)

      corr = df_corr.apply(lambda x: pd.factorize(x)[0]).corr()
      mask = np.triu(np.ones_like(corr, dtype=bool))
      plt.figure(figsize=(25, 10))
      ax = sns.heatmap(corr, mask=mask, xticklabels=corr.columns, yticklabels=corr.columns,
                       annot=True, linewidths=.2, cmap='coolwarm', vmin=-1, vmax=1)
```

ภาพประกอบ 33 โค้ดสำหรับความสัมพันธ์ระหว่างตัวแปรต่างๆ ในชุดข้อมูล

จากการวิเคราะห์ข้อมูลเบื้องต้น (Exploratory Data Analysis: EDA) เพื่อสำรวจลักษณะของข้อมูลลูกค้าที่ใช้ในการศึกษาครั้งนี้ พบประเด็นที่มีความน่าสนใจและมีนัยสำคัญต่อการออกแบบแบบจำลองการพยากรณ์พฤติกรรมการยกเลิกใช้บริการของลูกค้า (Customer Churn) จำนวน 2 ประเด็นหลัก ได้แก่ 1.ความไม่สมดุลของข้อมูลเป้าหมาย (Imbalanced Data) และ 2. ความสัมพันธ์ระหว่างตัวแปรอิสระกับสถานะการยกเลิกบริการ

ในประเด็นแรก พบว่าสัดส่วนของลูกค้าที่มีสถานะยกเลิกใช้บริการ (IsChurn = 1) อยู่ที่ประมาณ 33.2% ขณะที่ลูกค้าที่คงใช้บริการ (IsChurn = 0) มีสัดส่วนถึง 66.8% ซึ่งสะท้อนถึงปัญหาคลาสไม่สมดุล (Class Imbalance) ที่อาจส่งผลให้โมเดลเรียนรู้ไม่แม่นยำ โดยเฉพาะในกลุ่มที่มีจำนวนตัวอย่างน้อยกว่า จึงจำเป็นต้องมีการจัดการด้วยเทคนิคสำหรับข้อมูลไม่สมดุลในขั้นตอนก่อนฝึกแบบจำลอง

ในประเด็นที่สอง การวิเคราะห์เชิงสัมพันธ์ระหว่างตัวแปรอิสระกับตัวแปรเป้าหมายพบว่าหลายฟีเจอร์มีความสัมพันธ์ในระดับที่น่าสนใจ เช่น ตัวแปร Tenure (ระยะเวลาการใช้บริการ) มี

ความสัมพันธ์เชิงลบกับการยกเลิกบริการ โดยลูกค้าที่ใช้บริการต่อเนื่องยาวนานมีแนวโน้มคงอยู่มากกว่า ในขณะที่ตัวแปร PaidInvoiceCount (จำนวนใบแจ้งหนี้ที่ชำระแล้ว) แสดงแนวโน้มในทิศทางเดียวกัน กล่าวคือ ลูกค้าที่มีประวัติการชำระเงินสม่ำเสมอมักคงอยู่กับบริการ ในทางกลับกัน OverdueInvoiceCount (จำนวนใบแจ้งหนี้ค้างชำระ) แสดงความสัมพันธ์เชิงบวกกับการเลิกใช้งาน โดยลูกค้าที่มีการค้างชำระหลายครั้งมีแนวโน้มที่จะยกเลิกบริการ นอกจากนี้ยังพบว่าคะแนนความพึงพอใจ (PercentSatisfactionScore) ที่อยู่ในระดับต่ำ มีความสัมพันธ์กับการเลิกใช้งานในระดับสูงอย่างมีนัยสำคัญ

ข้อค้นพบจากการวิเคราะห์ EDA นี้ จึงมีประโยชน์ในการคัดเลือกตัวแปรที่เกี่ยวข้อง เพื่อนำไปใช้ในการพัฒนาแบบจำลองการพยากรณ์ที่มีประสิทธิภาพ และสามารถสะท้อนพฤติกรรมของลูกค้าได้อย่างเหมาะสมในบริบทของการบริหารความสัมพันธ์กับลูกค้า (Customer Relationship Management)

3.4 การแบ่งชุดข้อมูลและจัดการกับความไม่สมดุลของข้อมูล (Imbalance Data)

ชุดข้อมูลถูกแบ่งออกเป็นชุดข้อมูลฝึก (train) และชุดข้อมูลทดสอบ (test) ในอัตราส่วน 70:30 (ภาพประกอบ 34) โดยมีจำนวนชุดข้อมูลฝึก 1,179 รายการ และในชุดข้อมูลทดสอบ 479 รายการตามลำดับ นอกจากนี้ยังได้มีการใช้เทคนิค Label Encoding เพื่อแปลงตัวแปรเชิงพาณิชย์ให้เป็นตัวแปรตัวบ่งชี้สำหรับทั้งชุดข้อมูลฝึกและชุดข้อมูลทดสอบ (อ้างอิงจากหัวข้อ 3.2.2)

```
[52] X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.30, random_state = 40, stratify=y)
```

ภาพประกอบ 34 โค้ดสำหรับการแบ่งชุดข้อมูลเป็นชุดฝึก (Training Set) และชุดทดสอบ (Testing Set)

จากการวิเคราะห์ข้อมูลเบื้องต้น (Exploratory Data Analysis: EDA) พบว่าข้อมูลที่ใช้ในการพัฒนาแบบจำลองมีลักษณะไม่สมดุลระหว่างกลุ่มเป้าหมาย (Target Variable) โดยเฉพาะในตัวแปร IsChurn ซึ่งแสดงสถานะการยกเลิกใช้บริการของลูกค้า พบว่ากลุ่มที่ยังคงใช้บริการ (IsChurn = 0) มีสัดส่วนสูงถึง 66.8% ในขณะที่กลุ่มที่ยกเลิกบริการ (IsChurn = 1) มีเพียง 33.2% ของข้อมูลทั้งหมด ซึ่งอาจส่งผลให้แบบจำลองเรียนรู้ลักษณะของกลุ่มเสียงส่วนใหญ่ได้ดี แต่ไม่สามารถทำนายกลุ่มเสียงส่วนน้อยได้อย่างแม่นยำ

เพื่อจัดการกับปัญหาดังกล่าว จึงได้ใช้เทคนิค Synthetic Minority Over-sampling Technique (SMOTE) ซึ่งเป็นวิธีการเพิ่มจำนวนข้อมูลในกลุ่มที่มีจำนวนน้อย (Minority Class)

โดยการสร้างตัวอย่างข้อมูลใหม่แบบสังเคราะห์จากข้อมูลจริงที่มีอยู่ ซึ่งช่วยให้โมเดลสามารถเรียนรู้ลักษณะของทั้งสองกลุ่มได้อย่างสมดุลมากขึ้น

จากภาพประกอบ 35 และ 36 แสดงถึงผลลัพธ์ก่อนและหลังการใช้ SMOTE กับชุดข้อมูลฝึก (Training Set) ที่ได้จากการแบ่งชุดข้อมูลในอัตราส่วน 70:30 พบว่าก่อนทำ SMOTE จำนวนกลุ่ม IsChurn = 0 (ไม่ยกเลิกบริการ) เท่ากับ 746 ราย และจำนวนกลุ่ม IsChurn =1 (ยกเลิกบริการ) เท่ากับ 367 ราย และหลังทำ SMOTE จำนวนข้อมูลในทั้งสองกลุ่มเท่ากัน คือ 746 ราย

```
Before SMOTE class distribution (y_train):
IsChurn
0    746
1    371
Name: count, dtype: int64

After SMOTE class distribution (y_train):
IsChurn
1    746
0    746
Name: count, dtype: int64
```

ภาพประกอบ 35 จำนวนข้อมูลก่อนและหลังทำ SMOTE

```
[ ] X = df.drop('IsChurn', axis=1)
    y = df['IsChurn']

    import pandas as pd
    df_full = pd.concat([X, y], axis=1)
    df_full = df_full.dropna(subset=['IsChurn'])

    X = df_full.drop('IsChurn', axis=1)
    y = df_full['IsChurn']

    from sklearn.model_selection import train_test_split

    X_train, X_test, y_train, y_test = train_test_split(
        X, y,
        test_size=0.3,
        random_state=42,
        stratify=y)

    print("Before SMOTE class distribution (y_train):")
    print(y_train.value_counts())

    from imblearn.over_sampling import SMOTE

    smote = SMOTE(random_state=42)
    X_train, y_train = smote.fit_resample(X_train, y_train)
    y_train = pd.Series(y_train)

    print("\nAfter SMOTE class distribution (y_train):")
    print(y_train.value_counts())
```

ภาพประกอบ 36 โค้ดสำหรับการทำ SMOTE

ผลลัพธ์นี้แสดงให้เห็นว่า SMOTE ได้ช่วยปรับสมดุลของข้อมูลระหว่างกลุ่มได้สำเร็จ ซึ่งจะช่วยเพิ่มความสามารถในการเรียนรู้ของแบบจำลอง และลดปัญหาการลำเอียงไปยังกลุ่มที่มีข้อมูลมากกว่า ทั้งนี้ การใช้ข้อมูลที่สมดุลจะเป็นประโยชน์ในการพัฒนาแบบจำลองที่มีความแม่นยำและครอบคลุมทั้งสองกลุ่มเป้าหมายอย่างเท่าเทียม

3.5 การสร้างแบบจำลอง และประเมินผลของแบบจำลอง

ในงานศึกษานี้จะใช้โมเดลการเรียนรู้ของเครื่อง (Machine Learning) ที่มีความเหมาะสมและประสิทธิภาพสูงในการทำนายการยกเลิกบริการของลูกค้า (Customer Churn) ซึ่งประกอบด้วย Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbors , และ AdaBoost Classifier

ก่อนที่จะนำโมเดลต่างๆ ไปใช้ทำนายจะมีขั้นตอนการเตรียมข้อมูลที่สำคัญ ได้แก่ การทำ Data Scaling และการ Transform คอลัมน์ โดยจะทำการสเกลเฉพาะคอลัมน์ที่เป็นข้อมูลเชิงตัวเลข (Numerical Columns) เพื่อปรับขนาดให้เท่ากันและลดอคติที่อาจเกิดจากช่วงค่าที่แตกต่างกันในแต่ละคอลัมน์ ซึ่งจะช่วยให้โมเดลสามารถเรียนรู้ได้ดีขึ้นและไม่มีความลำเอียงจากค่าที่มีช่วงกว้างหรือแคบเกินไปในบางคอลัมน์

3.5.1 Decision Tree

การสร้างแบบจำลองประเภท Decision Tree (ภาพประกอบ 37) จะใช้ไลบรารี scikit-learn เพื่อจำแนกข้อมูล (Classification) ซึ่งสามารถอธิบายลำดับการทำงานและการตั้งค่าของแบบจำลอง ขั้นตอนแรก คือการกำหนดค่าเริ่มต้นของแบบจำลองด้วย DecisionTreeClassifier() โดยมีการตั้งค่าพื้นฐานได้แก่ Gini Impurity (criterion='gini') เป็นตัววัดประสิทธิภาพในการแบ่งข้อมูลในแต่ละขั้นตอน เพื่อให้ได้กลุ่มข้อมูลที่บริสุทธิ์มากที่สุด โดยเลือกจุดตัดที่เหมาะสมที่สุด (splitter='best') จากนั้นฝึกสอนโมเดลด้วยข้อมูลฝึก (training set) โดยใช้คำสั่ง fit() และนำโมเดลที่ได้มาทดสอบประสิทธิภาพด้วยชุดข้อมูลทดสอบ (test set) โดยใช้คำสั่ง predict() เพื่อคาดการณ์ผลลัพธ์

```
[79] dt_model = DecisionTreeClassifier()
     dt_model.fit(X_train,y_train)
     predictdt_y = dt_model.predict(X_test)
     accuracy_dt = dt_model.score(X_test,y_test)
     print("Decision Tree accuracy is :",accuracy_dt)
```

ภาพประกอบ 37 โค้ดสำหรับ การสร้างแบบจำลอง Decision Tree

3.5.2 Random Forest

สำหรับการสร้างแบบจำลอง Random Forest (ภาพประกอบ 38) ใช้ไลบรารี scikit-learn โดยกำหนดค่าพารามิเตอร์ต่างๆ เพื่อควบคุมการเรียนรู้ของโมเดลให้เหมาะสมกับชุดข้อมูลที่ใช้โดยโมเดลถูกสร้างขึ้นด้วยคำสั่ง RandomForestClassifier() ซึ่งเป็นอัลกอริธึมในกลุ่ม Ensemble Learning ที่ใช้แนวคิดของการรวมผลจากหลายๆต้นไม้ (Decision Trees) เพื่อเพิ่ม

ความแม่นยำและลดความเสี่ยงจาก overfitting โดยได้กำหนดจำนวนต้นไม้ในป่า (forest) ไว้ที่ 500 ต้น (n_estimators=500) ส่วนของ oob_score=True เป็นการเปิดใช้งานการประเมินผลด้วยเทคนิค Out-of-Bag (OOB) ซึ่งช่วยให้สามารถประเมินประสิทธิภาพของโมเดลได้โดยไม่ต้องแบ่งชุดข้อมูลเพิ่มเติมสำหรับ validation พารามิเตอร์ n_jobs=-1 เป็นการกำหนดให้ใช้ทุกคอร์ของ CPU ที่มีอยู่ เพื่อเร่งความเร็วในการฝึกโมเดลให้เร็วที่สุด ได้กำหนด random_state=50 เพื่อควบคุมการสุ่ม ให้สามารถทำซ้ำผลลัพธ์ได้อย่างเสถียรและตรวจสอบซ้ำได้ ในส่วนของการเลือกจำนวนคุณลักษณะ (features) ที่ใช้ในการพิจารณาแต่ละการแบ่ง node ได้ตั้งค่าเป็น sqrt (max_features="sqrt") ซึ่งเป็นค่าที่เหมาะสมสำหรับงาน classification เพราะช่วยสร้างความหลากหลายให้กับแต่ละต้นไม้ภายใน forest และสุดท้ายได้กำหนด max_leaf_nodes=30 เพื่อจำกัดจำนวน leaf node สูงสุดในแต่ละต้นไม้ ช่วยควบคุมไม่ให้โมเดลซับซ้อนจนเกินไป และลดความเสี่ยงจากการเกิด overfitting

```
[88] model_rf = RandomForestClassifier(n_estimators=500, oob_score=True, n_jobs=-1,
                                       random_state=50, max_features="sqrt",
                                       max_leaf_nodes=30)
     model_rf.fit(X_train, y_train)

     prediction_test = model_rf.predict(X_test)
     print("Random Forest accuracy is",metrics.accuracy_score(y_test, prediction_test))
```

ภาพประกอบ 38 โค้ดสำหรับ การสร้างแบบจำลอง Random Forest

3.5.3 Support Vector Machine

ในการสร้างแบบจำลองการจำแนกข้อมูลครั้งนี้ (ภาพประกอบ 39) ได้เลือกใช้อัลกอริธึม Support Vector Classification (SVC) ซึ่งเป็นหนึ่งในรูปแบบของ Support Vector Machine (SVM) โดยมีการจัดการกับข้อมูลที่มีค่า missing ด้วยวิธีการแทนที่ (imputation) ก่อนการฝึกโมเดลเพื่อให้สามารถประมวลผลได้อย่างสมบูรณ์

ขั้นตอนแรกเริ่มด้วยการจัดการค่า missing ในชุดข้อมูลโดยใช้คลาส SimpleImputer จากไลบรารี sklearn.impute ซึ่งตั้งค่าให้ใช้ค่าเฉลี่ย (strategy='mean') ของแต่ละคอลัมน์ในการแทนที่ค่าที่หายไป จากนั้นจึงแปลงชุดข้อมูลทั้ง training และ test ให้พร้อมใช้งานผ่านคำสั่ง fit_transform() และ transform() ตามลำดับ

หลังจากเตรียมข้อมูลเรียบร้อยแล้ว โมเดล SVM ถูกสร้างด้วยคลาส SVC() โดยในที่นี้ได้กำหนดค่า random_state=1 เพื่อควบคุมความสุ่มในการประมวลผล

เมื่อโมเดลได้รับการฝึกกับข้อมูล X_train_imputed และ y_train เสร็จเรียบร้อยแล้ว จึงนำไปใช้ในการทำนายผลบนชุดข้อมูลทดสอบ X_test_imputed

```
[65] imputer = SimpleImputer(strategy='mean')

     X_train_imputed = imputer.fit_transform(X_train)
     X_test_imputed = imputer.transform(X_test)

     svc_model = SVC(random_state = 1)
     svc_model.fit(X_train_imputed, y_train)

     predict_y = svc_model.predict(X_test_imputed)
     accuracy_svc = svc_model.score(X_test_imputed, y_test)
     print("SVM accuracy is:", accuracy_svc)
```

ภาพประกอบ 39 โค้ดสำหรับ การสร้างแบบจำลอง Support Vector Machine

3.5.4 Gradient Boosting Classifier

จากภาพประกอบ 40 โมเดลถูกสร้างขึ้นด้วยคำสั่ง GradientBoostingClassifier() ซึ่งใช้ค่าพารามิเตอร์เริ่มต้นทั้งหมด (default settings) โดยพารามิเตอร์หลักที่กำหนด ได้แก่ n_estimators=100, learning_rate=0.1, max_depth=3, subsample=1.0, random_state =None และ ไม่ได้กำหนด seed หลังจากฝึกโมเดลด้วยข้อมูลฝึกแล้ว โมเดลถูกนำไปใช้ในการทำนายผลบนชุดข้อมูลทดสอบ และวัดความแม่นยำด้วยฟังก์ชัน accuracy_score() จากไลบรารี sklearn.metrics

