เรื่อง การทำนายสินค้าคงคลัง โดยใช้เทคนิคเหมืองข้อมูล (กรณีศึกษา: สินค้าค้างส่ง)

สารนิพนธ์
ของ
เจนจิรา มหาสุข

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
กรกฎาคม 2561
ลิขสิทธิ์เป็นของมหาวิทยาลัยศรีนครินทรวิโรฒ

เรื่อง การทำนายสินค้าคงคลัง โดยใช้เทคนิคเหมืองข้อมูล (กรณีศึกษา: สินค้าค้างส่ง)

สารนิพนธ์
ของ
เจนจิรา มหาสุข

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
กรกฎาคม 2561

เรื่อง การทำนายสินค้าคงคลัง โดยใช้เทคนิคเหมืองข้อมูล (กรณีศึกษา: สินค้าค้างส่ง)

บทคัดย่อ

ของ

เจนจิรา มหาสุข

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
กรกฎาคม 2561

เจนจิรา มหาสุข. (2561). *การทำนายสินค้าคงคลังโดยใช้เทคนิคของเหมืองข้อมูล (กรณีศึกษา: สินค้าค้างส่ง)*. สารนิพนธ์ วท.ม. (เทคโนโลยีสารสนเทศ). กรุงเทพฯ: บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ. อาจารย์ที่ปรึกษาสารนิพนธ์: ผู้ช่วยศาสตราจารย์ ดร.นุวีย์ วิวัฒนวัฒนา.

การวิจัยในครั้งนี้ได้นำเสนอการทำนายสินค้าคงคลังในกรณีของสินค้าค้างส่ง หรือเป็นสินค้าที่ไม่สามารถจัดจำหน่ายได้ตามคำสั่งซื้อ (backorders) ซึ่งได้รับการสนับสนุนชุดข้อมูลจาก www.kaggle.com โดยมุ่งเน้นศึกษาเทคนิคการเรียนรู้ของเครื่อง (Machine Learning) ด้วยรูปแบบการเรียนรู้แบบมีผู้สอน (Supervised Learning) มีวัตถุประสงค์เพื่อศึกษาการจัดการกับข้อมูลที่ไม่สมดุล (Imbalanced) ที่ใช้ในการทำนายสินค้าคงคลังในกรณีการจำแนกประเภทสินค้าค้างส่ง (Classification) ซึ่งข้อมูลประเภทนี้มีสัดส่วนที่แตกต่างกันมาก หากนำไปสร้างแบบจำลองเพื่อทดสอบการจำแนกประเภทจะทำให้ผลทำนายมีแนวโน้มไปยังสัดส่วนที่มีจำนวนมากกว่าและส่งผลต่อความน่าเชื่อถือของผลลัพธ์จากการจำแนกประเภทด้วย นอกจากนี้ยังเปรียบเทียบประสิทธิภาพการจำแนกข้อมูลของอัลกอริทึมที่สนใจด้วยตัวชี้วัดหลายประเภทโดยต้องปรับปรุงและแบ่งข้อมูลด้วยวิธีการแบ่งแบบ Stratified Shuffle Split จากการศึกษาพบว่า อัลกอริทึม Random Forest มีความแม่นยำและมีค่าความถูกต้องมากที่สุดเมื่อเปรียบเทียบกับอัลกอริทึมอื่น คือ Accuracy เท่ากับร้อยละ 99.43 ค่าความถูกต้องของสินค้าที่เป็น backorders ได้แก่ ค่า Precision คิดเป็นร้อยละ 79.77 ค่า Recall คิดเป็นร้อยละ 20.60 ค่า F-1 score คิดเป็นร้อยละ 32.75 และค่าความถูกต้องของสินค้าที่ไม่เป็น backorders คือ ค่า Precision คิดเป็นร้อยละ 99.47 ค่า Recall คิดเป็นร้อยละ 99.96 ค่า F-1 score คิดเป็นร้อยละ 99.72 นอกจากนี้อัลกอริทึม Random Forest ยังให้ผลลัพธ์ของค่า Average Precision สูงสุดเมื่อเปรียบเทียบกับทุกอัลกอริทึม เท่ากับ 0.43 และมีพื้นที่ใต้กราฟ Precision-Recall มากที่สุดด้วย

INVENTORY PREDICTION USING DATA MINING TECHNIQUES
(CASE STUDY: BACKORDERS)

AN ABSTRACT

BY

JANJIRA MAHASOOK

Presented in Partial Fulfillment of the Requirements for the

Master of Science in Information Technology

at Srinakharinwirot University

July 2018

Janjira Mahasook. (2018). *Inventory Prediction Using Data Mining Techniques (Case Study: Backorders*) . Master's Project (Science in Information Technology) . Bangkok: Graduate School, Srinakharinwirot University. Project Advisor: Asst.Prof. Dr. Nuwee Wiwatwattana.

This research aims to study different algorithms to predict pending goods or Back Orders in the inventory using data supported by www.kaggle.com. In the case study, the focus was on different machine learning techniques with supervised machine learning methods. The main objective was to study the imbalanced situation for backorders classification with a different proportion of data. The imbalanced data resulted in bias towards the majority of data and reported evaluation metrics which were not able to reflect in the actual performances. The performance of all of these algorithms were compared using several metrics. Since the number of products were with backorders were substantially less than without imbalanced backorders data. The Stratified Shuffle Split was applied to the splitting of training and testing datasets. The results revealed that the Random Forest excelled in every metric with an accuracy of 99.43 percent, however backorders prediction had a precision of 79.77 percent, a recall of 20.60 percent and an F-1score of 32.75. Non-backorders prediction had a precision of 99.47 percent a recall of 99.96 percent and an F-1 score of 99.72 percent. In addition, Random Forest yielded the highest average precision of 0.43 and the largest area under the Precision Recall curve.