```
[94] gb = GradientBoostingClassifier()
     gb.fit(X_train, y_train)
     gb_pred = gb.predict(X_test)
     print("Gradient Boosting Classifier", accuracy_score(y_test, gb_pred))
```

ภาพประกอบ 40 โค้ดสำหรับ การสร้างแบบจำลอง Gradient Boosting Classifier

3.5.5 Logistic Regression

จากภาพประกอบ 41 การสร้างแบบจำลอง Logistic Regression โดยใช้ไลบรารี sklearn.linear_model ซึ่งเป็นอัลกอริธึมประเภทจำแนกข้อมูลแบบเชิงเส้น (linear classification model) ที่ใช้สมการ Logistic ในการประมาณความน่าจะเป็นของคลาสเป้าหมาย และเหมาะสำหรับงานที่มีตัวแปรเป้าหมายเป็นค่าจำแนก (classification) เช่น คลาส 0 และ 1

โมเดลถูกสร้างขึ้นโดยใช้คลาส LogisticRegression() มีการตั้งค่าพารามิเตอร์ ได้แก่ penalty='l2', solver='lbfgs', max_iter=100

หลังจากฝึกโมเดลด้วยข้อมูล X_train และ y_train แล้วโมเดลถูกนำไปทดสอบกับชุดข้อมูล X_test เพื่อประเมินประสิทธิภาพ

```
[75] lr_model = LogisticRegression()
     lr_model.fit(X_train,y_train)
     accuracy_lr = lr_model.score(X_test,y_test)
     print("Logistic Regression accuracy is :",accuracy_lr)
```

ภาพประกอบ 41 โค้ดสำหรับ การสร้างแบบจำลอง Logistic Regression

3.5.6 K-Nearest Neighbors

จากภาพประกอบ 42 ก่อนที่จะฝึกโมเดล KNN ได้มีการจัดการกับข้อมูลที่มีค่าว่าง (missing values) ด้วยเทคนิค mean imputation โดยใช้คลาส SimpleImputer จากไลบรารี sklearn.impute ซึ่งมีการกำหนด strategy='mean' เพื่อแทนค่าที่ขาดหายด้วยค่าเฉลี่ยของแต่ละคอลัมน์ในชุดข้อมูลฝึก (training set) เมื่อข้อมูลได้รับการจัดการเรียบร้อยแล้ว จึงนำไปฝึกด้วยโมเดล KNeighborsClassifier ซึ่งเป็น KNN classifier ในไลบรารี scikit-learn โดยได้ตั้งค่า n_neighbors=11

หลังจากฝึกโมเดลกับข้อมูลที่ถูกแทนค่าค่าว่างแล้ว จึงนำโมเดลไปใช้กับชุดข้อมูลทดสอบ (X_test_imputed) และประเมินประสิทธิภาพของโมเดลด้วยฟังก์ชัน score() ซึ่งจะคำนวณ accuracy หรืออัตราความแม่นยำของการจำแนกผลลัพธ์ที่ถูกต้องเทียบกับทั้งหมด

```
[ ] from sklearn.impute import SimpleImputer

    imputer = SimpleImputer(strategy='mean')

    X_train_imputed = imputer.fit_transform(X_train)
    X_test_imputed = imputer.transform(X_test)

    knn_model = KNeighborsClassifier(n_neighbors=11)
    knn_model.fit(X_train_imputed, y_train)
    predicted_y = knn_model.predict(X_test_imputed)
    accuracy_knn = knn_model.score(X_test_imputed, y_test)
    print("KNN accuracy:", accuracy_knn)
```

ภาพประกอบ 42 โค้ดสำหรับ การสร้างแบบจำลอง K-Nearest Neighbors

3.5.4 AdaBoost Classifier

จากภาพประกอบ 43 ได้เรียกใช้งานคลาส AdaBoostClassifier() จากไลบรารี sklearn.ensemble ซึ่งโดยค่าเริ่มต้นจะใช้ Decision Tree ขนาดเล็ก (ความลึก = 1) เป็นแบบจำลองพื้นฐาน (base estimator หรือ weak learner) และใช้วิธีการค่อย ๆ ปรับน้ำหนักของตัวอย่างที่จำแนกผิดในแต่ละรอบ เพื่อให้แบบจำลองในรอบถัดไปให้ความสำคัญกับตัวอย่างเหล่านั้นมากขึ้น พารามิเตอร์เริ่มต้นที่ใช้ในการฝึกโมเดลนี้ ได้แก่ n_estimators=50 ,

learning_rate=1.0, base_estimator=DecisionTreeClassifier(max_depth=1) , algorith = 'SAMME.R'

หลังจากโมเดลถูกฝึกด้วยข้อมูล X_train และ y_train แล้ว โมเดลจะถูกใช้ในการทำนายผลบนชุดข้อมูล X_test และประเมินประสิทธิภาพของโมเดลด้วยฟังก์ชัน accuracy_score() ซึ่งเป็นตัวชี้วัดสัดส่วนของการจำแนกที่ถูกต้อง

```
[ ]  a_model = AdaBoostClassifier()
     a_model.fit(X_train,y_train)
     a_preds = a_model.predict(X_test)
     print("AdaBoost Classifier accuracy")
     metrics.accuracy_score(y_test, a_preds)
```

ภาพประกอบ 43 โค้ดสำหรับ การสร้างแบบจำลอง AdaBoost Classifier

สำหรับการประเมินผลโมเดลต่างๆ จะใช้ตัวชี้วัดที่สำคัญ ได้แก่ Accuracy Score, Recall Score, Precision Score, F1 Score, Receiver Operating Characteristic (ROC) หลังนั้นผลลัพธ์จะถูกใช้ในการเลือกโมเดลที่มีประสิทธิภาพดีที่สุดสำหรับการสร้างโมเดลรวม (Emsemble Learning) เพื่อทำนายการยกเลิกบริการของลูกค้า

3.6 การปรับเทียบโมเดล (Fine tune) สำหรับการรวมโมเดล (Ensemble learning)

ในการศึกษาครั้งนี้ได้สร้างแบบจำลอง 7 แบบจำลอง ได้แก่ Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbor และ AdaBoost Classifier โดยมีวัตถุประสงค์เพื่อเปรียบเทียบประสิทธิภาพของอัลกอริธึมแต่ละประเภทภายใต้ชุดข้อมูลเดียวกัน และพิจารณาเลือกแบบจำลองที่มีสมรรถนะสูงสำหรับนำไปใช้ในกระบวนการเรียนรู้แบบรวม (Ensemble Learning)

หลังจากการประเมินผลเบื้องต้น พบว่าแบบจำลองที่ให้ผลลัพธ์ในระดับที่น่าพึงพอใจและมีลักษณะเสริมกันอย่างเหมาะสม ได้แก่ AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier ซึ่งได้รับการคัดเลือกเพื่อนำไปสร้างเป็นแบบจำลองแบบผสมโดยใช้เทคนิค Voting Classifier โดยวิธีการรวมผลลัพธ์ของแต่ละโมเดลด้วยหลักการ Voting เพื่อให้ได้คำตอบสุดท้ายที่มีความแม่นยำและมีเสถียรภาพมากยิ่งขึ้น

ทั้งนี้ เพื่อให้ได้แบบจำลองผสมที่มีประสิทธิภาพสูงสุด ได้มีการดำเนินการปรับเทียบค่าพารามิเตอร์ของแบบจำลองย่อย (Fine-tuning) ทั้งสามโมเดลด้วยเทคนิค Grid Search Cross-Validation (GridSearchCV) [26] ภายใต้กระบวนการ Pipeline ของไลบรารี scikit-learn ซึ่งประกอบด้วยขั้นตอนการปรับมาตรฐานข้อมูล (Standardization) ด้วย StandardScaler และการฝึกโมเดลแต่

ละประเภท โดยมีการกำหนดช่วงของพารามิเตอร์ (Parameter Grid) ที่เหมาะสมสำหรับการค้นหาค่าที่ให้ผลลัพธ์ดีที่สุด

การสร้าง Pipeline ที่กล่าวก่อนหน้านี้มีจุดประสงค์สำคัญเพื่อป้องกันการเกิดข้อมูลรั่วไหล (Data Leakage) [27] ระหว่างกระบวนการเตรียมข้อมูลและการฝึกโมเดล ซึ่งอาจเกิดขึ้นได้หากมีการแปลงค่าหรือปรับมาตรฐานข้อมูลนอกกระบวนการเรียนรู้แบบไขว้ (Cross-validation) โดย Pipeline จะทำให้การประมวลผลแต่ละขั้นเป็นอิสระและถูกนำไปใช้ภายใต้กรอบการประเมินที่ถูกต้อง

โค้ดที่ใช้ในการดำเนินการแสดงในภาพประกอบ 44 มีโครงสร้างประกอบด้วยการกำหนด Pipeline ของแต่ละโมเดล พร้อมพารามิเตอร์ที่ใช้ในการค้นหา จากนั้นใช้ GridSearchCV ทำการฝึกและประเมินผลโดยใช้ค่าความแม่นยำ (Accuracy) เป็นเกณฑ์หลัก และแสดงผลการประเมินด้วย Classification Report เพื่อเปรียบเทียบประสิทธิภาพของแต่ละแบบจำลองภายใต้ค่าพารามิเตอร์ที่เหมาะสมที่สุด จากนั้นผลลัพธ์ที่ได้จากการปรับค่าพารามิเตอร์ดังกล่าวจะถูกนำไปใช้ในขั้นตอนถัดไปของการสร้างโมเดล Ensemble Learning ต่อไป

```python
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, AdaBoostClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report

models_params = {
    "Decision Tree": {
        "pipeline": Pipeline([
            ('scaler', StandardScaler()),
            ('model', DecisionTreeClassifier(random_state=42))
        ]),
        "param_grid": {
            'model__max_depth': [3, 5, 10],
            'model__min_samples_split': [2, 5, 10]
        }
    },
    "Gradient Boosting": {
        "pipeline": Pipeline([
            ('scaler', StandardScaler()),
            ('model', GradientBoostingClassifier(random_state=42))
        ]),
        "param_grid": {
            'model__n_estimators': [100, 200],
            'model__learning_rate': [0.05, 0.1],
            'model__max_depth': [3, 5]
        }
    },
    "AdaBoost": {
        "pipeline": Pipeline([
            ('scaler', StandardScaler()),
            ('model', AdaBoostClassifier(random_state=42))
        ]),
        "param_grid": {
            'model__n_estimators': [50, 100],
            'model__learning_rate': [0.5, 1.0]
        }
    }
}

best_models = {}

for name, cfg in models_params.items():
    print(f"\n🔍 Training and tuning: {name}")
    grid = GridSearchCV(
        estimator=cfg["pipeline"],
        param_grid=cfg["param_grid"],
        cv=5,
        scoring='accuracy',
        n_jobs=-1
    )
    grid.fit(X_train, y_train)
    best_models[name] = grid.best_estimator_

    y_pred = grid.predict(X_test)
    print(f"\n Classification Report for {name}:")
    print(classification_report(y_test, y_pred))
```

ภาพประกอบ 44 โค้ดสำหรับปรับเทียบแบบจำลองและการสร้าง Pipeline

3.7 **การรวมโมเดล** (Ensemble Learning)

ในการศึกษานี้มีการประยุกต์ใช้เทคนิคการรวมโมเดล (Ensemble Learning) เพื่อเพิ่มประสิทธิภาพในการพยากรณ์ผลการยกเลิกบริการของลูกค้า โดยเลือกใช้วิธี Voting Classifier

ขั้นตอนเริ่มต้นคือการเลือกโมเดลที่มีผลการทำนายที่ดีในเบื้องต้นจำนวน 3 โมเดล ได้แก่ AdaBoostClassifier, DecisionTreeClassifier, และ GradientBoostingClassifier ซึ่งเป็นโมเดลในกลุ่มของ Ensemble เช่นเดียวกัน และมีคุณสมบัติที่สามารถจัดการกับข้อมูลที่ไม่เป็นเชิงเส้นและมีความซับซ้อนได้อย่างมีประสิทธิภาพ โมเดลทั้งสามถูกกำหนดด้วยพารามิเตอร์ random_state=50 เพื่อให้ได้ผลลัพธ์ที่สามารถทำซ้ำได้ (reproducible results)

จากนั้นได้ทำการรวมโมเดลทั้งสามเข้าด้วยกันผ่านคลาส VotingClassifier จากไลบรารี sklearn.ensemble โดยกำหนดพารามิเตอร์ voting='soft' ซึ่งหมายถึงการใช้ Soft Voting ในการรวมผลลัพธ์ ทำให้โมเดลแต่ละตัวจะให้ค่าความน่าจะเป็น (probability) ของแต่ละคลาสออกมา และ VotingClassifier จะรวมความน่าจะเป็นเหล่านั้นโดยการเฉลี่ย และเลือกคลาสที่มีค่าความน่าจะเป็นรวมสูงที่สุดเป็นคำตอบสุดท้าย ซึ่งต่างจาก Hard Voting ที่จะนับเสียงข้างมากของผลลัพธ์แต่ละโมเดลโดยไม่สนใจระดับความมั่นใจ

หลังจากการรวมโมเดลเสร็จสิ้น ได้ทำการฝึกโมเดลที่รวมแล้ว (eclf1) กับชุดข้อมูลฝึก (X_train, y_train) และนำโมเดลดังกล่าวไปใช้พยากรณ์กับชุดข้อมูลทดสอบ (X_test) จากนั้นประเมินผลลัพธ์ด้วย Accuracy Score และ Classification Report ซึ่งครอบคลุมค่า Precision, Recall, F1-score ของแต่ละคลาส

การเลือกใช้เทคนิค Ensemble โดยเฉพาะ Soft Voting ช่วยให้สามารถใช้จุดแข็งของแต่ละโมเดลร่วมกัน ลดผลกระทบของข้อด้อยเฉพาะของโมเดลใดโมเดลหนึ่ง และเพิ่มความแม่นยำโดยรวมของการทำนาย ซึ่งเหมาะสมกับปัญหาที่มีข้อมูลซับซ้อนและไม่สมดุลอย่างการทำนายการยกเลิกบริการของลูกค้าในงานศึกษานี้

```
[90] from sklearn.ensemble import VotingClassifier
     clf1 = AdaBoostClassifier(random_state=50)
     clf2 = DecisionTreeClassifier(random_state=50)
     clf3 = GradientBoostingClassifier(random_state=50)


     eclf1 = VotingClassifier(estimators=[('rf', clf1), ('dt', clf2), ('gb', clf3)], voting='soft')
     eclf1.fit(X_train, y_train)
     predictions = eclf1.predict(X_test)


     print("Final Accuracy Score ")
     print(accuracy_score(y_test, predictions))
     print(classification_report(y_test, predictions))
```

ภาพประกอบ 45 โค้ดสำหรับการรวมโมเดล (Ensemble Learning)

บทที่ 4
ผลการดำเนินวิจัย

จากการดำเนินงานวิจัยในครั้งนี้ได้ใช้แบบจำลองที่ใช้ในการศึกษา ได้แก่ Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbors, AdaBoost Classifier, และ Ensemble Voting Classifier ซึ่งเป็นการประยุกต์ใช้เทคนิคการเรียนรู้ของเครื่อง (Machine Learning) เพื่อพัฒนาแบบจำลองสำหรับทำนายพฤติกรรมการยกเลิกการใช้บริการของลูกค้า (Customer Churn) โดยทำการทดลองใช้งานแบบจำลองต่างๆ กับชุดข้อมูลจริง และประเมินประสิทธิภาพของแบบจำลองเหล่านั้นด้วยตัวชี้วัดทางสถิติด้านการจำแนกประเภท เพื่อคัดเลือกโมเดลที่เหมาะสมที่สุดสำหรับข้อมูลของผู้วิจัย ผลการวิจัยประกอบด้วยค่าประเมินจากแบบจำลอง 8 แบบ และค่าประเมินจากการรวมแบบจำลอง (Ensemble Model) เพื่อเปรียบเทียบประสิทธิภาพ

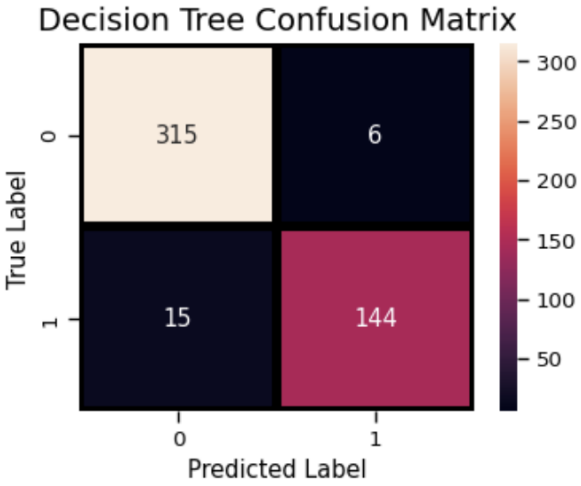
4.1 Decision Tree

ผลการประเมินโมเดล Decision Tree มีความแม่นยำโดยรวม (Accuracy) อยู่ที่ 95.21% (ภาพประกอบ 46) เป็นตัวเลขที่ค่อนข้างสูง แสดงให้เห็นว่าโมเดลมีความสามารถในการจำแนกค่อนข้างดี

```
[77] dt_model = DecisionTreeClassifier()
     dt_model.fit(X_train,y_train)
     predictdt_y = dt_model.predict(X_test)
     accuracy_dt = dt_model.score(X_test,y_test)
     print("Decision Tree accuracy is :",accuracy_dt)