งานวิจัยนี้ได้รับทุนอุดหนุนการวิจัย
จาก
บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

ประกาศคุณูปการ

งานวิจัยฉบับนี้สำเร็จลุล่วงไปด้วยดี เนื่องจากได้รับความอนุเคราะห์ของ ผศ.ดร.นุวีย์ วิวัฒนวัฒนา ซึ่งเป็นอาจารย์ที่ปรึกษางานวิจัยนี้ซึ่งท่านได้ให้คำแนะนำและข้อคิดเห็นต่าง ๆ อันเป็นประโยชน์อย่างยิ่งในการทำวิจัย ตลอดจนปรับปรุงแก้ไขข้อบกพร่องต่าง ๆ ที่เกิดขึ้นระหว่างการดำเนินการวิจัย

ขอขอบพระคุณคณาจารย์ คณะวิทยาศาสตร์ สาขาเทคโนโลยีสารสนเทศ ทุกท่านที่ได้อบรมและสั่งสอนให้วิชาความรู้เพื่อนำไปต่อยอดและให้คำแนะนำให้งานวิจัยนี้ถูกต้องสมบูรณ์ยิ่งขึ้น

ขอขอบพระคุณบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒที่อนุเคราะห์ทุนอุดหนุนในการวิจัยรวมไปถึงเจ้าหน้าที่ที่เกี่ยวข้องที่กรุณาให้คำแนะนำและประสานงานให้สำเร็จด้วยดี

ขอขอบคุณเพื่อนๆนักศึกษาระดับปริญญาโท หลักสูตร วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ ที่ช่วยให้คำแนะนำอันเป็นประโยชน์และช่วยเหลือเกื้อกูลเพื่อให้งานวิจัยนี้ลุล่วงไปได้ด้วยดี

สุดท้ายนี้ ผู้วิจัยขอขอบพระคุณครอบครัวที่เป็นกำลังใจที่สำคัญ และสนับสนุนเรื่องการศึกษาในทุกๆด้านจนกระทั่งทุกอย่างประสบความสำเร็จและลุล่วงมาจนถึงวันนี้ ขอขอบพระคุณอย่างสูง

เจนจิรา มหาสุข

สารบัญ

สารบัญ (ต่อ)

บัญชีตาราง

ตาราง **หน้า**

บัญชีภาพประกอบ

ภาพประกอบ **หน้า**

บัญชีภาพประกอบ

ภาพประกอบ **หน้า**

บทที่ 1

บทนำ

1. ความสำคัญของปัญหา

ปัจจุบันธุรกิจและการค้าหลายประเภท เช่น ธุรกิจค้าปลีก ธุรกิจซื้อมา-ขายไป ฯลฯ มีสินค้าคงคลัง (Inventory) เป็นปัจจัยที่จำเป็นอย่างหนึ่งเพื่อให้ธุรกิจสามารถดำเนินไปได้ "สินค้าคงคลัง หมายถึง วัสดุหรือสินค้าต่างๆ ที่ธุรกิจมีสำรองไว้เพื่อการใช้งาน เพื่อการผลิต เพื่อการจัดจำหน่ายในอนาคต เช่น ชิ้นส่วนอะไหล่ วัตถุดิบ รวมไปถึงสินค้าสำเร็จรูปต่างๆ" [1] ดังนั้น การบริหารสินค้าคงคลังที่มีประสิทธิภาพ และเพียงพอต่อความต้องการของลูกค้าจะทำให้การดำเนินการทางธุรกิจเป็นไปได้อย่างราบรื่น สร้างรายได้และลดความเสี่ยงต่างๆให้แก่องค์กรเป็นจำนวนมาก สำหรับธุรกิจที่ต้องสำรองสินค้าไว้จำหน่ายเพื่อตอบสนองคำสั่งซื้อของลูกค้า ผู้ดำเนินธุรกิจมักประสบปัญหาหลากหลายสถานการณ์ที่เกิดจากการจัดการสินค้าคงคลังที่ไม่มีประสิทธิภาพ เช่น การมีสินค้าคงคลังมากเกินไป (Overstock) เป็นปัญหากับธุรกิจทั้งในเรื่องต้นทุนการเก็บรักษาที่สูง สินค้าเสื่อมสภาพ หมดอายุ ล้าสมัย ยกตัวอย่างเช่น กรณีสินค้าประเภท bakery เช่น ขนมปัง เป็นสินค้าที่มีข้อจำกัดในเรื่องของการหมดอายุ หากมีการซื้อสินค้ามาเก็บไว้จำนวนมากเกินไปจนไม่ได้มีการหมุนเวียน (Dead stock) ออกไปจำหน่าย จะทำให้สูญเสียโอกาสต้นทุนของสินค้าชนิดนี้ไปหาประโยชน์ในด้านอื่นๆ นอกจากนี้ สถานการณ์ที่เกิดขึ้นและมักเป็นปัญหาและที่ยากต่อการควบคุม คือ สถานการณ์ที่มีสินค้าไม่เพียงพอต่อความต้องการ (backorders) ซึ่งเป็นผลมาจากประสิทธิภาพประกอบดีจากการขายสินค้า เช่น ช่วงที่มีน้ำท่วมในปี 2554 ลูกค้ามีความต้องการน้ำดื่มเพื่อยังชีพทำให้น้ำดื่มหมดไปจากร้านสะดวกซื้อเกือบทุกแห่งและขาดตลาดในที่สุด สถานการณ์การเกิด backorders ของสินค้านำไปสู่การยกเลิกคำสั่งซื้อทำให้สูญเสียลูกค้า ความน่าเชื่อถือ และความภักดีต่อผลิตภัณฑ์นั้นอีกด้วย

ในปัจจุบันองค์กรธุรกิจมีการตรวจสอบสินค้าและติดตามการเคลื่อนไหวของสินค้าคงคลังด้วยวิธีต่างๆ เช่นการทำใบบันทึกรายการสินค้า (Stock card) เพื่อใช้ในการบันทึกรายการซื้อ-ขายและรับ-คืนสินค้าหรือแม้กระทั่งทฤษฎีต่างๆที่เกี่ยวข้องเช่น ทฤษฎี ABC Analysis ซึ่งเป็นเครื่องมือที่ใช้ในการจัดแบ่งสินค้าคงคลังโดยจะตามมูลค่าออกเป็นกลุ่ม เพื่อช่วยในการตัดสินใจว่าสินค้าคงคลังใดควรได้รับการควบคุมในระดับใด แต่วิธีเหล่านั้นยังคงใช้การตรวจสอบด้วยคนเป็นหลักซึ่งอาจทำให้เกิดปัญหาสินค้าที่บันทึกลงไม่ตรงกับสินค้าที่มีอยู่จริง เกิดความล่าช้า และความผิดพลาดของข้อมูลทำให้ไม่สามารถรู้ยอดสินค้าคงคลังที่แท้จริง

จากปัญหาดังกล่าวจึงมีหลายบริษัทที่นำคอมพิวเตอร์เข้ามาใช้ในการคาดเดาสินค้าคงคลัง

ตัวอย่างเช่น บางบริษัทให้พนักงานคลังสินค้าเก็บข้อมูลการเคลื่อนไหวของสินค้าคงคลังแล้วให้เจ้าหน้าที่ฝ่ายจัดซื้อบันทึกข้อมูลเหล่านั้นลงไปในโปรแกรม Microsoft Excel แล้วใช้สูตรของโปรแกรมช่วยในคำนวณด้วยการบวกหรือหักลบเมื่อมีสินค้าเข้าหรือออกซึ่งวิธีดังกล่าวยังคงใช้การคาดเดาด้วยคนเป็นหลัก หรือใช้ดูจากข้อมูลการพยากรณ์การขายอย่างเดียว ซึ่งอาจยังไม่แม่นยำและข้อมูลที่ได้ยังไม่ถูกต้องและมีประสิทธิภาพในการนำมาทำนายการจัดซื้อสินค้าเพื่อตอบสนองความต้องการของลูกค้าได้ตามต้องการและเหมาะสม ผู้วิจัยจึงมีความสนใจที่จะทำการศึกษาการใช้การจำแนกประเภท (Classification rule) สำหรับทำนายข้อมูลรายการสินค้าที่ไม่เพียงพอต่อคำสั่งซื้อโดยใช้กรณีศึกษาของสินค้า backorders ด้วยการนำเทคนิคการเรียนรู้ของเครื่อง (Machine Learning) มาใช้ในการทำนายข้อมูลและจัดการกับสินค้าคงคลังให้มีความถูกต้อง และแม่นยำมากขึ้น

2. วัตถุประสงค์ของการวิจัย

ในการวิจัยครั้งนี้ผู้วิจัยได้ตั้งวัตถุประสงค์ในการทำงานวิจัยไว้ดังต่อไปนี้

2.1 เพื่อศึกษาการสร้างแบบจำลอง (Modeling) ในกรณีของจำแนกประเภทสินค้าค้างส่ง (backorders) ด้วยเทคนิคเหมืองข้อมูล (Data Mining) โดยอาศัยหลักการเรียนรู้ของเครื่อง (Machine Learning)

2.2 เพื่อศึกษาการเปรียบเทียบประสิทธิภาพของอัลกอริทึมที่ใช้ในการทำนายสินค้าคงคลังรวมไปถึงเทคนิคที่ใช้ในกรณีการจำแนกประเภทสินค้าค้างส่ง (backorders)

2.3 เพื่อศึกษาการใช้เทคนิคการแบ่งกลุ่มข้อมูลฝึกฝนและกลุ่มข้อมูลทดสอบ ตลอดการประเมินผลด้วยตัวชี้วัดที่เหมาะสมสำหรับข้อมูลที่มีความไม่สมดุลกัน (Imbalanced Data)

3. ขอบเขตการวิจัย

3.1 งานวิจัยนี้ทำการศึกษาการสร้างแบบจำลองในการทำนายข้อมูลด้วยเทคนิคเหมืองข้อมูล โดยใช้หลักการเรียนรู้ของเครื่อง ซึ่งข้อมูลตัวอย่างที่ใช้ในการศึกษาเป็นข้อมูลสินค้าของสินค้าคงคลังแห่งหนึ่งในหัวข้อ “Can You Predict Product backorders? Based on historical data predict backorders risk for products” โดยคำตอบที่ต้องการทราบจากการทำนายคือ สินค้าชนิดนั้นๆเป็นสินค้าที่ backorders หรือไม่ โดยข้อมูลนี้ได้ถูกเผยแพร่ไว้บนฐานข้อมูลของเว็บไซต์ https://www.kaggle.com/tiredgeek/predict-bo-trial/kernels

3.2 งานวิจัยที่ทำการศึกษาเทคนิคเหมืองข้อมูลที่จะประยุกต์ในงานวิจัยนี้ ประกอบด้วยเทคนิคการจำแนกประเภทแบบป่าสุ่ม (Random Forest) และ วิธีความใกล้เคียงกันมากที่สุด K-nearest neighbor (KNN) จากการเรียนรู้ของผู้ทำการศึกษามาก่อน แล้วนำไปปรับใช้กับรูปแบบของปัญหาที่เป็นการจำแนกประเภท (Classification) และนอกจากนี้ศึกษาอัลกอริทึมที่ใช้ในการจำแนกเพิ่มเติมได้แก่ นาอีฟ เบย์ (Naive Bayes) ต้นไม้การตัดสินใจ (Decision Tree) การจำแนก

แบบ Adaptive Boosting Algorithm (AdaBoost) และ การจำแนกแบบ Extreme Gradient Boosting (XGBoost) โดยจะวัดประสิทธิภาพและความแม่นยำในการทำนายของแต่ละเทคนิคแล้วนำมาเปรียบเทียบประสิทธิภาพของแต่ละอัลกอริทึมให้ได้อัลกอริทึมที่มีประสิทธิภาพมากที่สุด

3.3 เครื่องมือและเทคโนโลยีที่ผู้วิจัยเลือกใช้กับงานวิจัยนี้มีดังนี้

3.3.1 ภาษาโปรแกรม Python เวอร์ชั่น 3

3.3.2 Library ที่สามารถใช้ร่วมกับภาษา Python 3 ได้แก่ Numpy Pandas Matplotlib Seaborn Scikit-learn

3.3.3 เครื่องมือที่ใช้เขียนคำสั่ง และประมวลผล คือ Jupyter Notebook (https://csit.swu.ac.th)

4. วิธีดำเนินการวิจัย

4.1 ศึกษาค้นคว้าทบทวนงานวิจัยที่เกี่ยวข้อง (Literature Review) ที่เกี่ยวข้องกับขั้นตอนการทำ Data Mining Techniques Machine Learning Techniques รวมไปถึงการทำงานของอัลกอริทึมที่ใช้ในการจำแนกประเภท (Classification)

4.2 ศึกษาและทำความเข้าใจชุดข้อมูลที่เราต้องการศึกษา เช่น ประเภทของข้อมูล ลักษณะของข้อมูล ระบุคำตอบที่ต้องการทราบจากข้อมูลชุดนี้

4.3 ศึกษาการเตรียมข้อมูลให้อยู่ในรูปแบบที่พร้อมต่อการใช้งานกับแบบจำลองที่สร้างขึ้นด้วยกระบวนการ Data Preparation