     Decision Tree accuracy is : 0.9520833333333333
```

ภาพประกอบ 46 ค่า Accuracy Score ของ Decision tree

เมื่อพิจารณารายละเอียดของผลลัพธ์จาก confusion matrix (ภาพประกอบ 47 และ 48) จะพบว่า โมเดลสามารถจำแนกกลุ่มตัวอย่างที่เป็นกลุ่มลบ (Class 0) ได้ดีมาก โดยมี True Negative จำนวน 315 ตัวอย่าง และ False Positive จำนวน 6 ตัวอย่าง แสดงให้เห็นว่าโมเดลผิดพลาดในการจำแนกกลุ่มลบน้อยมาก ในขณะเดียวกัน โมเดลยังมีความสามารถในการจำแนกกลุ่มบวก (Class 1) ที่ดีเช่นกัน โดยสามารถจำแนกได้ถูกต้อง (True Positive) จำนวน 144 ตัวอย่าง แต่ก็มีการจำแนกผิดพลาด (False Negative) จำนวน 15 ตัวอย่าง



ภาพประกอบ 47 confusion matrix ของ Decision tree

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(4, 3))
sns.heatmap(confusion_matrix(y_test, predictdt_y),
            annot=True, fmt="d", linecolor="k", linewidths=3)

plt.title("Decision Tree Confusion Matrix", fontsize=14)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```

ภาพประกอบ 48 โค้ดสำหรับ confusion matrix ของ Decision tree

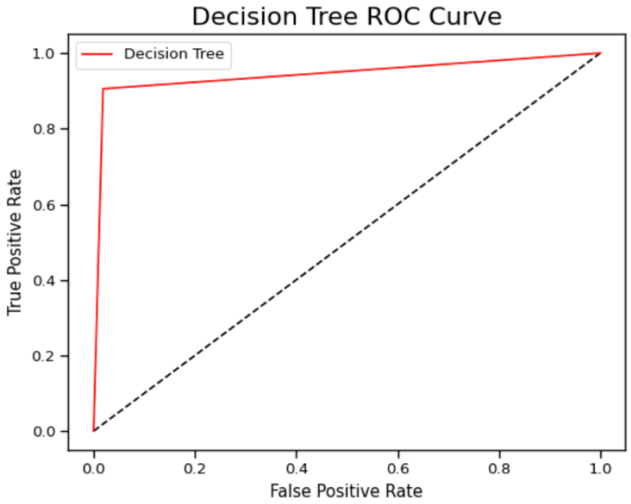
จากค่า Precision, Recall และ F1-score ของแต่ละกลุ่ม (ภาพประกอบ 49) กลุ่มลบมีค่า Precision และ Recall อยู่ที่ประมาณ 95% และ 98% ตามลำดับ ขณะที่กลุ่มบวกมีค่า Precision 95% และ Recall 91% แสดงว่าโมเดลมีความสามารถในการจำแนกข้อมูลกลุ่มลบสูงกว่าเล็กน้อยเมื่อเปรียบเทียบกับกลุ่มบวก โดย ซึ่งหมายถึงโมเดลสามารถระบุกลุ่มตัวอย่างที่เป็นบวกได้ค่อนข้างแม่นยำ (Precision สูง)

```
[78] print(classification_report(y_test, predictdt_y))
```

	precision	recall	f1-score	support
0	0.95	0.98	0.96	321
1	0.95	0.91	0.93	159
accuracy			0.95	480
macro avg	0.95	0.94	0.95	480
weighted avg	0.95	0.95	0.95	480

ภาพประกอบ 49 ค่า Precision, Recall และ F1-score ของ Decision tree

จากกราฟ ROC Curve (ภาพประกอบ 50 และ 51) แสดงให้เห็นถึงความสามารถในการจำแนกของโมเดลที่ดีมาก โดยมีพื้นที่ใต้กราฟ (Area Under Curve หรือ AUC) สูง แสดงถึงประสิทธิภาพในการแยกแยะระหว่างกลุ่มที่เป็นบวกและกลุ่มลบได้อย่างชัดเจน



ภาพประกอบ 50 ROC Curve ของ Decision tree

```
[79] from sklearn.metrics import roc_curve
     import matplotlib.pyplot as plt

     y_pred_prob_dt = dt_model.predict_proba(X_test)[:, 1]

     fpr_dt, tpr_dt, thresholds_dt = roc_curve(y_test, y_pred_prob_dt)

     plt.plot([0, 1], [0, 1], 'k--')  # reference line
     plt.plot(fpr_dt, tpr_dt, label='Decision Tree', color="r")
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('Decision Tree ROC Curve', fontsize=16)
     plt.legend()
     plt.show()
```

ภาพประกอบ 51 โค้ดสำหรับ ROC Curve ของ Decision tree

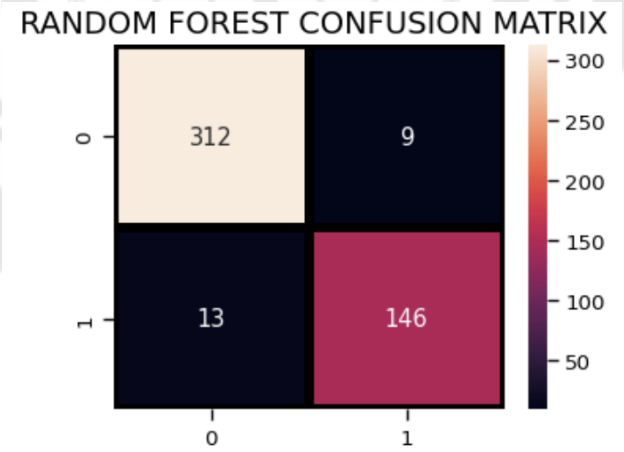
4.2 Random Forest

จากผลการประเมินแบบจำลอง Random Forest พบว่าโมเดลนี้มีความแม่นยำ (Accuracy) สูงถึง 95.42% (ภาพประกอบ 52) ซึ่งเป็นค่าที่ค่อนข้างสูง บ่งชี้ว่าแบบจำลองมีประสิทธิภาพในการจำแนกกลุ่มข้อมูลที่ดีมาก โดยจำแนกได้ถูกต้องเป็นส่วนใหญ่จากข้อมูลทั้งหมด 480 ตัวอย่างที่นำมาทดสอบ

```
[69] model_rf = RandomForestClassifier(n_estimators=500, oob_score=True, n_jobs=-1,
                                       random_state=50, max_features="sqrt",
                                       max_leaf_nodes=30)
     model_rf.fit(X_train, y_train)

     prediction_test = model_rf.predict(X_test)
     print("Random Forest accuracy is",metrics.accuracy_score(y_test, prediction_test))

Random Forest accuracy is 0.9541666666666667
```

ภาพประกอบ 52 Accuracy score ของ Random forest

เมื่อลงลึกถึงรายละเอียดใน Confusion matrix (ภาพประกอบ 53) พบว่ากลุ่มตัวอย่างที่เป็นกลุ่มลบ (Class 0) ถูกจำแนกถูกต้องถึง 312 ตัวอย่าง (True Negative) และจำแนกผิดพลาดเป็นกลุ่มบวกเพียง 9 ตัวอย่าง (False Positive) สะท้อนให้เห็นว่าโมเดลมีประสิทธิภาพในการระบุผู้ที่ไม่ใช่กลุ่มเป้าหมายได้อย่างแม่นยำมาก สำหรับกลุ่มตัวอย่างที่เป็นบวก (Class 1) โมเดลจำแนกได้ถูกต้องจำนวน 146 ตัวอย่าง (True Positive) แต่มีตัวอย่างที่โมเดลจำแนกผิดพลาดเป็นกลุ่มลบจำนวน 13 ตัวอย่าง (False Negative) ซึ่งยังคงแสดงให้เห็นถึงประสิทธิภาพที่ค่อนข้างสูง แต่ยังคงมีจุดที่อาจจะปรับปรุงได้อีก โดยเฉพาะในบริบทที่การพลาดการจำแนกกลุ่มบวกมีต้นทุนหรือความเสี่ยงสูง



ภาพประกอบ 53 Confusion matrix ของ Random forest

```
plt.figure(figsize=(4,3))
sns.heatmap(confusion_matrix(y_test, prediction_test),
                annot=True,fmt = "d",linecolor="k",linewidths=3)

plt.title(" RANDOM FOREST CONFUSION MATRIX",fontsize=14)
plt.show()
```

ภาพประกอบ 54 โค้ดสำหรับ confusion matrix ของ Random forest

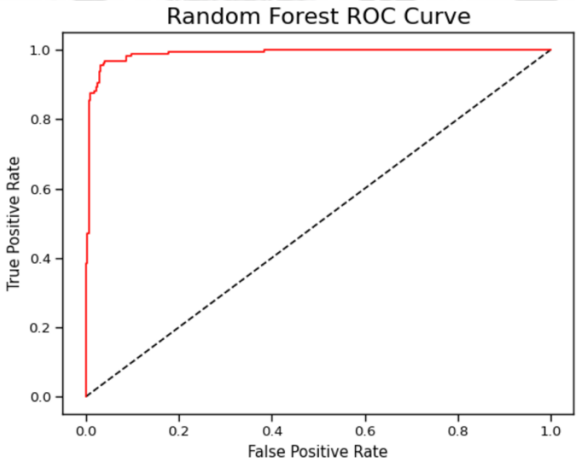
เมื่อพิจารณาค่า Precision และ Recall (ภาพประกอบ 55) พบว่าโมเดล Random Forest มีค่า Precision ของกลุ่มลบอยู่ที่ 96% และค่า Recall 97% ซึ่งหมายความว่าโมเดลสามารถจำแนกกลุ่มลบได้แม่นยำและครอบคลุมสูง ส่วนกลุ่มบวกนั้น มีค่า Precision 94% และค่า Recall 92% แสดงว่าแม้โมเดลจะมีความแม่นยำในการระบุกลุ่มบวกดี แต่ความครอบคลุมหรือความสามารถในการตรวจจับกลุ่มบวกทั้งหมดอาจจะลดลงเล็กน้อยเมื่อเทียบกับกลุ่มลบ

```
[70] print(classification_report(y_test, prediction_test))
```

	precision	recall	f1-score	support
0	0.96	0.97	0.97	321
1	0.94	0.92	0.93	159
accuracy			0.95	480
macro avg	0.95	0.95	0.95	480
weighted avg	0.95	0.95	0.95	480

ภาพประกอบ 55 ค่า Precision, Recall และ F1-score ของ Random forest

ในส่วนของกราฟ ROC Curve (ภาพประกอบ 56 และ 57) พบว่าพื้นที่ใต้กราฟ (AUC) มีค่าสูงสะท้อนถึงประสิทธิภาพที่ดีของโมเดลในการแยกแยะระหว่างกลุ่มบวกและกลุ่มลบ โดยที่อัตราการเกิด False Positive ต่ำเมื่อเปรียบเทียบกับอัตราการเกิด True Positive ซึ่งเป็นคุณลักษณะที่ดีของโมเดลที่ใช้ในการจำแนกข้อมูล



ภาพประกอบ 56 ROC Curve ของ Random forest

```
y_rfpred_prob = model_rf.predict_proba(X_test)[:,1]
fpr_rf, tpr_rf, thresholds = roc_curve(y_test, y_rfpred_prob)
plt.plot([0, 1], [0, 1], 'k--' )
plt.plot(fpr_rf, tpr_rf, label='Random Forest',color = "r")
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Random Forest ROC Curve',fontsize=16)
plt.show();
```

ภาพประกอบ 57 โค้ดสำหรับ ROC Curve ของ Random forest

โดยสรุปแล้ว โมเดล Random Forest นี้ถือว่าเป็นโมเดลที่มีประสิทธิภาพสูงในการจำแนกข้อมูล โดยมีความแม่นยำและความสมดุลในการจำแนกกลุ่มลบและกลุ่มบวกที่ดีมาก แต่ในการใช้งานจริงจำเป็นต้องพิจารณาความสำคัญของผลลัพธ์ที่ผิดพลาด (False Negative) เพิ่มเติม โดยเฉพาะหากงานที่นำไปใช้มีความอ่อนไหวต่อการพลาดกลุ่มบวก อาจต้องมีการปรับจุดตัด (threshold) หรือปรับแต่งโมเดลเพิ่มเติมเพื่อเพิ่มความสามารถในการตรวจจับกลุ่มบวกให้มีประสิทธิภาพมากขึ้น

4.3 Support Vector Machine

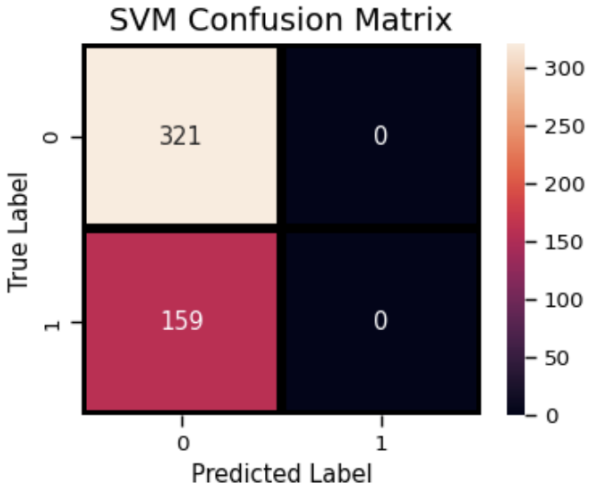
จากผลการประเมินโมเดล Support Vector Machine (SVM) พบว่าโมเดลมีค่าความแม่นยำ (Accuracy) อยู่ที่ประมาณ 66.88% (ภาพประกอบ 58) ซึ่งถือว่าต่ำเมื่อเทียบกับโมเดล Decision Tree และ Random Forest ที่ประเมินไปก่อนหน้านี้

```
[65] imputer = SimpleImputer(strategy='mean')

     X_train_imputed = imputer.fit_transform(X_train)
     X_test_imputed = imputer.transform(X_test)

     svc_model = SVC(random_state = 1)
     svc_model.fit(X_train_imputed, y_train)

     predict_y = svc_model.predict(X_test_imputed)
     accuracy_svc = svc_model.score(X_test_imputed, y_test)
     print("SVM accuracy is:", accuracy_svc)

SVM accuracy is: 0.66875
```

ภาพประกอบ 58 Accuracy score ของ Support Vector Machine

เมื่อพิจารณา Confusion matrix (ภาพประกอบ 59 และ 60) จะเห็นได้ชัดเจนว่าโมเดลนี้มีปัญหาในการแยกแยะกลุ่มบวก (Class 1) ออกจากกลุ่มลบ (Class 0) อย่างมาก โดยโมเดลทำนายกลุ่มตัวอย่างทั้งหมดเป็นกลุ่มลบ (Class 0) เท่านั้น ส่งผลให้ True Negative (ทำนายถูกใน

กลุ่มลบ) มีจำนวนถึง 321 ตัวอย่าง ในขณะที่ False Negative มีจำนวนมากถึง 159 ตัวอย่าง ซึ่งสะท้อนว่าโมเดลนี้ไม่สามารถจำแนกกลุ่มบวกได้เลย ทำให้ Recall และ Precision สำหรับกลุ่มบวกมีค่าเป็นศูนย์



ภาพประกอบ 59 Confusion matrix ของ Support Vector Machine

```
[68] from sklearn.metrics import confusion_matrix
     import seaborn as sns
     import matplotlib.pyplot as plt

     predict_y = svc_model.predict(X_test_imputed)

     cm = confusion_matrix(y_test, predict_y)

     plt.figure(figsize=(4, 3))
     sns.heatmap(cm, annot=True, fmt="d", linewidths=3, linecolor="k")
     plt.title("SVM Confusion Matrix", fontsize=14)
     plt.xlabel('Predicted Label')
     plt.ylabel('True Label')
     plt.show()
```

ภาพประกอบ 60 โค้ดสำหรับ Confusion matrix ของ Support Vector Machine

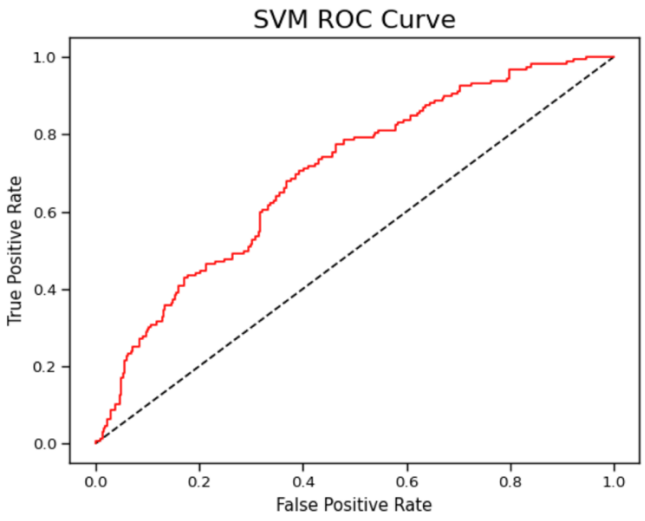
ค่า Precision และ Recall ในกลุ่มลบมีค่าสูงที่ 67% และ 100% ตามลำดับ (ภาพประกอบ 61) ซึ่งแม้จะดูดีแต่เป็นเพียงผลจากการที่โมเดลทำนายทุกตัวอย่างว่าเป็นกลุ่มลบทั้งหมดเท่านั้น ทำให้โมเดลนี้ขาดประสิทธิภาพในการนำไปใช้จริง โดยเฉพาะในกรณีที่กลุ่มเป้าหมาย (Class 1) มีความสำคัญสูงและจำเป็นต้องได้รับการระบุให้ถูกต้อง

```
[66] print(classification_report(y_test, predict_y))
```

	precision	recall	f1-score	support
0	0.67	1.00	0.80	321
1	0.00	0.00	0.00	159
accuracy			0.67	480
macro avg	0.33	0.50	0.40	480
weighted avg	0.45	0.67	0.54	480

ภาพประกอบ 61 Precision, Recall และ F1-score ของ Support Vector Machine

จากกราฟ ROC Curve ที่แสดงพื้นที่ใต้กราฟ (AUC) นั้น โมเดล SVM นี้มีค่า AUC ที่ต่ำมาก (ภาพประกอบ 62 และ 63) ซึ่งสะท้อนถึงความสามารถที่ต่ำในการแยกแยะข้อมูลระหว่างสองกลุ่มให้ชัดเจน ถือเป็นข้อบกพร่องที่ชัดเจนของโมเดลนี้



ภาพประกอบ 62 ROC Curve ของ Support Vector Machine