4.4 ศึกษาการการแบ่งชุดข้อมูลฝึกฝน (Training data) ก่อนนำมาทำการสร้างแบบจำลองเพื่อลดการเกิดข้อมูลไม่สมดุลกัน

4.5 ศึกษาการสร้างแบบจำลองเพื่อใช้อัลกอริทึมที่ใช้ในการจำแนกประเภท

4.6 ศึกษาการทดสอบประสิทธิภาพ และความแม่นยำของอัลกอริทึมโดยอาศัยตัวชี้วัดดังต่อไปนี้ได้แก่ Accuracy Precision Recall F1 score กราฟ ROC Curve ค่า Average Precision และกราฟ Precision-Recall แล้วนำมาเปรียบเทียบกัน

4.7 ทำการประเมินผลและจัดทำผลสรุปของงานวิจัยนี้

5. ประโยชน์ที่คาดว่าจะได้รับจากการวิจัย

5.1 ใช้กระบวนการที่สร้างขึ้นมาเป็นแนวทางในการจัดการกับระบบสินค้าคงคลังให้มีประสิทธิภาพ เช่น ลดต้นทุนของค่าใช้จ่ายในการจัดเก็บสินค้าที่ไม่จำเป็นและในทางกลับกันยังช่วยเพิ่มผลกำไรให้กับองค์กรอีกด้วย

5.2 ได้แบบจำลองที่ช่วยลดความผิดพลาดในการตรวจสอบรายการสินค้าที่เป็น backorders และ จำนวนสินค้าจากคลังสินค้าด้วยกระบวนการและแนวทางที่สร้างขึ้น

5.3 ได้แบบจำลองที่ช่วยลดความเสียหายจากปัญหาสินค้าไม่ทันต่อความต้องการ

บทที่ 2
แนวคิด ทฤษฎี งานวิจัยที่เกี่ยวข้องในการวิจัย

สำหรับการศึกษาในครั้งนี้ผู้วิจัยได้รวบรวมทฤษฎีและงานวิจัยที่เกี่ยวข้องกับการประยุกต์ใช้เทคนิคการทำเหมืองข้อมูล (Data Mining) ด้วยวิธีของการเรียนรู้ของเครื่อง (Machine Leaning) ในรูปแบบของการเรียนรู้แบบมีผู้สอน (Supervised Learning) โดยใช้หลักการการจำแนกประเภทข้อมูล (Classification) กับข้อมูลที่เกิดขึ้นทางธุรกิจซึ่งเป็นข้อมูลที่ซับซ้อน และรวมไปถึงงานวิจัยที่อ้างอิงที่เกี่ยวข้องกับการจัดการกับข้อมูลที่ไม่สมดุลกัน และหลักการทำงานของอัลกอริทึมที่ใช้ในการจำแนกประเภทของข้อมูลไว้ดังต่อไปนี้

1. ทฤษฎีที่เกี่ยวข้อง

1.1 ทฤษฎีการทำเหมืองข้อมูล (Data Mining)

เป็นกระบวนการในการกระทำกับข้อมูลที่มีขนาดใหญ่ เพื่อศึกษาหารูปแบบ แนวทาง จากชุดข้อมูลนั้นโดยในกระบวนการการทำ Data Mining จะอาศัยหลักการทางสถิติ และ การเรียนรู้ของเครื่อง เพื่อให้ได้ผลลัพธ์ที่สามารถนำไปใช้ประโยชน์ได้ และใช้ในการตัดสินใจในการทำธุรกิจ [2] ซึ่งเทคนิคของการทำ Data Mining ประกอบไปด้วยเทคนิคต่างๆ ได้แก่ เทคนิคที่ใช้ในการหาความเชื่อมโยงของ (Association Analysis Technique) เทคนิคที่ใช้ในการจำแนกกลุ่มข้อมูล (Cluster analysis Technique) รวมไปถึงเทคนิคการจำแนกประเภทข้อมูล (Classification Technique) ซึ่งเป็นเทคนิคที่ใช้ในการศึกษากับงานวิจัยนี้

Classification Technique เป็นกระบวนการสร้างแบบจำลองที่เป็นเทคนิคที่ใช้ในการแบ่งประเภทข้อมูลโดยจะต้องมีชุดข้อมูลที่ใช้ในการฝึกฝน หรือชุดข้อมูลต้นแบบ (Training Data) กับชุดข้อมูลทดสอบ (Test Data) เรียกการเรียนรู้จากชุดข้อมูลเหล่านี้ว่า การเรียนรู้แบบมีผู้สอน (Supervised Learning) กระบวนการการเรียนรู้ของเครื่องในรูปแบบการเรียนรู้แบบมีผู้สอนสำหรับใช้ในการจำแนกประเภทของข้อมูลการทำการศึกษาในครั้งนี้สามารถแบ่งได้เป็นขั้นตอนดังต่อไปนี้ [3]

1.1.1 ระบุปัญหาและคำตอบที่ต้องการ
1.1.2 การเตรียมข้อมูล (Data Preparation)
1.1.3 ระบุชุดข้อมูลฝึกฝน (Training data)
1.1.4 เลือกอัลกอริทึมและสร้างแบบจำลอง (Modeling)
1.1.5 ทดสอบประสิทธิภาพกับชุดข้อมูลทดสอบ (Test data)
1.1.6 ประเมินผลแบบจำลองที่สร้างขึ้น (Evaluation)
1.1.7 นำแบบจำลองที่ได้ไปใช้งาน (Deployment)

1.2 ทฤษฎีที่เกี่ยวกับปัญหาข้อมูลที่ไม่สมดุลกัน (Imbalanced Problem)