```
[67] from sklearn.metrics import roc_curve
     import matplotlib.pyplot as plt

     svc_model = SVC(random_state=1, probability=True)
     svc_model.fit(X_train_imputed, y_train)

     y_pred_prob = svc_model.predict_proba(X_test_imputed)[:, 1]

     fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)

     plt.plot([0, 1], [0, 1], 'k--')
     plt.plot(fpr, tpr, label='SVM', color="r")
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('SVM ROC Curve', fontsize=16)
     plt.show()
```

ภาพประกอบ 63 โค้ดสำหรับ ROC Curve ของ Support Vector Machine

สรุปแล้ว โมเดล Support Vector Machine ในกรณีนี้ไม่เหมาะสมในการนำไปใช้ทำนายข้อมูลที่มีลักษณะเช่นนี้ เนื่องจากโมเดลไม่มีความสามารถในการจำแนกกลุ่มเป้าหมายได้อย่างถูกต้องและครบถ้วน

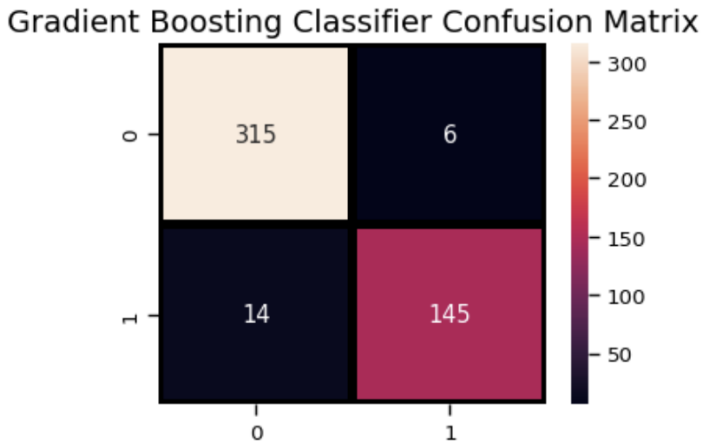
4.4 Gradient Boosting Classifier

จากผลการประเมินโมเดล Gradient Boosting Classifier พบว่าโมเดลมีความแม่นยำ (Accuracy) ที่สูงมากถึง 95.83% (ภาพประกอบ 64) ซึ่งถือว่าอยู่ในระดับสูงและใกล้เคียงกับโมเดล Decision Tree และ Random Forest ที่มีการทดสอบก่อนหน้า โดยโมเดลสามารถจำแนกตัวอย่างข้อมูลได้อย่างถูกต้องจำนวนมาก จากตัวอย่างที่ทดสอบทั้งหมด 480 รายการ

```
[85] gb = GradientBoostingClassifier()
     gb.fit(X_train, y_train)
     gb_pred = gb.predict(X_test)
     print("Gradient Boosting Classifier", accuracy_score(y_test, gb_pred))

     Gradient Boosting Classifier 0.9583333333333334
```

ภาพประกอบ 64 Accuracy ของ Gradient Boosting Classifier

เมื่อวิเคราะห์ผลจาก Confusion matrix (ภาพประกอบ 65 และ 66) จะเห็นว่าโมเดลมีความสามารถในการจำแนกกลุ่มลบ (Class 0) ได้ดีมาก มีการทำนายถูกต้องสูงถึง 315 ตัวอย่าง และมีการทำนายผิดพลาดเพียง 6 ตัวอย่างเท่านั้น ในขณะที่กลุ่มบวก (Class 1) ก็ถูกจำแนกได้ดีเช่นกัน โดยโมเดลสามารถทำนายถูกต้องจำนวน 145 ตัวอย่าง และมีการทำนายผิดพลาดจำนวน 14 ตัวอย่าง สะท้อนให้เห็นว่าประสิทธิภาพในการระบุผู้ที่เป็นกลุ่มเป้าหมายยังคงดีมาก แต่มีข้อผิดพลาดที่อาจจำเป็นต้องระมัดระวังในการใช้งานจริง



ภาพประกอบ 65 Confusion matrix ของ Gradient Boosting Classifier

```
[88] plt.figure(figsize=(4,3))
     sns.heatmap(confusion_matrix(y_test, gb_pred),
                 annot=True,fmt = "d",linecolor="k",linewidths=3)

     plt.title("Gradient Boosting Classifier Confusion Matrix",fontsize=14)
     plt.show()
```

ภาพประกอบ 66 Confusion matrix ของ Gradient Boosting Classifier

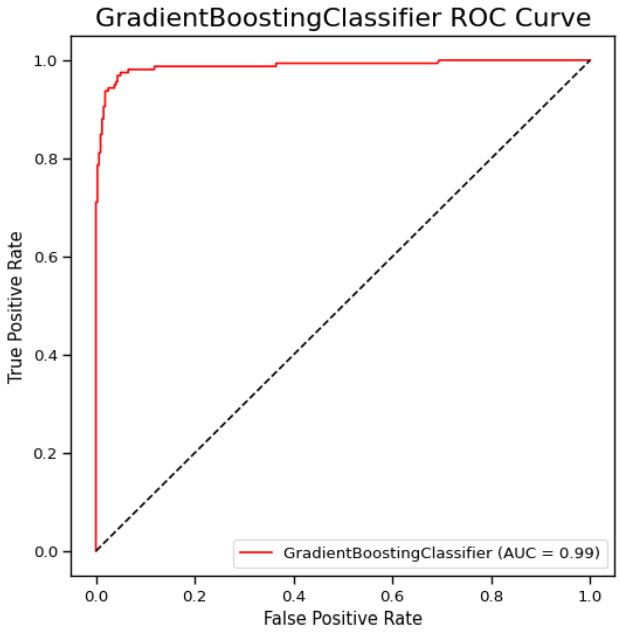
ค่า Precision และ Recall ของโมเดลนี้ (ภาพประกอบ 67) แสดงให้เห็นถึงความสามารถที่ดีในการจำแนกทั้งสองกลุ่ม โดยมี Precision และ Recall ของกลุ่มลบสูงถึงประมาณ 96% และ 98% ตามลำดับ ในขณะที่กลุ่มบวกมีค่า Precision 96% และ Recall 91% ชี้ให้เห็นว่าโมเดลมีความแม่นยำสูงในการระบุว่าตัวอย่างใดเป็นกลกลุ่มบวก แต่ยังคงมีโอกาสที่โมเดลจะพลาดในการระบุบางกรณีของกลุ่มบวกอยู่บ้างเล็กน้อย

```
[86] print(classification_report(y_test, gb_pred))
```

	precision	recall	f1-score	support
0	0.96	0.98	0.97	321
1	0.96	0.91	0.94	159
accuracy			0.96	480
macro avg	0.96	0.95	0.95	480
weighted avg	0.96	0.96	0.96	480

ภาพประกอบ 67 Precision, Recall และ F1-score ของ Gradient Boosting Classifier

สำหรับ ROC Curve ของโมเดล Gradient Boosting Classifier มีค่า AUC สูงถึง 0.99 (ภาพประกอบ 68 และ 69) แสดงว่าโมเดลนี้มีประสิทธิภาพในการจำแนกกลุ่มข้อมูลระหว่างกลุ่มบวกและกลุ่มลบได้อย่างชัดเจนและแม่นยำมาก ซึ่งหมายความว่าโมเดลนี้มีความสามารถสูงในการแยกแยะความแตกต่างของข้อมูลทั้งสองกลุ่มได้ดีเยี่ยม



ภาพประกอบ 68 ROC Curve ของ Gradient Boosting Classifier

```
[88] from sklearn.metrics import roc_curve, auc
     import matplotlib.pyplot as plt

     y_pred_prob = gb.predict_proba(X_test)[:, 1]

     fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)

     roc_auc = auc(fpr, tpr)

     plt.figure(figsize=(6, 6))
     plt.plot(fpr, tpr, color='r', label=f'GradientBoostingClassifier (AUC = {roc_auc:.2f})')
     plt.plot([0, 1], [0, 1], 'k--')
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('GradientBoostingClassifier ROC Curve', fontsize=16)
     plt.legend(loc='lower right')
     plt.show()
```

ภาพประกอบ 69 โค้ดสำหรับ ROC Curve ของ Gradient Boosting Classifier

โดยสรุป โมเดล Gradient Boosting Classifier มีประสิทธิภาพในการจำแนกข้อมูลสูงมาก และเหมาะสมอย่างยิ่งที่จะนำไปใช้ในการวิเคราะห์หรือการคาดการณ์จริง โดยเฉพาะอย่างยิ่งในบริบทที่ต้องการลดความผิดพลาดและเพิ่มความแม่นยำในการทำนาย แต่หากต้องการนำไปใช้ในสถานการณ์ที่มีผลกระทบรุนแรงจากการพลาดกลุ่มบวก ควรพิจารณาปรับเกณฑ์การตัดสินใจหรือการปรับเทียบพารามิเตอร์เพิ่มเติมเพื่อเพิ่มประสิทธิภาพในการจำแนกกลุ่มเป้าหมายให้ครอบคลุมมากขึ้น

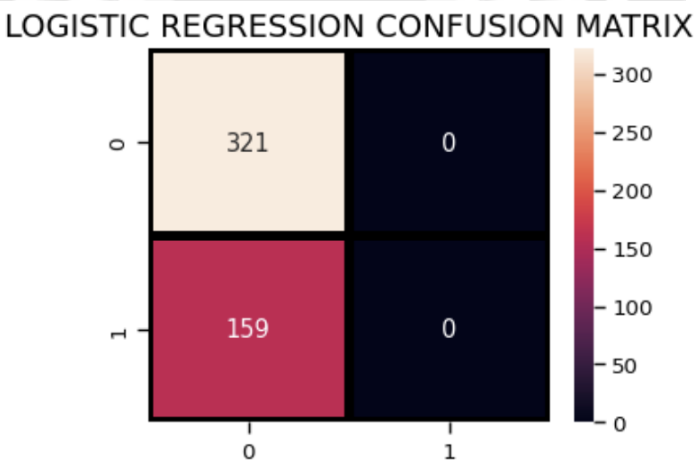
4.5 Logistic Regression

จากผลการประเมินโมเดล Logistic Regression พบว่า โมเดลมีค่าความแม่นยำ (Accuracy) อยู่ที่ 66.88% (ภาพประกอบ 70) ซึ่งถือเป็นค่าที่ค่อนข้างต่ำ และแสดงให้เห็นว่าโมเดลมีปัญหาในการจำแนกกลุ่มข้อมูลค่อนข้างมาก

```
[73] lr_model = LogisticRegression()
     lr_model.fit(X_train,y_train)
     accuracy_lr = lr_model.score(X_test,y_test)
     print("Logistic Regression accuracy is :",accuracy_lr)

     Logistic Regression accuracy is : 0.66875
```

ภาพประกอบ 70 Accuracy ของ Logistic Regression

เมื่อตรวจสอบ Confusion matrix อย่างละเอียด (ภาพประกอบ 71 และ 72)จะพบว่าโมเดลทำนายทุกตัวอย่างข้อมูลให้เป็นกลุ่มลบ (Class 0) เท่านั้น โดยทำนายกลุ่มลบถูกต้องทั้งหมด 321 ตัวอย่าง แต่กลับไม่สามารถระบุกลุ่มบวก (Class 1) ได้ ส่งผลให้เกิด False Negative สูงถึง 159 ตัวอย่าง หมายความว่าโมเดล Logistic Regression นี้ไม่มีประสิทธิภาพในการแยกแยะกลุ่มบวกออกจากกลุ่มลบ



ภาพประกอบ 71 Confusion matrix ของ Logistic Regression

```
[75] plt.figure(figsize=(4,3))
     sns.heatmap(confusion_matrix(y_test, lr_pred),
                 annot=True,fmt = "d",linecolor="k",linewidths=3)

     plt.title("LOGISTIC REGRESSION CONFUSION MATRIX",fontsize=14)
     plt.show()
```

ภาพประกอบ 72 โค้ดสำหรับ Confusion matrix ของ Logistic Regression

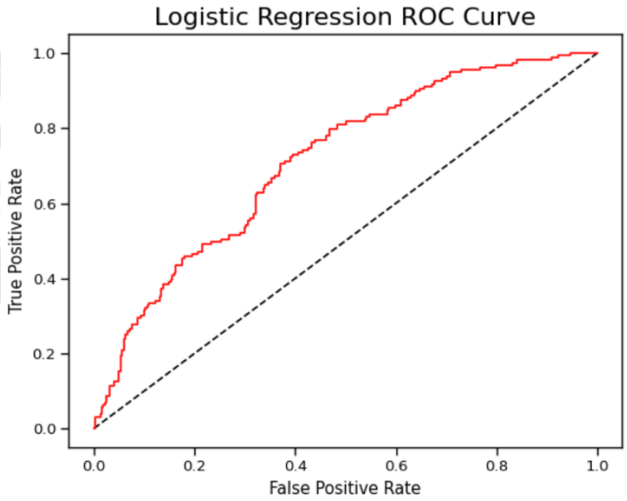
จากค่า Precision และ Recall (ภาพประกอบ 73) สำหรับกลุ่มบวกจะพบว่ามีค่าเป็นศูนย์ทั้งคู่ เนื่องจากโมเดลไม่สามารถระบุหรือทำนายตัวอย่างที่เป็นกลุ่มบวกได้เลย ในขณะที่กลุ่มลบมีค่า Precision เท่ากับ 67% และ Recall เท่ากับ 100% เนื่องจากโมเดลทำนายกลุ่มลบทั้งหมดโดยไม่มีการคำนึงถึงความเป็นไปได้ในการเป็นกลุ่มบวก

```
[74] lr_pred= lr_model.predict(X_test)
     report = classification_report(y_test,lr_pred)
     print(report)
```

```
              precision    recall  f1-score   support

           0       0.67      1.00      0.80       321
           1       0.00      0.00      0.00       159

    accuracy                           0.67       480
   macro avg       0.33      0.50      0.40       480
weighted avg       0.45      0.67      0.54       480
```

ภาพประกอบ 73 Precision, Recall และ F1-score ของ Logistic Regression

นอกจากนี้ กราฟ ROC Curve (ภาพประกอบ 74 และ 75) ยังแสดงถึงประสิทธิภาพที่ต่ำของโมเดล โดยมีพื้นที่ใต้เส้นโค้ง (AUC) ไม่สูง แสดงให้เห็นว่าโมเดลนี้ไม่สามารถแยกแยะข้อมูลสองกลุ่มได้ดีพอที่จะนำไปใช้ในทางปฏิบัติจริงได้อย่างมีประสิทธิภาพ



ภาพประกอบ 74 ROC Curve ของ Logistic Regression

```
[76] y_pred_prob = lr_model.predict_proba(X_test)[:,1]
     fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
     plt.plot([0, 1], [0, 1], 'k--' )
     plt.plot(fpr, tpr, label='Logistic Regression',color = "r")
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('Logistic Regression ROC Curve',fontsize=16)
     plt.show();
```

ภาพประกอบ 75 โค้ดสำหรับ ROC Curve ของ Logistic Regression

โดยสรุปแล้ว โมเดล Logistic Regression นี้ยังขาดประสิทธิภาพอย่างชัดเจนในการจำแนกกลุ่มข้อมูล และไม่เหมาะสมที่จะนำไปใช้ในบริบทที่จำเป็นต้องระบุหรือคาดการณ์กลุ่มบวกอย่างถูกต้อง

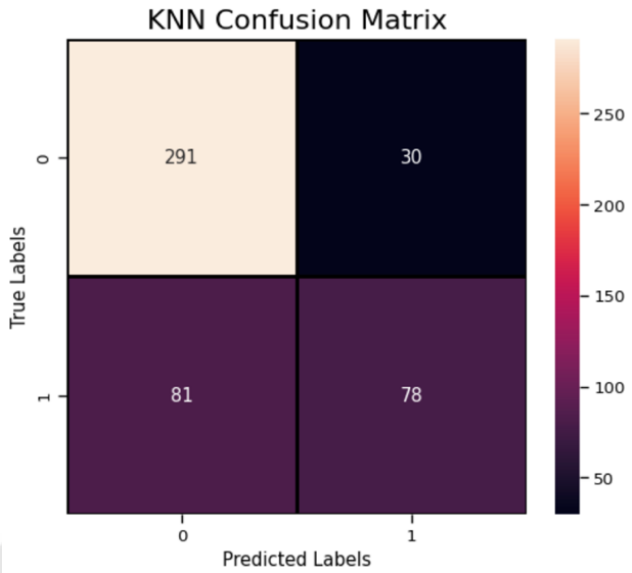
4.6 K-Nearest Neighbor

จากผลการประเมินโมเดล K-Nearest Neighbor (ภาพประกอบ 76) พบว่าโมเดลนี้มีค่าความแม่นยำ (Accuracy) ประมาณ 66.04% ซึ่งแสดงให้เห็นว่าประสิทธิภาพโดยรวมของโมเดลอยู่ในระดับค่อนข้างต่ำ โดยมีความแม่นยำต่ำกว่าเมื่อเปรียบเทียบกับโมเดลที่มีประสิทธิภาพสูงอย่าง Decision Tree, Random Forest Boost

```
[61] from sklearn.impute import SimpleImputer
     from sklearn.neighbors import KNeighborsClassifier

     imputer = SimpleImputer(strategy='mean')

     X_train_imputed = imputer.fit_transform(X_train)
     X_test_imputed = imputer.transform(X_test)

     knn_model = KNeighborsClassifier(n_neighbors=11)
     knn_model.fit(X_train_imputed, y_train)
     predicted_y = knn_model.predict(X_test_imputed)
     accuracy_knn = knn_model.score(X_test_imputed, y_test)
     print("KNN accuracy:", accuracy_knn)

     KNN accuracy: 0.6604166666666667
```

ภาพประกอบ 76 Accuracy ของ K-Nearest Neighbor

เมื่อพิจารณา Confusion matrix (ภาพประกอบ 77 และ 78) จะพบว่าโมเดลสามารถจำแนกกลุ่มลบ (Class 0) ได้ค่อนข้างดี โดยทำนายถูกต้องถึง 291 ตัวอย่าง แต่มีการทำนายผิดพลาดจำนวน 30 ตัวอย่าง ในส่วนของกลุ่มบวก (Class 1) โมเดลสามารถทำนายถูกต้องได้เพียง 78 ตัวอย่างจากทั้งหมด 159 ตัวอย่าง ซึ่งถือเป็นจำนวนที่ค่อนข้างต่ำ โดยมีตัวอย่างที่ทำนายผิดพลาดสูงถึง 81 ตัวอย่าง ทำให้เห็นว่าโมเดลยังมีจุดอ่อนที่สำคัญในการจำแนกกลุ่มบวกอย่างชัดเจน



ภาพประกอบ 77 Confusion matrix ของ K-Nearest Neighbor