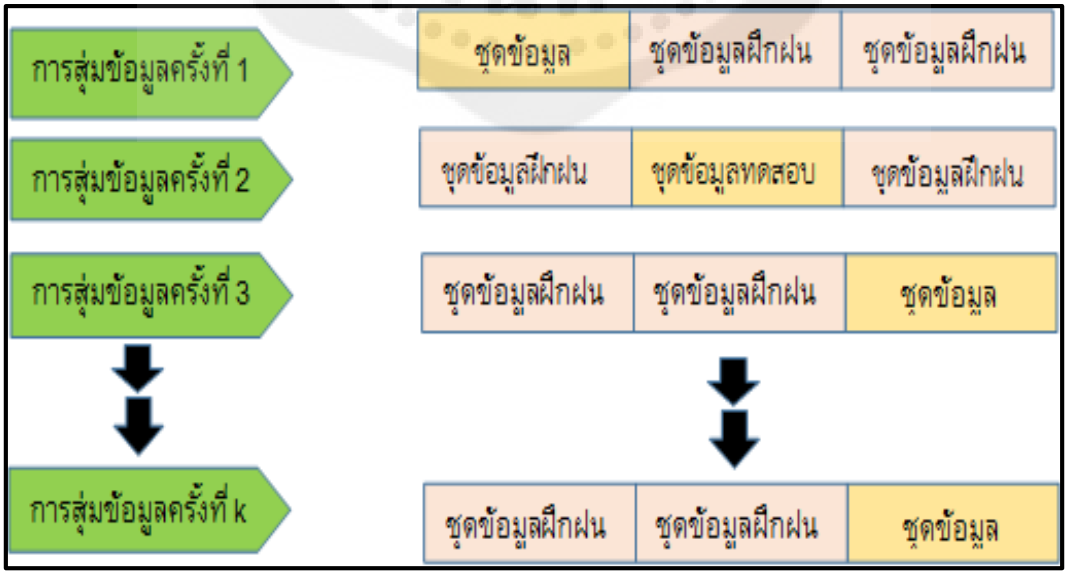
ปัญหาข้อมูลที่ไม่สมดุลกันมักเกิดขึ้นกับการเรียนรู้ในรูปแบบ Supervised Learning

ซึ่งเป็นปัญหาที่สำคัญสำหรับการจำแนกข้อมูล เนื่องจากสัดส่วนของคลาสของข้อมูลที่สนใจมีสัดส่วนที่มีความแตกต่างกันมากส่งผลต่อประสิทธิภาพการจำแนกข้อมูลทำให้ข้อมูลที่มีแนวโน้มเอนเอียงผลลัพธ์ไปในทิศทางใดทิศทางหนึ่งมากเกินไป โดยในงานวิจัยนี้จะกล่าวถึงเกี่ยวกับกรณีของข้อมูลที่ไม่สมดุลคือคลาสที่เราสนใจสินค้านั้นเป็นสินค้าที่เป็น backorders หรือไม่ แต่พบว่าสินค้า backorders มีจำนวนน้อยกว่าสินค้าที่ไม่เป็น backorders อยู่จำนวนมาก เนื่องจากเกือบร้อยละ 100 เป็นสินค้าที่ไม่เป็น backorders ทำให้หากเรานำข้อมูลชุดนั้นมาทั้งหมดโดยปราศจากการจัดการกับข้อมูลที่ไม่สมดุลก่อนที่จะสร้างแบบจำลองจะทำให้ผลลัพธ์ที่ได้ของค่าความถูกต้องที่มีค่าสูงเกินไปและส่งผลต่อการวัดผลที่มีประสิทธิภาพ [3]

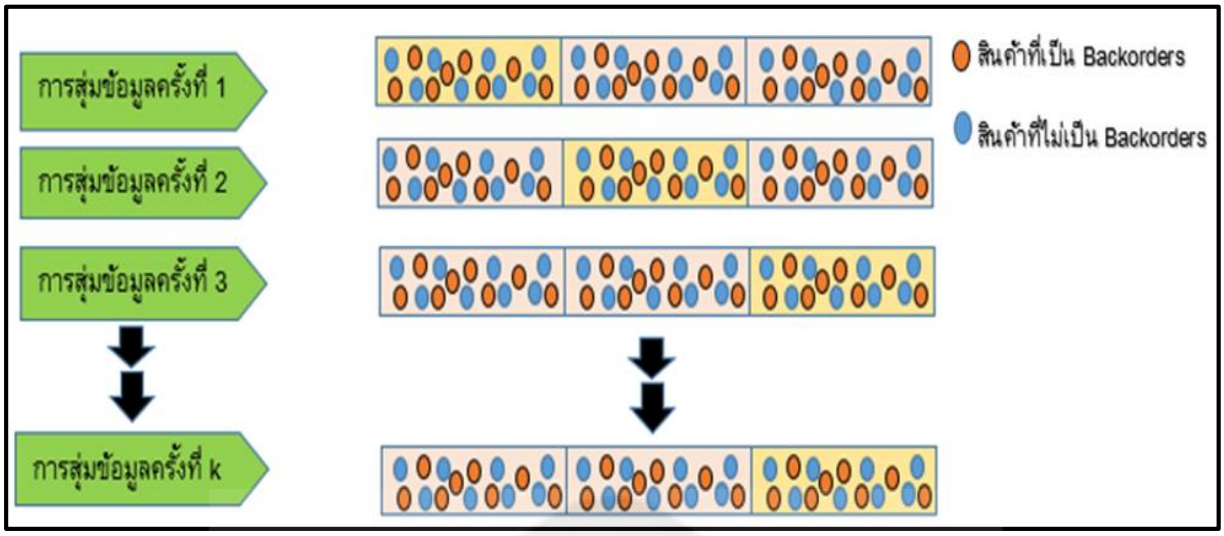
1.3 การสุ่มข้อมูลแบบ K-Fold Cross Validation แบบ Stratified Shuffle Split

เมื่อข้อมูลที่เราสนใจมีสัดส่วนของคลาสที่แตกต่างกันมาก ทำให้เกิดกรณีข้อมูลมีความไม่สมดุลกัน (Imbalanced Problem) การที่จะใช้วิธีการเดิม คือ วิธีสุ่มข้อมูลแบบร้อยละ คือ แบ่งเป็นชุดข้อมูลฝึกฝน (Training data) เป็นร้อยละ 70 และ ข้อมูลชุดทดสอบ (Test data) เป็นร้อยละ 30 จึงไม่สามารถทำได้เพราะอาจทำให้เกิดปัญหาการเกิด Overfitting และ ค่าความถูกต้องในการทำนายมีมากเกินความเป็นจริงและไม่น่าเชื่อถือ [4] วิธีการสุ่มข้อมูลที่เหมาะสมกับข้อมูลที่เป็น Imbalanced อีกวิธีหนึ่งคือ K-Fold Cross Validation แบบ Stratified Shuffle Split เพราะมีหลักการทำงานด้วยการแบ่งข้อมูลออกเป็นหลายส่วนเท่าๆกันโดยแทนค่าด้วย ค่า k แทนด้วยจำนวนครั้งที่ต้องการสุ่มข้อมูล ดังนั้นวิธีนี้จะสุ่มตัวอย่างจากข้อมูลทั้งหมดด้วยการทำซ้ำเพื่อสร้างชุดการฝึกและชุดทดสอบ และทำสลับกับไปมาเพื่อให้ข้อมูลในกลุ่มที่เหมือนกันกระจายไปในส่วนต่างๆให้เกิดความสมดุลและเท่ากันทั้งหมด

ข้อดีของการสุ่มข้อมูล K-Fold Cross Validation แบบ Stratified Shuffle Split คือจะทำให้การสุ่มข้อมูลมีความเที่ยงตรงและลดการเกิด over fitting ได้ดี ดังภาพประกอบ 2



ภาพประกอบ 1 การสุ่มข้อมูลแบบ K-Fold Cross Validation



ภาพประกอบ 2 การสุ่มข้อมูลแบบ K-Fold Cross Validation แบบ Stratified Shuffle Split

1.4 ทฤษฎีพื้นฐานหลักการทำงานของอัลกอริทึมที่ใช้ในการสร้างแบบจำลอง

1.4.1 ทฤษฎีเกี่ยวกับการเรียนรู้แบบ นาอีฟ เบย์ (Naive Bayes)

เป็นอัลกอริทึมที่ใช้ในการจำแนกประเภทข้อมูลเพื่อหาสมมุติฐานที่ดีที่สุดโดยอาศัยหลักความน่าจะเป็นด้วยทฤษฎีของเบย์ (Bayes' Theorem) ซึ่งวิเคราะห์หาความน่าจะเป็นของเหตุการณ์ที่ยังไม่เกิดขึ้นพยากรณ์จากเหตุการณ์ที่เกิดขึ้นมาก่อนโดยใช้สมการความน่าจะเป็นดังนี้ [5]

$$P(C|A)=(P(A|C) \times P(C)) / (P(A)) \quad (1)$$

โดยที่ P แทนค่าความน่าจะเป็นของเหตุการณ์

P(C) ความน่าจะเป็นของสินค้าที่เป็น backorders

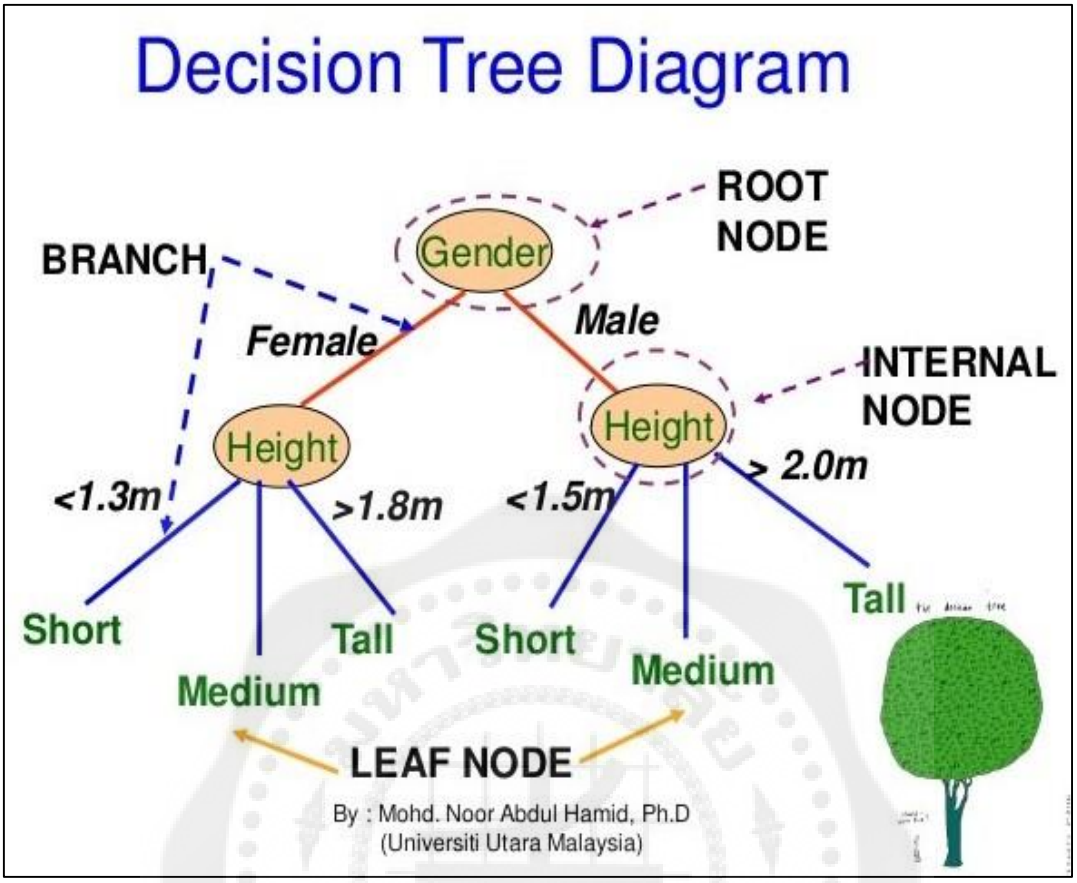
P(A) ความน่าจะเป็นของที่ไม่เป็นสินค้า backorders

P(C|A) คือ ความน่าจะเป็นของ สินค้า backorders เมื่อรู้ สินค้าที่ไม่เป็น backorders

P(A|C) คือ ความน่าจะเป็นของ สินค้าที่ไม่เป็น backorders เมื่อรู้ สินค้าที่เป็น backorders

1.4.2 ทฤษฎีเกี่ยวกับอัลกอริทึมต้นไม้ตัดสินใจ (Decision Tree)

อัลกอริทึมนี้มีรูปแบบการทำงานแบบต้นไม้ที่มีการแบ่งข้อมูลออกเป็นกลุ่มเรียกว่าคลาส (class) ประกอบด้วยคุณสมบัติของข้อมูลเป็นตัวจำแนกประเภท เรียกว่า Attribute กิ่ง (link) ที่ใช้แทนเงื่อนไขในการทดสอบ และโหนดปลาย (leaf node) แทนกลุ่มเป้าหมาย ซึ่งมีลักษณะการค้นหาคำตอบของปัญหาจากบนไปล่าง (Top – Down) โดยเริ่มจากการเลือกคุณสมบัติที่ดีที่สุดเป็นโหนดราก (Root node) และวนสร้างโหนดลูกไปเรื่อยๆจนกว่าข้อมูลจะเป็นกลุ่มเดียวกัน [6] ดังภาพประกอบ 3