```
[64] from sklearn.metrics import confusion_matrix
     import seaborn as sns

     cm = confusion_matrix(y_test, predicted_y)

     plt.figure(figsize=(6, 5))
     sns.heatmap(cm, annot=True, fmt='d', linecolor='k', linewidths=2)

     plt.title('KNN Confusion Matrix', fontsize=16)
     plt.xlabel('Predicted Labels')
     plt.ylabel('True Labels')
     plt.show()
```

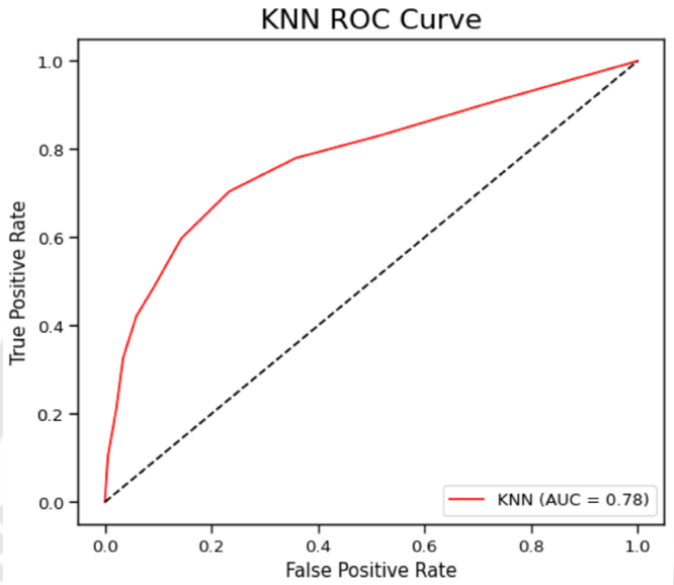
ภาพประกอบ 78 โค้ดสำหรับ Confusion matrix ของ K-Nearest Neighbor

จากภาพประกอบ 79 ค่า Precision และ Recall ช่วยสะท้อนปัญหาดังกล่าวได้ชัดขึ้น โดยกลุ่มลบมีค่า Precision และ Recall อยู่ที่ 71% และ 84% ตามลำดับ แสดงถึงความแม่นยำและการครอบคลุมที่ค่อนข้างดีในการจำแนกกลุ่มลบ แต่สำหรับกลุ่มบวกมีค่า Precision เพียง 48% และ Recall ต่ำมากที่ 30% เท่านั้น ซึ่งแสดงว่าโมเดลมีข้อผิดพลาดสูงมากในการจำแนกกลุ่มเป้าหมายที่แท้จริงได้ครบถ้วน

```
[62] print(classification_report(y_test, predicted_y))

                precision    recall  f1-score   support

             0       0.71      0.84      0.77       321
             1       0.48      0.30      0.37       159

      accuracy                           0.66       480
     macro avg       0.59      0.57      0.57       480
  weighted avg       0.63      0.66      0.64       480
```

ภาพประกอบ 79 Precision, Recall และ F1-score ของ K-Nearest Neighbor

จากภาพประกอบ 80 และ 81 ROC Curve ของโมเดล KNN มีค่า AUC ประมาณ 0.78 (ภาพประกอบ 80) ซึ่งถือว่าปานกลาง สะท้อนให้เห็นว่าโมเดลยังมีข้อจำกัดในการแยกแยะความแตกต่างระหว่างกลุ่มข้อมูลทั้งสองกลุ่มได้อย่างสมบูรณ์แบบ แต่ก็ยังถือว่าดีกว่าโมเดล Logistic Regression และ SVM ที่มีค่า AUC ต่ำกว่าอย่างชัดเจน



ภาพประกอบ 80 ROC Curve ของ K-Nearest Neighbor

```
[63] from sklearn.metrics import roc_curve, auc
     import matplotlib.pyplot as plt

     y_pred_prob = knn_model.predict_proba(X_test_imputed)[:, 1]
     fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
     roc_auc = auc(fpr, tpr)

     plt.figure(figsize=(6, 5))
     plt.plot(fpr, tpr, color='r', label=f'KNN (AUC = {roc_auc:.2f})')
     plt.plot([0, 1], [0, 1], 'k--')
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('KNN ROC Curve', fontsize=16)
     plt.legend(loc='lower right')
     plt.show()
```

ภาพประกอบ 81 โค้ดสำหรับ ROC Curve ของ K-Nearest Neighbors

โดยสรุปแล้ว โมเดล KNN ยังไม่เหมาะสมสำหรับการใช้งานที่ต้องการการจำแนกข้อมูลที่มีประสิทธิภาพสูง โดยเฉพาะในกรณีที่ต้องการลดการพลาดในการจำแนกกลุ่มบวก ซึ่งอาจส่งผลเสียอย่างมากในการนำไปใช้จริง หากจำเป็นต้องใช้โมเดลนี้ ควรพิจารณาการปรับแต่งค่า K หรือใช้เทคนิคการปรับข้อมูลให้เหมาะสมเพิ่มเติม หรือเลือกใช้โมเดลอื่นที่มีประสิทธิภาพในการแยกแยะข้อมูลที่ดีกว่า เช่น Random Forest หรือ Gradient Boosting Classifier

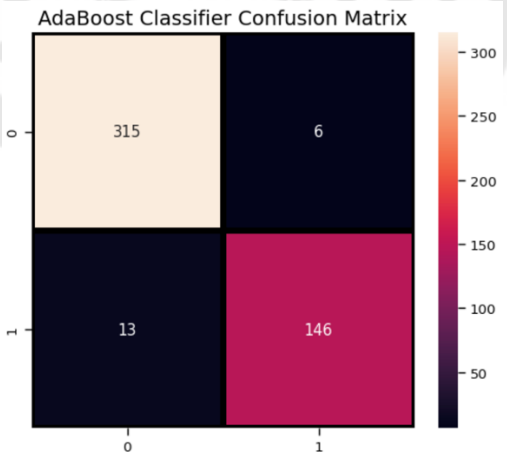
4.7 AdaBoost Classifier

จากผลการประเมินแบบจำลอง AdaBoost Classifier พบว่าโมเดลมีค่าความแม่นยำ (Accuracy) ที่สูงมากอยู่ที่ 96.04% (ภาพประกอบ 82) ซึ่งถือว่าเป็นผลลัพธ์ที่ดีมากและสามารถจำแนกข้อมูลได้อย่างถูกต้องแม่นยำ

```
[81] a_model = AdaBoostClassifier()
     a_model.fit(X_train,y_train)
     a_preds = a_model.predict(X_test)
     print("AdaBoost Classifier accuracy")
     metrics.accuracy_score(y_test, a_preds)

     AdaBoost Classifier accuracy
     0.9604166666666667
```

ภาพประกอบ 82 Accuracy ของ AdaBoost Classifier

เมื่อพิจารณา Confusion matrix (ภาพประกอบ 83 และ 84) จะพบว่า โมเดลสามารถจำแนกกลุ่มตัวอย่างที่เป็นลบ (Class 0) ได้อย่างมีประสิทธิภาพ โดยจำแนกได้ถูกต้องถึง 315 ตัวอย่าง และมีข้อผิดพลาดในการจำแนกกลุ่มลบผิดไปเพียง 6 ตัวอย่างเท่านั้น ในขณะที่กลุ่มตัวอย่างบวก (Class 1) โมเดลก็ยังสามารถทำงานได้อย่างดี โดยมีการจำแนกที่ถูกต้องอยู่ที่ 146 ตัวอย่าง และมีข้อผิดพลาดที่โมเดลจำแนกผิดไปเพียง 13 ตัวอย่าง แสดงให้เห็นว่าโมเดลมีประสิทธิภาพที่ดีในการระบุกลุ่มเป้าหมาย แต่อาจจะมีการพลาดกลุ่มเป้าหมายบางส่วนซึ่งต้องระมัดระวังในการใช้งานจริง



ภาพประกอบ 83 Confusion matrix ของ AdaBoost Classifier

```
[84] plt.figure(figsize=(6,5))
     sns.heatmap(confusion_matrix(y_test, a_preds),
                 annot=True,fmt = "d",linecolor="k",linewidths=3)

     plt.title("AdaBoost Classifier Confusion Matrix",fontsize=14)
     plt.show()
```

ภาพประกอบ 84 โค้ดสำหรับ Confusion matrix ของ AdaBoost Classifier

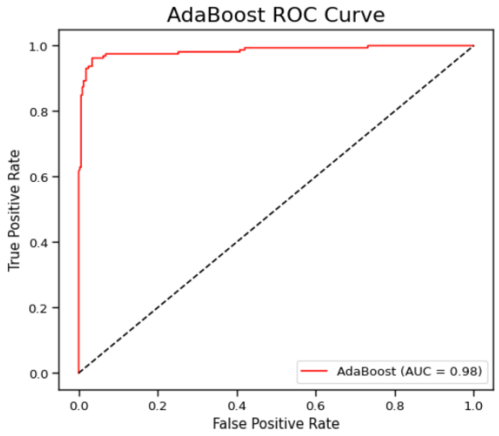
เมื่อพิจารณาค่า Precision และ Recall ของกลุ่มลบ พบว่ามีค่าอยู่ที่ประมาณ 96% และ 98% ตามลำดับ (ภาพประกอบ 85) สะท้อนว่าโมเดลมีความสามารถสูงในการระบุกลุ่มลบได้ถูกต้องและครอบคลุม ส่วนกลุ่มบวกมีค่า Precision และ Recall อยู่ที่ประมาณ 96% และ 91% ตามลำดับ แสดงให้เห็นว่าโมเดลมีความแม่นยำสูงแต่ยังมีจุดอ่อนเล็กน้อยในการตรวจจับกลุ่มเป้าหมายให้ครบถ้วน

```
[86] print(classification_report(y_test, gb_pred))
```

	precision	recall	f1-score	support
0	0.96	0.98	0.97	321
1	0.96	0.91	0.94	159
accuracy			0.96	480
macro avg	0.96	0.95	0.95	480
weighted avg	0.96	0.96	0.96	480

ภาพประกอบ 85 Precision, Recall และ F1-score ของ AdaBoost Classifier

ในส่วนของ ROC Curve (ภาพประกอบ 86 และ 87) พบว่าพื้นที่ใต้กราฟ (AUC) ของ AdaBoost มีค่าสูงมากที่ 0.98 แสดงให้เห็นว่าโมเดลนี้มีประสิทธิภาพยอดเยี่ยมในการแยกแยะระหว่างกลุ่มข้อมูลที่เป็นบวกและลบได้อย่างแม่นยำมาก โดยมีอัตราการเกิด False Positive ต่ำเมื่อเปรียบเทียบกับ True Positive ที่ได้รับจากโมเดล



ภาพประกอบ 86 ROC Curve ของ AdaBoost Classifier

```
[83] from sklearn.metrics import roc_curve, auc
     import matplotlib.pyplot as plt

     y_pred_prob = a_model.predict_proba(X_test)[:, 1]
     fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
     roc_auc = auc(fpr, tpr)

     plt.figure(figsize=(6, 5))
     plt.plot(fpr, tpr, color='r', label=f'AdaBoost (AUC = {roc_auc:.2f})')
     plt.plot([0, 1], [0, 1], 'k--')
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('AdaBoost ROC Curve', fontsize=16)
     plt.legend(loc='lower right')
     plt.show()
```

ภาพประกอบ 87 โค้ดสำหรับ ROC Curve ของ AdaBoost Classifier

สรุปโดยรวมได้ว่า AdaBoost Classifier เป็นโมเดลที่มีประสิทธิภาพสูงมากในการจำแนกข้อมูลชุดนี้ สามารถนำไปใช้งานจริงได้ดีมาก แต่มีความอ่อนไหวต่อการพลาดการตรวจจับกลุ่มบวก

4.8 Ensemble Voting Classifier

ในการทำ Ensemble Voting Classifier ได้มีการเลือกโมเดลมาทำทั้งหมด 3 โมเดล ได้แก่ AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier ซึ่งแบ่งหัวข้อดังต่อไปนี้

1. Ensemble Voting Classifier ที่ได้จากโมเดลที่ยังไม่ได้ปรับเทียบ
2. ผลจากการปรับเทียบทั้ง 3 โมเดล (AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier)
3. Ensemble Voting Classifier ที่ได้จากโมเดลที่ปรับเทียบแล้ว

เหตุผลหลักที่เลือกโมเดลทั้งสามนี้เนื่องจากโมเดลเหล่านี้มีประสิทธิภาพ(ก่อนปรับเทียบ)ในการจำแนกข้อมูลสูง โดยสังเกตจากค่าความแม่นยำ (Accuracy) ที่สูงกว่า 95% และค่า AUC จาก ROC Curve ที่อยู่ในระดับสูงใกล้เคียงกัน ซึ่งแสดงให้เห็นถึงความสามารถในการแยกแยะกลุ่มตัวอย่างได้อย่างชัดเจนและแม่นยำ อีกทั้งยังมีความสมดุลในการทำนายกลุ่มบวกและลบเป็นอย่างดี โดยเฉพาะ AdaBoost และ Gradient Boosting ที่มีประสิทธิภาพในการจำแนกข้อมูลกลุ่มบวก (Class 1) และกลุ่มลบ (Class 0) ได้ค่อนข้างดีเยี่ยม

4.8.1 Ensemble Voting Classifier ที่ได้จากโมเดลที่ยังไม่ได้ปรับเทียบ

จากผลลัพธ์การประเมินโมเดล Ensemble Voting Classifier ที่รวมโมเดล AdaBoost, Decision Tree และ Gradient Boosting Classifier เข้าด้วยกัน จากภาพกระกอบ 88

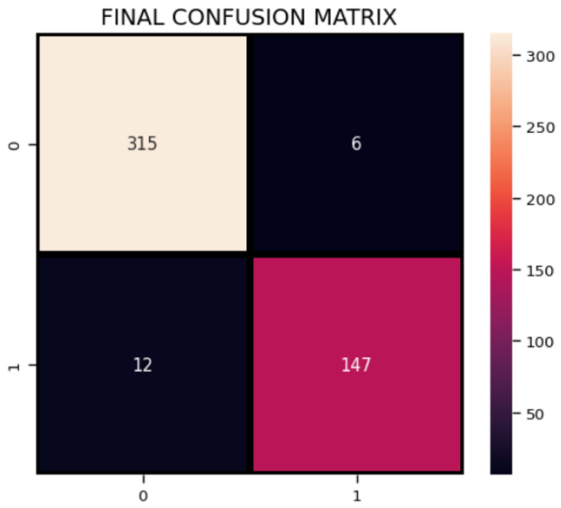
พบว่าโมเดลมีค่าความแม่นยำโดยรวม (Accuracy) อยู่ที่ประมาณ 96.04% ซึ่งเป็นค่าที่สูงกว่าโมเดลเดี่ยวทั้งสามโมเดลที่นำมาใช้ในการสร้าง Ensemble เล็กน้อย

ค่า Precision และ Recall ของกลุ่มลบอยู่ในระดับที่ดีมากคือ 96% และ 98% ตามลำดับ ในขณะที่กลุ่มบวกมีค่า Precision อยู่ที่ 96% และ Recall 92% สะท้อนถึงความแม่นยำและความครอบคลุมที่สูงมากในการจำแนกทั้งสองกลุ่ม โดยเฉพาะอย่างยิ่งการจำแนกกลุ่มบวกที่มีความสำคัญในงานหลายประเภทนั้นมีความแม่นยำสูงขึ้นอย่างเห็นได้ชัดเมื่อเทียบกับบางโมเดลเดี่ยว

```
[93] from sklearn.ensemble import VotingClassifier
     clf1 = GradientBoostingClassifier()
     clf2 = DecisionTreeClassifier()
     clf3 = AdaBoostClassifier()
     eclf1 = VotingClassifier(estimators=[('gbc', clf1), ('lr', clf2), ('abc', clf3)], voting='soft')
     eclf1.fit(X_train, y_train)
     predictions = eclf1.predict(X_test)
     print("Final Accuracy Score ")
     print(accuracy_score(y_test, predictions))
     print(classification_report(y_test, predictions))

     Final Accuracy Score
     0.9604166666666667
                   precision    recall  f1-score   support

                0       0.96      0.98      0.97       321
                1       0.96      0.92      0.94       159

         accuracy                           0.96       480
        macro avg       0.96      0.95      0.95       480
     weighted avg       0.96      0.96      0.96       480
```

ภาพประกอบ 88 Accuracy และ Precision, Recall และ F1-score ของ Ensemble Voting Classifier (ก่อนปรับเทียบ)

เมื่อพิจารณา Confusion Matrix อย่างละเอียด (ภาพประกอบ 89 และ 90) พบว่า Ensemble model สามารถจำแนกกลุ่มลบ (Class 0) ได้ถูกต้องถึง 315 ตัวอย่าง และจำแนกผิดพลาดเพียง 6 ตัวอย่าง ในขณะเดียวกันสำหรับกลุ่มบวก (Class 1) โมเดลนี้สามารถจำแนกได้ถูกต้องมากถึง 147 ตัวอย่าง โดยมีตัวอย่างที่จำแนกผิดไปเป็นจำนวนเพียง 12 ตัวอย่าง ซึ่งหมายความว่าโมเดล Ensemble นี้มีความสามารถในการตรวจจับกลุ่มเป้าหมาย (Class 1) ได้ดีกว่าโมเดลเดี่ยวบางตัวอย่างชัดเจน

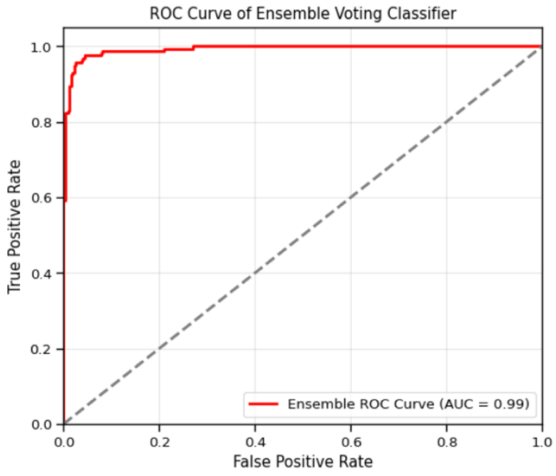


ภาพประกอบ 89 Confusion Matrix ของ Ensemble Voting Classifier (ก่อนปรับเทียบ)