ภาพประกอบ 3 ส่วนประกอบของต้นไม้ตัดสินใจ [8]

(ที่มา: Classification Using Decision โดย NoorAbdulHamid Mohd (2015) จาก https://www.slideshare.net/knottisme/classification-using-decision-tree-53984611)

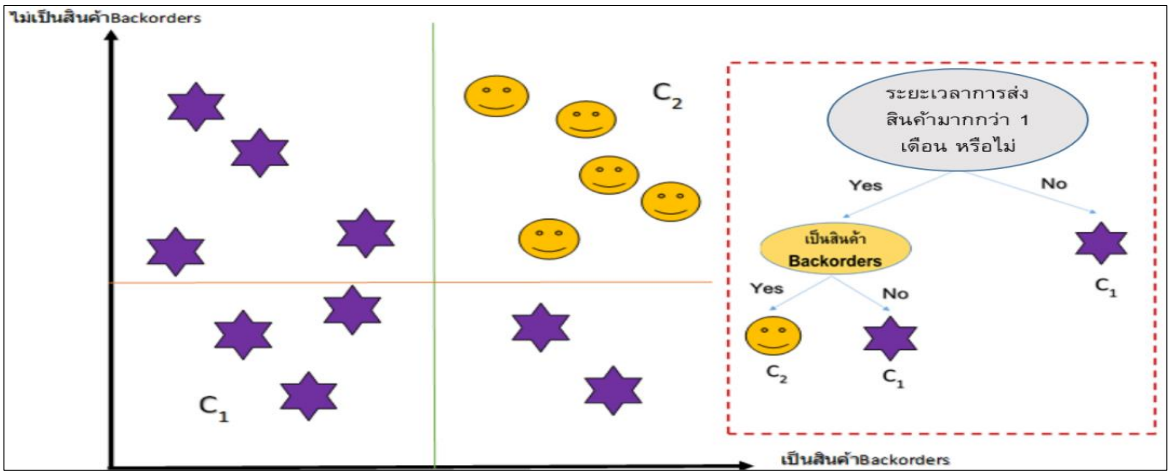
เนื่องจากข้อมูลที่เราต้องการจำแนกประเภทเป็นข้อมูลที่มีความต่อเนื่องกัน (Continuous) โดยเป็นข้อมูลที่เป็นตัวเลขและถูกบันทึกไว้ในช่วงระยะเวลาหนึ่งอย่างต่อเนื่อง ดังนั้น วิธีการแบ่งประเภทของข้อมูลโดยใช้วิธีการสร้างต้นไม้การตัดสินใจสามารถแสดงเป็นขั้นตอนดังนี้

(1) กำหนด Attribute ที่มีต้องการนำมาแบ่งข้อมูล โดยเริ่มจากำหนดให้เป็น Root Node ที่เป็นตัวตั้งต้นของปัญหาและ กำหนดเป้าหมายเป็นผลลัพธ์เรียกว่า Leaf Node

(2) นำคำตอบที่มีความเป็นไปได้ที่จะเกิดขึ้นจาก Attribute ที่ถูกเลือกแตกออกเป็นกลุ่มย่อยๆ

(3) แบ่งข้อมูลตามกลุ่มที่ได้แตกกิ่งออกมาจาก Root Node แบ่งเป็นกลุ่มที่เล็กลงจนกว่าได้คำตอบสุดท้ายที่ต้องการนำไปใช้ในการตัดสินใจ

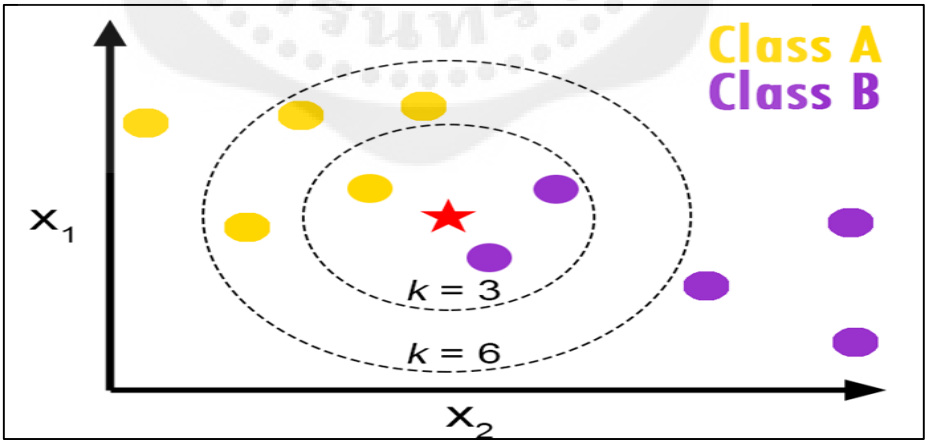
(4) ย้อนกลับไปทำในขั้นตอนแรกเพื่อหา Attribute ที่สำคัญที่สุด ที่นำมาใช้ในการจำแนกประเภทต่อไป ดังภาพประกอบ 4



ภาพประกอบ 4 การทำงานของอัลกอริทึม Decision Tree

1.4.3 ทฤษฎีเกี่ยวกับอัลกอริทึมเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-nearest neighbor)

เป็นเทคนิคที่ใช้ในการจำแนกประเภทของข้อมูลที่ใช้งานง่ายโดยอัลกอริทึมนี้เป็นการจัดกลุ่มข้อมูลที่มีความใกล้เคียงกันโดยพิจารณาจากค่า k ซึ่งหมายถึงจำนวนของข้อมูลที่เรานำมาใช้ในการพิจารณา เช่น ถ้า k = 3 หมายถึง เราจะพิจารณาข้อมูลจำนวน 3 จุดที่อยู่ใกล้เคียงกับจุดที่ต้องการพิจารณา (โดยที่ค่า k ควรกำหนดเป็นเลขคี่เสมอ) เพราะไม่เช่นนั้นจะทำให้ไม่สามารถแบ่งข้อมูลออกเป็นกลุ่มได้อย่างชัดเจน [9]



ภาพประกอบ 5 วิธีการจำแนกแบบ K-nearest neighbor [10]