```
[91] plt.figure(figsize=(6,5))
     sns.heatmap(confusion_matrix(y_test, predictions),
                 annot=True,fmt = "d",linecolor="k",linewidths=3)

     plt.title("FINAL CONFUSION MATRIX",fontsize=14)
     plt.show()
```

ภาพประกอบ 90 โค้ดสำหรับ Confusion Matrix ของ Ensemble Voting Classifier (ก่อนปรับเทียบ)

จากกราฟภาพประกอบ 91 และ 92 ROC Curve พบว่าค่าพื้นที่ใต้กราฟ (AUC) อยู่ที่ประมาณ 0.99 ซึ่งเป็นตัวบ่งชี้ที่ชัดเจนถึงประสิทธิภาพที่ยอดเยี่ยมของโมเดลนี้ในการแยกแยะระหว่างกลุ่มที่เป็นบวกและลบ โดยพื้นที่ใต้เส้น ROC นี้สูงกว่าโมเดลเดี่ยวหรือเทียบเท่ากับโมเดลที่ดีที่สุดในกลุ่มที่เลือกมา



ภาพประกอบ 91 ROC Curve ของ Ensemble Voting Classifier (ก่อนปรับเทียบ)

```
[92] from sklearn.metrics import roc_curve, auc
     import matplotlib.pyplot as plt

     pred_proba = eclf1.predict_proba(X_test)[:,1]

     fpr, tpr, thresholds = roc_curve(y_test, pred_proba)
     roc_auc = auc(fpr, tpr)

     plt.figure(figsize=(6,5))
     plt.plot(fpr, tpr, color='r', lw=2, label='Ensemble ROC Curve (AUC = %0.2f)' % roc_auc)
     plt.plot([0, 1], [0, 1], color='gray', lw=2, linestyle='--')
     plt.xlim([0.0, 1.0])
     plt.ylim([0.0, 1.05])
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('ROC Curve of Ensemble Voting Classifier')
     plt.legend(loc="lower right")
     plt.grid(alpha=0.3)
     plt.show()
```

ภาพประกอบ 92 โค้ดสำหรับ ROC Curve ของ Ensemble Voting Classifier (ก่อนปรับเทียบ)

4.8.2 ผลจากการปรับเทียบทั้ง 3 โมเดล (AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier)

4.8.2.1 AdaBoost Classifier

จากการปรับเทียบโมเดล AdaBoost Classifier ได้ผลลัพธ์ตามตาราง 13 และภาพประกอบ 93 ค่าAccuracy โดยรวมลดลงเล็กน้อย (จาก 96.04% เป็น 95.42%) หลังการปรับเทียบอาจเหมือนโมเดลแย่ลง แต่ Recallของคลาส 1 เพิ่มขึ้น (จาก 91% เป็น 93%) ซึ่งถือว่าสำคัญมากในบริบทที่ต้องการจับกลุ่มเสี่ยงหรือกลุ่มเป้าหมายเช่น ลูกค้าที่อาจจะยกเลิกบริการ Precision ของคลาส 1 ลดลงเล็กน้อย (จาก 96% เป็น 93%) หมายความว่ามี false positive เพิ่มขึ้น และค่า Cross-Validation สูงมาก (96.78%) แสดงว่าโมเดลหลังปรับเทียบ มีแนวโน้มจะ generalize ได้ดีเมื่อเจอกับข้อมูลใหม่ โมเดลหลังปรับเทียบ มีโครงสร้างที่ลึกขึ้น (max_depth=3) และเรียนรู้ช้าลง (learning_rate=0.1) ซึ่งช่วยลดความเสี่ยงของ overfitting จากโมเดลต้นแบบที่ใช้ค่า default ดังนั้นจึงสามารถสรุปได้ว่าการปรับเทียบแบบจำลองในครั้งนี้มีความเหมาะสมและควรนำไปใช้ในกระบวนการพัฒนาโมเดลขั้นสุดท้ายต่อไป

ตาราง 13 เปรียบเทียบผลลัพธ์โมเดล AdaBoost Classifier ก่อนและหลังปรับเทียบ

Metric	ก่อนปรับเทียบ	หลังปรับเทียบ
Accuracy	0.9604	0.9542
Precision (Class 0)	0.96	0.97
Recall (Class 0)	0.98	0.97
F1-score (Class 0)	0.97	0.97
Precision (Class 1)	0.96	0.93
Recall (Class 1)	0.92	0.93
F1-score (Class 1)	0.94	0.93
Macro Avg F1-score	0.95	0.95
Cross-Validation Score	–	0.9678
Best Params	Default (max_depth=1, learning_rate=1)	max_depth=3, learning_rate=0.1, n=50

```
[84] #fine tune AdaBoost
     from sklearn.ensemble import AdaBoostClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.model_selection import GridSearchCV
     from sklearn.metrics import accuracy_score, classification_report

     param_grid = {
         'n_estimators': [50, 100, 200],
         'learning_rate': [0.01, 0.1, 1.0],
         'estimator': [DecisionTreeClassifier(max_depth=1),
                       DecisionTreeClassifier(max_depth=3)]
     }

     ada = AdaBoostClassifier(random_state=42)

     grid_search = GridSearchCV(estimator=ada,
                                param_grid=param_grid,
                                cv=5,
                                scoring='accuracy',
                                n_jobs=-1,
                                verbose=1)

     grid_search.fit(X_train, y_train)

     print("Best parameters found:", grid_search.best_params_)
     print("Best cross-validation score:", grid_search.best_score_)

     best_ada = grid_search.best_estimator_
     a_preds = best_ada.predict(X_test)

     print("Optimized AdaBoost Classifier accuracy:", accuracy_score(y_test, a_preds))
     print(classification_report(y_test, a_preds))

Fitting 5 folds for each of 18 candidates, totalling 90 fits
Best parameters found: {'estimator': DecisionTreeClassifier(max_depth=3), 'learning_rate': 0.1, 'n_estimators': 50}
Best cross-validation score: 0.9677730621396542
Optimized AdaBoost Classifier accuracy: 0.9541666666666667
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       321
           1       0.93      0.93      0.93       159

    accuracy                           0.95       480
   macro avg       0.95      0.95      0.95       480
weighted avg       0.95      0.95      0.95       480
```

ภาพประกอบ 93 โค้ดการปรับเทียบโมเดล AdaBoost Classifier

4.8.2.2 Decision Tree

จากการปรับเทียบโมเดล Decision Tree ได้ผลลัพธ์ตามตาราง 14 และภาพประกอบ 93 ค่า Accuracy เพิ่มขึ้นเล็กน้อย (จาก 95.63% เป็น 95.83%) หลังการปรับเทียบ แม้จะต่างกันเล็กน้อย แต่ยังสะท้อนถึงการพัฒนาโมเดลที่ดีขึ้น F1-score ของคลาส 1 (กลุ่มบวก) เพิ่มขึ้นจาก 93% เป็น 94% ค่า Cross-validation สูงขึ้นมาก (จากไม่กำหนด เป็น 97.67%) แสดงถึงความสามารถในการทำงานได้ดีกับข้อมูลใหม่ (generalization) โมเดลหลังปรับเทียบสามารถควบคุมความซับซ้อนดีขึ้น ด้วยการกำหนด max_depth=5 และ min_samples_leaf=4 ซึ่งช่วยลด overfitting จากการปล่อยให้ต้นไม้เติบโตแบบไม่จำกัด

แม้ความแม่นยำโดยรวมจะเพิ่มขึ้นไม่มาก แต่โมเดลที่ผ่านการปรับเทียบแสดงให้เห็นถึง ประสิทธิภาพที่ดีขึ้น, ความเสถียรที่มากกว่า, และมีการควบคุมความซับซ้อนของโครงสร้างโมเดลอย่างเหมาะสม จึงสามารถสรุปได้ว่า ควรทำ Fine-Tune โมเดล เพื่อให้ได้แบบจำลองที่มีความสมดุลทั้งด้านประสิทธิภาพและความสามารถในการนำไปใช้งานจริงกับข้อมูลที่ไม่เคยเห็นมาก่อน

ตาราง 14 เปรียบเทียบผลลัพธ์โมเดล Decision Tree ก่อนและหลังปรับเทียบ

Metric	ก่อนปรับเทียบ	หลังปรับเทียบ
Accuracy	0.9563	0.9583
Precision (Class 0)	0.96	0.96
Recall (Class 0)	0.98	0.98
F1-score (Class 0)	0.97	0.97
Precision (Class 1)	0.95	0.95
Recall (Class 1)	0.92	0.92
F1-score (Class 1)	0.93	0.94
Macro Avg F1-score	0.95	0.95
Cross-Validation Score	–	0.9767
Best Params	default	criterion='gini',max_depth=5,min_samples_leaf=4, min_samples_split=2

```python
#fine tune decision tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report, accuracy_score

param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 3, 5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': [None, 'sqrt', 'log2']
}

dt = DecisionTreeClassifier(random_state=42)

grid_search = GridSearchCV(estimator=dt,
                           param_grid=param_grid,
                           cv=5,
                           n_jobs=-1,
                           scoring='accuracy',
                           verbose=1)

grid_search.fit(X_train, y_train)

print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score: ", grid_search.best_score_)

best_dt = grid_search.best_estimator_
y_pred = best_dt.predict(X_test)

print("Optimized Decision Tree accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Fitting 5 folds for each of 270 candidates, totalling 1350 fits
Best parameters found:  {'criterion': 'gini', 'max_depth': 5, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 2}
Best cross-validation score:  0.9767216527866752
Optimized Decision Tree accuracy: 0.9583333333333334
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       321
           1       0.95      0.92      0.94       159

    accuracy                           0.96       480
   macro avg       0.96      0.95      0.95       480
weighted avg       0.96      0.96      0.96       480
```

ภาพประกอบ 94 โค้ดการปรับเทียบโมเดล Decision Tree

4.8.2.3 Gradient Boosting Classifier

จากการปรับเทียบโมเดล Decision Tree ได้ผลลัพธ์ตามตาราง 15 และภาพประกอบ 95 ค่า Accuracy เพิ่มขึ้น จาก 95.83% เป็น 96.46% แสดงว่าโมเดลมีความแม่นยำโดยรวมมากขึ้นหลังการปรับเทียบ Recall และ Precision ของคลาส 1 (กลุ่มเป้าหมาย) เพิ่มขึ้นทั้งคู่ โดยเฉพาะ Precision ที่สูงขึ้นเป็น 97 ซึ่งแสดงให้เห็นว่าโมเดลสามารถลดการแจ้งเตือนผิดพลาด (false positives) ได้ดีขึ้น Recall ของคลาส 0 ก็เพิ่มขึ้นเช่นกัน (จาก 98% เป็น 99%) สะท้อนว่าโมเดลแทบไม่พลาดกลุ่มที่เป็นคลาส 0 ค่า cross-validation score สูงถึง 97.13% ซึ่งสะท้อนว่าโมเดลที่ได้จากการ Fine-Tune สามารถทำงานได้ดีทั้งกับข้อมูลฝึกและมีแนวโน้ม generalize ได้ดีกับข้อมูลใหม่

โมเดลหลังปรับเทียบดีกว่าอย่างชัดเจน ทั้งในแง่ของ accuracy, recall ของคลาสที่สำคัญ, และค่าเฉลี่ยแบบ cross-validation ที่สูงขึ้น จึงสามารถสรุปได้ว่า การปรับเทียบ Gradient Boosting Classifier เป็นกระบวนการที่มีความจำเป็นและให้ผลลัพธ์ที่มีประสิทธิภาพสูงขึ้นอย่างแท้จริง ทั้งในด้านความแม่นยำและความเสถียรของโมเดล

ตาราง 15 เปรียบเทียบผลลัพธ์โมเดล Gradient Boosting Classifier ก่อนและหลังปรับเทียบ

Metric	ก่อนปรับเทียบ	หลังปรับเทียบ
Accuracy	0.9583	0.9646
Precision (Class 0)	0.96	0.96
Recall (Class 0)	0.98	0.99
F1-score (Class 0)	0.97	0.97
Precision (Class 1)	0.96	0.97
Recall (Class 1)	0.91	0.92
F1-score (Class 1)	0.94	0.94
Macro Avg F1-score	0.95	0.96
Cross-Validation Score	–	0.9713
Best Params	Default	learning_rate=0.2, max_depth=3, min_samples_leaf=2, min_samples_split=2, n_estimators=200

```
[91] #fine tune Gradient Boosting Classifier
     from sklearn.ensemble import GradientBoostingClassifier
     from sklearn.model_selection import GridSearchCV
     from sklearn.metrics import accuracy_score, classification_report

     param_grid = {
         'n_estimators': [100, 200],
         'learning_rate': [0.01, 0.1, 0.2],
         'max_depth': [3, 5, 7],
         'min_samples_split': [2, 5],
         'min_samples_leaf': [1, 2]
     }

     gb = GradientBoostingClassifier(random_state=42)

     grid_search = GridSearchCV(estimator=gb,
                                param_grid=param_grid,
                                cv=5,
                                scoring='accuracy',
                                n_jobs=-1,
                                verbose=1)

     grid_search.fit(X_train, y_train)

     print("Best parameters found:", grid_search.best_params_)
     print("Best cross-validation score:", grid_search.best_score_)

     best_gb = grid_search.best_estimator_
     gb_pred = best_gb.predict(X_test)

     print("Optimized Gradient Boosting accuracy:", accuracy_score(y_test, gb_pred))
     print(classification_report(y_test, gb_pred))

⇥ Fitting 5 folds for each of 72 candidates, totalling 360 fits
  Best parameters found: {'learning_rate': 0.2, 'max_depth': 3, 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 200}
  Best cross-validation score: 0.9713444907110826
  Optimized Gradient Boosting accuracy: 0.9645833333333333
                precision    recall  f1-score   support

             0       0.96      0.99      0.97       321
             1       0.97      0.92      0.94       159

      accuracy                           0.96       480
     macro avg       0.97      0.95      0.96       480
  weighted avg       0.96      0.96      0.96       480
```

ภาพประกอบ 95 ผลลัพธ์จากการปรับเทียบโมเดล Gradient Boosting Classifier

4.8.3 Ensemble Voting Classifier ที่ได้จากโมเดลที่ปรับเทียบแล้ว

จากผลลัพธ์การประเมินโมเดล Ensemble Voting Classifier (Fine-Tuned Models) (ภาพประกอบ 96) เป็นการรวมโมเดลที่ผ่านการปรับเทียบแล้ว ได้แก่ AdaBoost Classifier, Decision Tree Classifier, และ Gradient Boosting Classifier พบว่าโมเดลมีค่าความแม่นยำโดยรวม (Accuracy) อยู่ที่ 96.46% ซึ่งเป็นค่าที่สูงและมีเสถียรภาพมากเมื่อเทียบกับโมเดลเดี่ยวแต่ละตัว โมเดลสามารถจำแนกกลุ่มลบ (Class 0) ได้ดีมาก โดยมีค่า Precision เท่ากับ 97% และ Recall เท่ากับ 98% ขณะที่การจำแนกกลุ่มบวก (Class 1) ซึ่งมักเป็นกลุ่มเป้าหมายที่ต้องการวิเคราะห์เชิงธุรกิจหรือการคัดกรองความเสี่ยง ก็แสดงผลลัพธ์ได้อย่างยอดเยี่ยมเช่นกัน โดยมีค่า Precision เท่ากับ 96% และ Recall เท่ากับ 94% ซึ่งสะท้อนถึงความแม่นยำและความครอบคลุมที่สูงในการจำแนกข้อมูลทั้งสองกลุ่ม

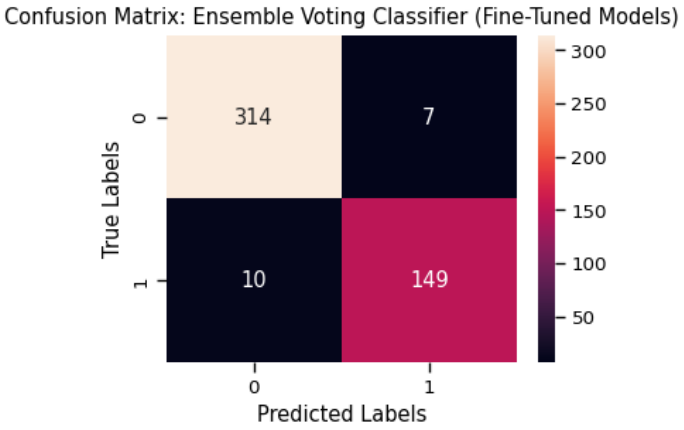
```
[ ] from sklearn.ensemble import VotingClassifier
    from sklearn.metrics import accuracy_score, classification_report

    ensemble_finetuned = VotingClassifier(
        estimators=[
            ('ada', best_ada),
            ('dt', best_dt),
            ('gb', best_gb)
        ],
        voting='soft'
    )

    ensemble_finetuned.fit(X_train, y_train)

    ensemble_preds = ensemble_finetuned.predict(X_test)

    print("Ensemble Voting Classifier (Fine-Tuned Models) Accuracy:")
    print(accuracy_score(y_test, ensemble_preds))
    print(classification_report(y_test, ensemble_preds))
```

```
Ensemble Voting Classifier (Fine-Tuned Models) Accuracy:
0.9645833333333333
              precision    recall  f1-score   support

           0       0.97      0.98      0.97       321
           1       0.96      0.94      0.95       159

    accuracy                           0.96       480
   macro avg       0.96      0.96      0.96       480
weighted avg       0.96      0.96      0.96       480
```

ภาพประกอบ 96 Accuracy และ Precision, Recall และ F1-score ของ Ensemble Voting Classifier (หลังปรับเทียบ)

จากการพิจารณา Confusion Matrix (ภาพประกอบ 97 และ 98) พบว่าโมเดลสามารถจำแนกตัวอย่างในกลุ่มลบได้ถูกต้องจำนวน 314 ตัวอย่าง จากทั้งหมด 321 ตัวอย่าง และ

มีการจำแนกผิดเพียง 7 ตัวอย่าง ในขณะที่กลุ่มบวกจำแนกถูกต้อง 149 ตัวอย่าง จากทั้งหมด 159 ตัวอย่าง และจำแนกผิดเพียง 10 ตัวอย่าง ถือเป็นอัตราความผิดพลาดที่ต่ำมาก

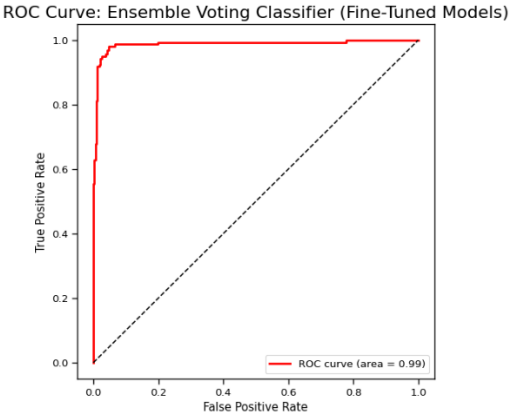


ภาพประกอบ 97 Confusion Matrix ของ Ensemble Voting Classifier (หลังปรับเทียบ)

```python
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

cm = confusion_matrix(y_test, ensemble_preds)

plt.figure(figsize=(4, 3))
sns.heatmap(cm, annot=True, fmt='d', xticklabels=[0, 1], yticklabels=[0, 1])
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix: Ensemble Voting Classifier (Fine-Tuned Models)')
plt.show()
```

ภาพประกอบ 98 โค้ดสำหรับ Confusion Matrix ของ Ensemble Voting Classifier (หลังปรับเทียบ)

จากการประเมินผลผ่าน ROC Curve(ภาพประกอบ 99 และ 100) โมเดลมีค่าพื้นที่ใต้กราฟ (AUC) เท่ากับ 0.99 ซึ่งแสดงให้เห็นว่าแบบจำลอง Ensemble นี้มีประสิทธิภาพสูงมากในการแยกแยะระหว่างกลุ่มบวกและกลุ่มลบ โดยค่า AUC ดังกล่าวอยู่ในระดับใกล้เคียงหรือสูงกว่าโมเดลเดี่ยวที่ดีที่สุดในชุดโมเดลที่นำมาเปรียบเทียบ

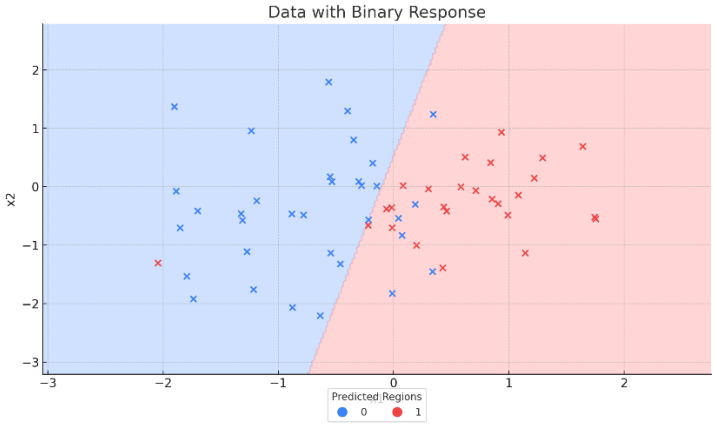


ภาพประกอบ 99 ROC Curve ของ Ensemble Voting Classifier (หลังปรับเทียบ)

```python
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
from sklearn.preprocessing import label_binarize

y_score = ensemble_finetuned.predict_proba(X_test)[:, 1]

fpr, tpr, thresholds = roc_curve(y_test, y_score)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(6, 6))
plt.plot(fpr, tpr, color='r', lw=2, label='ROC curve (area = {:.2f})'.format(roc_auc))
plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve: Ensemble Voting Classifier (Fine-Tuned Models)', fontsize=16)
plt.legend(loc="lower right")
plt.show()
```

ภาพประกอบ 100 โค้ดสำหรับ ROC Curve ของ Ensemble Voting Classifier (หลังปรับเทียบ)

จากภาพประกอบ 101 และ 102 เป็นการพิจารณาผลการจำแนกกลุ่มข้อมูลทดสอบด้วยการลดมิติของข้อมูลด้วยเทคนิค Principal Component Analysis (PCA) เหลือเพียงสองมิติ เพื่อให้สามารถแสดงผลในรูปแบบภาพได้อย่างชัดเจน พื้นหลังสีฟ้าแสดงถึงบริเวณที่โมเดลทำนายว่าเป็นคลาส 0 ขณะที่พื้นหลังสีแดงแสดงบริเวณที่โมเดลทำนายว่าเป็นคลาส 1 จุดข้อมูลแต่ละจุดแทนตัวอย่างลูกค้าในชุดทดสอบ โดยใช้สีของจุดเพื่อระบุคลาสจริง (น้ำเงินสำหรับคลาส 0 และแดงสำหรับคลาส 1) และใช้สัญลักษณ์ × แสดงตำแหน่งของแต่ละตัวอย่าง ขอบเขตการจำแนกในภาพปรากฏเป็นเส้นตรงในเชิงแนวโน้ม ซึ่งแสดงให้เห็นว่าโมเดลสามารถแบ่งแยกกลุ่มได้ในระดับที่ชัดเจน โดยมีจุดที่โมเดลทำนายผิดพลาดเพียงไม่กี่จุดที่ปรากฏเป็นจุดสีตัดกับพื้นหลัง ซึ่งช่วยยืนยันว่าแบบจำลอง Ensemble Learning ที่พัฒนาขึ้นสามารถแยกแยะกลุ่มลูกค้าที่มีแนวโน้มจะยกเลิกบริการกับกลุ่มที่ยังคงใช้บริการได้อย่างมีประสิทธิภาพ



ภาพประกอบ 101 ผลการจำแนกกลุ่มลูกค้าด้วยแบบจำลอง Ensemble Learning บนระนาบสองมิติหลังการลดมิติด้วย PCA

```
[ ] import numpy as np
    import matplotlib.pyplot as plt
    from matplotlib.colors import ListedColormap
    from sklearn.decomposition import PCA

    pca = PCA(n_components=2)
    X_train_pca = pca.fit_transform(X_train)
    X_test_pca = pca.transform(X_test)

    eclf1_pca = VotingClassifier(estimators=[
        ('dt', clf1),
        ('gb', clf2),
        ('ada', clf3)
    ], voting='soft')

    eclf1_pca.fit(X_train_pca, y_train)

    x_min, x_max = X_test_pca[:, 0].min() - 1, X_test_pca[:, 0].max() + 1
    y_min, y_max = X_test_pca[:, 1].min() - 1, X_test_pca[:, 1].max() + 1
    xx, yy = np.meshgrid(np.linspace(x_min, x_max, 300),
                         np.linspace(y_min, y_max, 300))
    Z = eclf1_pca.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.figure(figsize=(10, 6))
    cmap_background = ListedColormap(['#A0C4FF', '#FFADAD'])
    cmap_points = ListedColormap(['#3B82F6', '#EF4444'])

    plt.contourf(xx, yy, Z, alpha=0.4, cmap=cmap_background)
    plt.scatter(X_test_pca[:, 0], X_test_pca[:, 1], c=y_test, cmap=cmap_points, edgecolor='k', s=60)

    plt.title('Data with Binary Response')
    plt.xlabel('x1')
    plt.ylabel('x2')

    plt.legend(handles=[
        plt.Line2D([0], [0], marker='o', color='w', label='0', markerfacecolor='#3B82F6', markeredgecolor='k', markersize=8),
        plt.Line2D([0], [0], marker='o', color='w', label='1', markerfacecolor='#EF4444', markeredgecolor='k', markersize=8)
    ], title="Predicted Regions", loc='lower center', bbox_to_anchor=(0.5, -0.15), ncol=2)

    plt.tight_layout()
    plt.show()
```

ภาพประกอบ 102 โค้ดสำหรับผลการจำแนกกลุ่มลูกค้าด้วยแบบจำลอง Ensemble Learning บนระนาบสองมิติหลังการลดมิติด้วย PCA

จากตาราง 16 ค่า Accuracy เพิ่มขึ้นเล็กน้อย (จาก 96.04% เป็น 96.46%) ซึ่งสะท้อนถึงการปรับปรุงโดยรวมที่ดีขึ้นของโมเดลหลังการ Fine-Tune Recall ของ Class 1 เพิ่มขึ้นจาก 92% เป็น 94% แสดงว่าโมเดลสามารถตรวจจับกลุ่มเป้าหมาย (กลุ่มบวก) ได้แม่นยำมากขึ้น ซึ่งสำคัญมากในบริบทที่ต้องวิเคราะห์กลุ่มเสี่ยง ค่า Precision และ F1-score โดยรวมดีขึ้นหรือคงที่ ทั้งใน Class 0 และ Class 1 ค่าเฉลี่ยแบบ Macro F1-score เพิ่มขึ้นจาก 0.95 เป็น 0.96

แสดงว่าโมเดลมีความสมดุลในการจัดการทั้งสองคลาสได้ดีขึ้น เมื่อพิจารณาร่วมกับผล ROC Curve และ Confusion Matrix โมเดลหลังปรับเทียบยังแสดงให้เห็นถึงความสามารถในการแยกแยะคลาสได้ชัดเจนขึ้น แม้การเพิ่มขึ้นของ Accuracy จะไม่มากนัก แต่โมเดลหลังปรับเทียบแสดงให้เห็นถึง ความสามารถที่ดีขึ้นในการจำแนกกลุ่มเป้าหมาย, ความแม่นยำโดยรวมที่เสถียรขึ้น, และ สมดุลในการจำแนกข้อมูลทั้งสองคลาสที่ดีขึ้น ดังนั้นจึงสามารถสรุปได้ว่า การปรับเทียบโมเดลย่อยภายใน Ensemble Voting Classifier เป็นสิ่งที่ควรทำ เพื่อให้ได้โมเดลที่มีประสิทธิภาพสูงและเหมาะสมสำหรับการใช้งานจริง

ตาราง 16 เปรียบเทียบผลลัพธ์ Ensemble Voting Classifier ก่อนและหลังปรับเทียบ

Metric	ก่อนปรับเทียบ	หลังปรับเทียบ
Accuracy	0.9604	0.9646
Precision (Class 0)	0.96	0.97
Recall (Class 0)	0.98	0.98
F1-score (Class 0)	0.97	0.97
Precision (Class 1)	0.96	0.96
Recall (Class 1)	0.92	0.94
F1-score (Class 1)	0.94	0.95
Macro Avg F1-score	0.95	0.96
Weighted Avg F1-score	0.96	0.96

จากการวิจัยครั้งนี้มีการทดสอบแบบจำลอง 7 แบบ ได้แก่ Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting, Logistic Regression, K-Nearest Neighbors และ AdaBoost Classifier แต่ละแบบจำลองแสดงประสิทธิภาพที่แตกต่างกันในการจำแนกพฤติกรรมการยกเลิกการใช้บริการของลูกค้า โดยโมเดล AdaBoost, Decision Tree และ Gradient Boosting แสดงผลลัพธ์ที่โดดเด่นจึงนำไปปรับแต่งค่าพารามิเตอร์ด้วยเทคนิค Grid Search Cross-Validation และนำมาสร้างแบบจำลองรวม(Ensemble Learning)ด้วยเทคนิค Soft Voting ผลการประเมินแสดงให้เห็นว่าแบบจำลองที่ผ่านการปรับแต่งมีค่า Accuracy และ F1-score สูงขึ้น โดย Ensemble Voting Classifier ที่รวมโมเดลทั้งสามแบบที่ผ่านการ fine-tune มีความแม่นยำเพิ่มขึ้นจาก 96.04% เป็น 96.46% และมีความสามารถในการจำแนกกลุ่มลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการได้ดีขึ้น โมเดลรวมนี้ยังคงรักษาสมดุลระหว่าง Precision และ Recall ได้อย่างมีประสิทธิภาพทั้งในกลุ่มที่เลิกใช้และยังใช้บริการอยู่

บทที่ 5
สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การศึกษานี้มีเป้าหมายเปรียบเทียบประสิทธิภาพของโมเดลการเรียนรู้ของเครื่อง (Machine Learning) จากนั้นนำโมเดลที่มีประสิทธิภาพดีและเหมาะสมกับข้อมูลจำนวน 3 โมเดล มาปรับเทียบ (Fine-tune) และดำเนินการรวมโมเดล (Ensemble Learning) เพื่อสร้างแบบจำลองที่สามารถจำแนกการสูญเสียลูกค้าจากการยกเลิกการใช้บริการได้อย่างแม่นยำ สำหรับโมเดลที่นำมาเปรียบเทียบในงานวิจัยนี้ได้แก่ Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbor และ AdaBoost Classifier รวมทั้งสิ้น 7 โมเดล

จากผลการศึกษาพบว่า โมเดล AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier มีประสิทธิภาพในการจำแนกข้อมูลสูงสุด (ตาราง 17) โดย AdaBoost Classifier ก่อนการปรับเทียบมีค่าความแม่นยำ (Accuracy) อยู่ที่ 96.04% มีค่า Precision ทั้งสองคลาสเท่ากับ 96% ค่า Recall สองคลาสเฉลี่ยเท่ากับ 95% และมีค่า F1-score สองคลาสเฉลี่ยเท่ากับ 95.5% ในขณะที่ Decision Tree ก่อนการปรับเทียบมีค่าความแม่นยำ 95.21% Precision ทั้งสองคลาสเท่ากับ 95% Recall สองคลาสเฉลี่ยเท่ากับ 94.5% และ F1-score สองคลาสเฉลี่ยเท่ากับ 94.5% ส่วน Gradient Boosting Classifier ก่อนการปรับเทียบมีค่าความแม่นยำ 95.83% Precision ทั้งสองคลาส 96% Recall สองคลาสเฉลี่ยเท่ากับ 94.5% และ F1-score สองคลาสเฉลี่ยเท่ากับ 95.5% โดยโมเดลทั้งสามมีค่า Area Under Curve (AUC) สูงกว่า 0.95

เมื่อนำโมเดล AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier มาปรับเทียบค่าพารามิเตอร์ด้วยวิธี Grid Search CV พบว่าโมเดลมีประสิทธิภาพสูงขึ้นและมีความเหมาะสมในการนำไปทำ Ensemble Learning มากยิ่งขึ้น (ตาราง 17) โดย AdaBoost Classifier หลังปรับเทียบมีค่าความแม่นยำ 95.42% Precision ทั้งสองคลาสเท่ากับ 95% Recall สองคลาสเฉลี่ยเท่ากับ 95% F1-score สองคลาสเฉลี่ยเท่ากับ 95% และค่า Cross-Validation Score เท่ากับ 96.78% สำหรับ Decision Tree หลังปรับเทียบมีค่าความแม่นยำ 95.83% Precision สองคลาสเฉลี่ยเท่ากับ 95.5% Recall สองคลาสเฉลี่ยเท่ากับ 95% F1-score สองคลาสเฉลี่ยเท่ากับ 95.5% และมีค่า Cross-Validation Score เท่ากับ 97.67% ขณะที่ Gradient Boosting Classifier หลังปรับเทียบมีค่าความแม่นยำ 96.46% Precision สองคลาสเฉลี่ยเท่ากับ

96.5% Recall สองคลาสเฉลี่ยเท่ากับ 95.5% F1-score สองคลาสเฉลี่ยเท่ากับ 95.5%พร้อมกับมีค่า Cross-Validation Score เท่ากับ 97.13%

สำหรับการสร้าง Ensemble Learning จากทั้ง 3 โมเดล (ตาราง 17) พบว่าแบบจำลอง Ensemble ที่ได้ก่อนการปรับเทียบโมเดลมีค่าความแม่นยำ 96.04% Precision สองคลาสเฉลี่ยเท่ากับ 95% Recall สองคลาสเฉลี่ยเท่ากับ 95% และ F1-score สองคลาสเฉลี่ยเท่ากับ 95.5% หลังจากดำเนินการปรับเทียบโมเดล (Fine-tune model) แล้ว แบบจำลอง Ensemble Learning มีค่าความแม่นยำเพิ่มขึ้นเป็น 96.46% Precision สองคลาสเฉลี่ยเท่ากับ 96.5% Recall สองคลาสเฉลี่ยเท่ากับ 96% และ F1-score สองคลาสเฉลี่ยเท่ากับ 96% ซึ่งสะท้อนให้เห็นถึงประสิทธิภาพที่สูงขึ้นจากการปรับแต่งโมเดลก่อนการรวมกัน

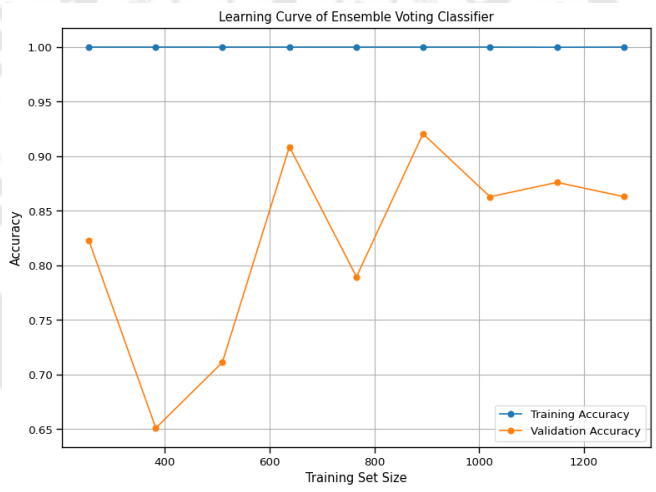
สำหรับการวิเคราะห์ความสำคัญของฟีเจอร์ ได้ใช้วิธีการเปรียบเทียบค่าความสำคัญจากทั้งสามแบบจำลอง Tree-based และคำนวณค่าเฉลี่ยร่วมกัน พบว่าฟีเจอร์ DayFromLastPayment มีอิทธิพลต่อการทำนายสูงที่สุด รองลงมาคือ OverdueInvoiceCount และ PlanName ซึ่งสะท้อนถึงบทบาทของพฤติกรรมด้านการชำระเงินและลักษณะแผนบริการที่มีผลต่อแนวโน้มการเลิกใช้บริการของลูกค้า

นอกจากนี้ จากการเปรียบเทียบค่า Training Accuracy และ Validation Accuracy ของแบบจำลอง Ensemble Learning (ภาพประกอบ 103 และ 104) พบว่าแบบจำลองสามารถเรียนรู้ข้อมูลฝึกได้อย่างสมบูรณ์ แต่ยังพบความแตกต่างระหว่างผลลัพธ์บนข้อมูลฝึกและข้อมูลทดสอบ ซึ่งอาจเป็นสัญญาณของการเกิด overfitting :ซึ่งควรพิจารณาการเพิ่มจำนวนข้อมูล หรือใช้เทคนิคเพื่อลดความซับซ้อนของแบบจำลองในการวิจัยครั้งถัดไป เพื่อเพิ่มความสามารถในการประเมินข้อมูลใหม่อย่างเสถียร

ดังนั้นการศึกษานี้สามารถสร้างแบบจำลองการจำแนกกลุ่มลูกค้าที่มีแนวโน้มจะยกเลิกการใช้บริการได้อย่างมีประสิทธิภาพ ผ่านการเลือกโมเดลที่เหมาะสม การปรับเทียบค่าพารามิเตอร์ และการประยุกต์ใช้เทคนิค Ensemble Learning ทั้งนี้ ผลลัพธ์ที่ได้มีความน่าเชื่อถือสูง และสามารถนำไปใช้เป็นแนวทางในการประยุกต์ใช้ในภาคธุรกิจเพื่อการบริหารความสัมพันธ์กับลูกค้า (Customer Relationship Management) และการวางกลยุทธ์ในการรักษาฐานลูกค้าในธุรกิจที่ดำเนินกิจการในลักษณะ B2B ได้

ตาราง 17 ค่า Accuracy, Precision, Recall, F1-Score, AUC, และ CV Score ของแบบจำลองก่อนและหลัง Fine-tune

แบบจำลอง	ขั้นตอน	Accuracy	Precision	Recall	F1-score	AUC	CV Score
Decision Tree	ก่อน Fine-tune	95.21%	95%	94.5%	94.5%	>0.95	-
	หลัง Fine-tune	95.83%	95.5%	95%	95.5%	>0.95	97.67%
Gradient Boosting	ก่อน Fine-tune	95.83%	96%	94.5%	95.5%	>0.95	-
	หลัง Fine-tune	96.46%	96.5%	95.5%	95.5%	>0.95	97.13%
AdaBoost Classifier	ก่อน Fine-tune	96.04%	96%	95%	95.5%	>0.95	-
	หลัง Fine-tune	95.42%	95%	95%	95%	>0.95	96.78%
Ensemble (Voting)	ก่อน Fine-tune	96.04%	95%	95%	95.5%	>0.95	-
	หลัง Fine-tune	96.46%	96.5%	96%	96%	>0.95	-



ภาพประกอบ 103 Learning Curve แสดงความแม่นยำของ Training และ Validation ของแบบจำลอง Ensemble (Voting)

```
[ ]  from sklearn.model_selection import learning_curve
     import numpy as np
     import matplotlib.pyplot as plt

     ensemble_model = VotingClassifier(
         estimators=[('dt', dt_model), ('ada', ada_model), ('gb', gb_model)],
         voting='soft'
     )

     train_sizes, train_scores, val_scores = learning_curve(
         estimator=ensemble_model,
         X=X, y=y,
         train_sizes=np.linspace(0.1, 1.0, 10),
         cv=5,
         scoring='accuracy',
         n_jobs=-1
     )

     train_mean = np.mean(train_scores, axis=1)
     val_mean = np.mean(val_scores, axis=1)

     plt.figure(figsize=(8, 6))
     plt.plot(train_sizes, train_mean, 'o-', label='Training Accuracy')
     plt.plot(train_sizes, val_mean, 'o-', label='Validation Accuracy')
     plt.title('Learning Curve of Ensemble Voting Classifier')
     plt.xlabel('Training Set Size')
     plt.ylabel('Accuracy')
     plt.legend(loc='best')
     plt.grid(True)
     plt.tight_layout()
     plt.show()
```

ภาพประกอบ 104 โค้ดสำหรับ Learning Curve แสดงความแม่นยำของ Training และ Validation ของแบบจำลอง Ensemble (Voting)

5.2 อภิปรายผล

จากการศึกษาพบว่า โมเดล AdaBoost Classifier, Decision Tree และ Gradient Boosting Classifier เป็นแบบจำลองที่มีประสิทธิภาพสูงที่สุดในการจำแนกกลุ่มลูกค้าที่จะยกเลิกการใช้บริการและกลุ่มที่ยังคงใช้บริการอยู่ โดยทั้งสามแบบจำลองให้ค่าความแม่นยำ (Accuracy) อยู่ในช่วง 95-96% และมีค่า Precision, Recall และ F1-score ของทั้งสองคลาสในระดับที่ดีมาก นอกจากนี้ เมื่อนำแบบจำลองทั้งสามมาทำการปรับเทียบพารามิเตอร์ (Fine-tune) ด้วย Grid Search Cross-Validation ผลลัพธ์ที่ได้แสดงให้เห็นถึงการปรับปรุงในด้านความแม่นยำและความสามารถในการประมวลผลข้อมูลใหม่ (generalization) โดยเฉพาะในด้านค่า Cross-validation score ซึ่งเพิ่มขึ้นอย่างชัดเจนในทุกแบบจำลอง

เมื่อเปรียบเทียบผลลัพธ์ก่อนและหลังการปรับเทียบโมเดล พบว่าแบบจำลองที่ผ่านการปรับเทียบมีประสิทธิภาพโดยรวมดีขึ้น แม้ว่าค่าดัชนีบางตัว เช่น Accuracy และ F1-score จะเพิ่มขึ้นเพียงเล็กน้อย แต่ค่าคะแนน Cross-validation ที่สูงขึ้นถือเป็นตัวชี้วัดที่สำคัญที่แสดงถึงเสถียรภาพของโมเดลและความสามารถในการประเมินข้อมูลใหม่ได้ดีกว่าเดิม นอกจากนี้ การนำ

แบบจำลองที่ผ่านการปรับเทียบมาสร้างเป็น Ensemble Voting Classifier ยังส่งผลให้ผลลัพธ์โดยรวมดีขึ้นกว่า Ensemble เดิม ทั้งในด้าน Accuracy, Precision และ Recall โดยเฉพาะในคลาส 1 (กลุ่มลูกค้าที่จะยกเลิกบริการ) ที่มีความสำคัญเชิงธุรกิจ ซึ่งค่า Recall เพิ่มขึ้นจาก 92% เป็น 94% และ Precision เพิ่มจาก 94% เป็น 96% สะท้อนถึงความสามารถในการตรวจจับกลุ่มเป้าหมายได้แม่นยำยิ่งขึ้น

ในส่วนของการทดสอบ สมมติฐานที่ 1 โมเดล Decision tree จะมีประสิทธิภาพสูงสุดในการทำนายจำแนกการสูญเสียลูกค้าจากการยกเลิกการใช้บริการ เมื่อเปรียบเทียบกับโมเดลอื่น ๆ เช่น Random Forest, Support Vector Machine, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbors, และ AdaBoost Classifier ผลการวิจัยพบว่า โมเดล Decision Tree มีประสิทธิภาพที่ดี โดยเฉพาะหลังการปรับเทียบพารามิเตอร์ซึ่งสามารถให้ค่าความแม่นยำ (Accuracy) ได้ถึง 95.83% และค่า F1-score เฉลี่ยอยู่ที่ 95.5% แต่เมื่อเปรียบเทียบกับโมเดล AdaBoost Classifier และ Gradient Boosting Classifier พบว่า AdaBoost มีค่า Accuracy สูงสุดก่อนการปรับเทียบที่ 96.04% ในขณะที่ Gradient Boosting Classifier มีค่า Accuracy สูงสุดหลังการปรับเทียบที่ 96.46% และทั้งสองโมเดลยังมีค่า Cross-validation score สูงกว่า Decision Tree อย่างชัดเจน สะท้อนให้เห็นว่า Decision Tree เป็นหนึ่งในโมเดลที่มีประสิทธิภาพดี แต่ไม่ใช่โมเดลที่มีประสิทธิภาพสูงสุด สมมติฐานที่ 1 จึงไม่ได้รับการสนับสนุนจากผลการวิจัยอย่างชัดเจน

สำหรับสมมติฐานที่ 2 ลูกค้าที่มีรูปแบบการชำระเงินที่ไม่สม่ำเสมอจะมีแนวโน้มที่จะเลิกใช้บริการมากกว่าลูกค้าที่มีรูปแบบการชำระเงินที่สม่ำเสมอ จากผลการวิเคราะห์ฟีเจอร์จากแบบจำลอง Tree-based ทั้งสามพบว่า ตัวแปร DayFromLastPayment ซึ่งสะท้อนถึงจำนวนวันที่ผ่านไปนับจากการชำระเงินครั้งล่าสุด และตัวแปร OverdueInvoiceCount ที่แสดงถึงจำนวนใบแจ้งหนี้ที่ค้างชำระ มีความสำคัญสูงสุดในการจำแนกกลุ่มลูกค้าที่เลิกใช้บริการ โดยทั้งสองฟีเจอร์เป็นตัวแทนของพฤติกรรมการชำระเงินที่ไม่สม่ำเสมอ ซึ่งสอดคล้องกับแนวคิดในสมมติฐานนี้ ดังนั้น สมมติฐานที่ 2 ได้รับการสนับสนุนจากผลการวิจัย และแสดงให้เห็นถึงความสำคัญของพฤติกรรมการเงินของลูกค้าในการคาดการณ์ความเสี่ยงต่อการเลิกใช้บริการ

จากผลการวิจัยสามารถสรุปได้ว่า การใช้โมเดลการเรียนรู้ของเครื่องร่วมกับกระบวนการ Fine-tune และ Ensemble Learning เป็นแนวทางที่มีประสิทธิภาพในการวิเคราะห์พฤติกรรมของลูกค้า โดยเฉพาะอย่างยิ่งในบริบทของการคาดการณ์การสูญเสียลูกค้า (Customer Churn Prediction) โมเดลที่ได้สามารถนำไปใช้เป็นเครื่องมือประกอบการตัดสินใจเชิงกลยุทธ์ในภาค

ธุรกิจ เช่น การวางแผนรักษาฐานลูกค้า การจัดสรรงบประมาณด้านการตลาดอย่างแม่นยำ ตลอดจนการส่งเสริมความภักดีของลูกค้าในระยะยาว ซึ่งถือเป็นประโยชน์ที่สามารถนำไปสู่การเพิ่มมูลค่าทางธุรกิจอย่างยั่งยืน

5.3 ข้อจำกัดและอุปสรรคของงานวิจัย

งานวิจัยในครั้งนี้ใช้ข้อมูลจริงจากบริษัทผู้ให้บริการด้านการจัดการทรัพยากรบุคคลแห่งหนึ่ง ซึ่งเป็นผู้พัฒนาและให้บริการแพลตฟอร์มสำหรับการบริหารจัดการทรัพยากรบุคคล โดยข้อมูลที่นำมาใช้มีจำนวนทั้งสิ้น 1,597 แถว ซึ่งแม้ว่าจะเป็นข้อมูลจากสถานการณ์การใช้งานจริง และสามารถสะท้อนพฤติกรรมของผู้ใช้งานได้ในระดับหนึ่ง แต่อาจยังมีข้อจำกัดในแง่ของ ขนาดของชุดข้อมูล ที่ไม่ใหญ่เพียงพอสำหรับการฝึกแบบจำลองที่มีความซับซ้อนสูง หรือการประเมินผลในเชิงสถิติที่ต้องการความแม่นยำสูงในระดับประชากร

นอกจากนี้ เนื่องจากข้อมูลที่ใช้ในการวิจัยเป็นข้อมูลที่มาจากระบบการใช้งานจริง จึงอาจมีการเปลี่ยนแปลงอยู่ตลอดเวลา ทั้งในด้านของคุณลักษณะข้อมูล พฤติกรรมของผู้ใช้ หรือการปรับเปลี่ยนฟีเจอร์ภายในแพลตฟอร์ม ซึ่งอาจส่งผลต่อความเสถียรและความต่อเนื่องของรูปแบบพฤติกรรมที่แบบจำลองได้เรียนรู้ไว้ ทำให้แบบจำลองที่สร้างขึ้นจากข้อมูลชุดนี้อาจมีข้อจำกัดในการใช้งานกับข้อมูลใหม่ในอนาคตหากไม่มีการอัปเดตหรือปรับเทียบโมเดลอย่างต่อเนื่อง

ดังนั้น ข้อจำกัดเหล่านี้ควรถูกนำมาพิจารณาในการวางแผนการนำแบบจำลองไปใช้จริง ตลอดจนการพัฒนางานวิจัยต่อในอนาคต ซึ่งอาจต้องอาศัยข้อมูลที่มีขนาดใหญ่ขึ้น และมีความหลากหลายมากยิ่งขึ้น รวมถึงการออกแบบกระบวนการปรับปรุงโมเดลอย่างสม่ำเสมอเพื่อรองรับการเปลี่ยนแปลงของข้อมูลในสภาพแวดล้อมจริง

5.4 ข้อเสนอแนะ

5.4.1 ควรพิจารณาใช้ ชุดข้อมูลที่มีขนาดใหญ่และหลากหลายมากยิ่งขึ้น เพื่อเพิ่มความครอบคลุมและความแม่นยำของแบบจำลอง โดยเฉพาะการนำข้อมูลจากหลายช่วงเวลา หรือจากกลุ่มลูกค้าที่มีลักษณะแตกต่างกัน จะช่วยให้โมเดลสามารถเรียนรู้รูปแบบพฤติกรรมได้หลากหลายขึ้น และเพิ่มความสามารถในการนำไปประยุกต์ใช้กับข้อมูลใหม่ได้อย่างมีประสิทธิภาพ

5.4.2 ควรมีการอัปเดตแบบจำลองอย่างต่อเนื่อง (model re-training) เพื่อรองรับการเปลี่ยนแปลงของพฤติกรรมผู้ใช้และโครงสร้างข้อมูลในระบบจริง ซึ่งมักมีการเปลี่ยนแปลงอยู่ตลอดเวลา การพัฒนา pipeline ที่สามารถตรวจสอบและปรับเทียบโมเดลอย่างสม่ำเสมอจะช่วยให้แบบจำลองสามารถคงประสิทธิภาพในระยะยาวได้

5.4.3 ในขั้นตอนการแปลงข้อมูลเชิงหมวดหมู่เป็นข้อมูลเชิงตัวเลข งานวิจัยนี้เลือกใช้วิธี Label Encoding เพื่อหลีกเลี่ยงความซับซ้อนของข้อมูล อย่างไรก็ตาม ควรมีการทดลองเปรียบเทียบผลลัพธ์กับการใช้ One-Hot Encoding เพื่อประเมินความเหมาะสมเพิ่มเติม

บรรณานุกรม

1. Zenkina, I., *E-commerce development as a business trend and driver of e-commerce analytics.* Economic Analysis: Theory and Practice, 2025. 24: p. 4-22.
2. วังคีอาจ, พ., & พลประเสริฐ, ช., *Machine Learning Approach to Predict E-commerce Customer Satisfaction Score*, in 2023 8*th International Conference on Business and Industrial Research (ICBIR)*. 2023, IEEE. p. 1–6.
3. Jiang, X., *Customer churn data analysis using data mining*, in 2024 4*th International Conference on Applied Computing and Engineering Technology (ACET* 2024). 2024, EWA Publishing.
4. *Machine Learning Methods, Techniques and Applications.* International Journal for Research in Applied Science and Engineering Technology, 2025. 13: p. 7220-7224.
5. Afolabi, S., et al., *Predicting diabetes using supervised machine learning algorithms on E-health records.* Informatics and Health, 2025. 2(1): p. 9-16.
6. Oye, E., E. Frank, and J. Owen, *Unsupervised and Supervised Models in Machine Learning.* 2024.
7. Algorithms, N., *A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms.* 1999.
8. Hou, R., et al., *Decision Tree Extraction for Clinical Decision Support System With If-Else Pseudocode and PlanSelect Strategy.* IEEE Journal of Biomedical and Health Informatics, 2025. **PP**: p. 1-13.
9. Revelas, C., O. Boldea, and B. Werker, *When do Random Forests work?* 2025.
10. Lederer, J., *Support-Vector Machines*. 2025. p. 187-243.
11. Ok, E. and M. Emmanuel, *Understanding the Gradient Boosting Algorithm in XGBoost.* 2025.
12. Trần, V., *Logistic Regression*. 2025.
13. Ortiz-Villaseñor, D., et al., *K-Nearest Neighbors Regression and Applications*. 2025. p. 295-316.

14. Mathanker, S., et al., *AdaBoost classifiers for pecan defect classification.* Computers and Electronics in Agriculture, 2011. 77: p. 60-68.
15. healthcare, M., *Machine learning algorithms for healthcare.* World Journal of Advanced Research and Reviews, 2025. 25: p. 1139-1143.
16. Bairam, D.M. and S. Layaq, *Asymptomatic Data: An Instance Of Imbalance Data.* Journal of Critical Reviews, 2024. 7: p. 2020.
17. Laksono, B., et al., *Integration of SMOTE and Ensemble Models for Predicting Airline Passenger Satisfaction.* Innovation in Research of Informatics (Innovatics), 2025. 7.
18. Christensen, R., *Model Selection Methods and Model Evaluation*. 2025. p. 235-282.
19. Alshammari, A., *Implementation of Model Evaluation using Confusion Matrix in Python.* International Journal of Computer Applications, 2024. 186: p. 42-48.
20. Wu, Y. and C. Cannistraci, *Accuracy Score for Evaluation of Classification on Imbalanced Data*. 2025.
21. Olojede, O., O. Olatunde, and O. Alo, *Performance Evaluation of Some Machine Learning Models for Music Genre Classification.* International Journal of Latest Technology in Engineering Management & Applied Science, 2025. 14: p. 18-24.
22. Sun, Y., A. Tenesa, and J. Vines, *Human-Precision Medicine Interaction: Public Perceptions of Polygenic Risk Score for Genetic Health Prediction*. 2025. 1-26.
23. Garg, A., *Optimizing Supernova Classification: A Metric-Aware Approach with PR-AUC and F1-score*. 2025.
24. Fanjul Hevia, A., J.C. Pardo-Fernandez, and W. Gonzãlez-Manteiga, *A new test for assessing the covariate effect in ROC curves*. 2024.
25. Bingren, C., et al., *A Logarithm Depth Quantum Converter: From One-hot Encoding to Binary Encoding*. 2022.
26. Fatmawati and N. Rifai, *Klasifikasi Penyakit Diabetes Retinopati Menggunakan Support Vector Machine dengan Algoritma Grid Search Cross-validation.* Jurnal Riset Statistika, 2023: p. 79-86.

27. Christen, P., R. Schnell, and A. Vidanage, *Information Leakage in Data Linkage*. 2025.

ประวัติผู้เขียน

ชื่อ-สกุล ญาตา ชัยยารังกิจรัตน์

สถานที่เกิด กรุงเทพฯ