(ที่มา: Classification Using K-Nearest Neighbor in R โดย Morgun Ivan (2017) จาก http://en.proft.me/2017/01/22/classification-using-k-nearest-neighbors-r)

1.4.4 ทฤษฎีเกี่ยวกับอัลกอริทึมที่นำการเรียนรู้ที่หลายแบบมาฝึกฝนร่วมกัน (Ensemble Learning)

อัลกอริทึมนี้จะนำข้อดีของแต่ละแบบมาทดแทนข้อด้อยของอีกแบบทำให้ได้ผลการทำนายดีกว่าการใช้การเรียนรู้แบบเดียว ได้แก่

1.4.4.1 การจำแนกแบบป่าสุ่ม (Random Forest)

เป็นเทคนิคที่ทำการสุ่มเลือกกลุ่มของชุดข้อมูลฝึกฝนขึ้นหลายๆครั้ง ซึ่งในแต่ละครั้งอาจซ้ำหรือไม่ซ้ำกันก็ได้เรียกการสุ่มแบบนี้ว่า “การสุ่มแบบบู๊ตสแตร็ป” (bootstrap) เมื่อสุ่มแล้วนำกลุ่มข้อมูลที่สุ่มมาสร้างต้นไม้การตัดสินใจได้หลายต้น (Multiple Decision Trees) จากหลายกลุ่ม ในแต่ละต้นจะถูกสร้างขึ้นอย่างเป็นอิสระต่อกันและแสดงผลลัพธ์ของตนออกมา จากนั้นในขั้นตอนสุดท้ายจะทำการรวมว่าผลลัพธ์การทำนายคำตอบใดถูกโหวตมากที่สุดจึงเลือกคำตอบนั้นมาเป็นคำตอบสุดท้าย Random Forest เป็นอัลกอริทึมที่ทำงานได้ดีกับข้อมูลที่มีลักษณะไม่สมดุล (Imbalanced) เนื่องจากการทำงานของอัลกอริทึมจะมีการแบ่งส่วนของในการทำงานทำให้ค่าความผิดพลาดไม่น้อยกว่าการพยากรณ์แบบอื่น

ตัวอย่างค่าพารามิเตอร์ที่ในการสร้างแบบจำลองค่าพารามิเตอร์ตามค่าตั้งต้นในรูปแบบของ scikit-learn แบบ Random Forest ในงานวิจัยนี้ คือ

- n_estimators = 10 คือ จำนวนต้นไม้ทั้งหมดที่ใช้ในตัวทำนายจำนวน 10 ต้น โดยทั่วไปยิ่งมีค่ามากยิ่งทำให้การทำนายมีความแม่นยำ แต่ก็ใช้ระยะเวลาในการคำนวณนานด้วย [11]
- min_samples_split คือ จำนวนตัวอย่างที่น้อยที่สุดที่ต้องการแบ่ง
- criterion คือ เกณฑ์ในการวัดการกระจายของข้อมูล ซึ่งในค่าตั้งต้นกำหนดให้ใช้ gini เป็นเกณฑ์
- max_features คือ การกำหนดค่าของจำนวน Features ที่ใช้ในการแบ่งต้นไม้ในการสร้างแบบจำลอง

1.4.4.2 การจำแนกแบบ Adaptive Boosting Algorithm หรือ AdaBoost

เป็นอัลกอริทึมที่อาศัยเทคนิคการรวมกลุ่มเพื่อช่วยเพิ่มประสิทธิภาพในการสร้างแบบจำลองด้วยการกำหนดน้ำหนัก (Weight) ให้กับข้อมูลชุดฝึกฝน (Training data) โดยในแต่ละรอบที่มีการสร้างแบบจำลอง ค่าของน้ำหนักจะถูกปรับเปลี่ยนไป [12] ในงานวิจัยนี้จะมีการกำหนดค่าพารามิเตอร์ตามค่าตั้งต้นในรูปแบบของ scikit-learn ดังนี้

- n_estimators = 50 คือ จำนวนที่ใช้ในการ Boost จำนวน 50 ตัว
- Algorithm = 'SAMME.R' คือ Algorithm ที่ใช้ในการ boost

1.4.4.3 การจำแนกแบบ Extreme Gradient Boosting หรือ XGBoost

เป็นอัลกอริทึมในกลุ่มของการเรียนรู้แบบมีผู้สอน (supervised learning) ที่ใช้กระบวนการในการ Boosting และใช้กรอบแนวคิดเดียวกับ Gradient boosting โดยในการสร้างแบบจำลองขึ้นใหม่ในทุกๆครั้ง จะมีการแก้ไขและปรับปรุงข้อบกพร่องของแบบจำลองก่อนหน้าและ

นำมาปรับใช้กับแบบจำลองใหม่ให้ได้แบบจำลองที่ดีกว่าเดิมเพื่อปรับปรุงแบบจำลองให้ได้ผลการทำนายที่แม่นยำที่สุด จนได้ผลลัพธ์ของคำตอบสุดท้ายที่ดีที่สุด คล้ายกับอัลกอริทึม AdaBoost แต่มีความยืดหยุ่นกว่า ใช้ระยะเวลาน้อยกว่าและเกิด overfitting น้อยกว่า [13] โดยในงานวิจัยนี้จะมีการกำหนดค่าพารามิเตอร์ตามค่าตั้งต้นในรูปแบบของ scikit-learn ดังนี้

- max_depth คือ พารามิเตอร์ที่กำหนดค่าความลึกในการสร้างต้นไม้เท่ากับ 3
- n_estimators คือจำนวนต้นไม้ของปัญหาให้เท่ากับ 100 ต้น
- learning_rate คือ อัตราในการเรียนรู้ หากมีค่าสูงเกินไปจะทำให้แบบจำลองไม่มีประสิทธิภาพ และถ้าต่ำเกินไปก็จะใช้เวลานานแต่จะได้ประสิทธิภาพมากกว่า ซึ่งค่าตั้งต้นปกติจะมีค่า 0.1 โดยทั่วไปมักจะกำหนดค่า Learning Rate ไม่เกิน 1

2. งานวิจัยที่เกี่ยวข้อง

Jiawei Han และคณะ (2011) ได้บรรยายถึงแนวคิดในการวิเคราะห์ข้อมูลด้วยการใช้เทคนิคของเหมืองข้อมูล (Data mining) ได้แก่ การเตรียมข้อมูล ปัญหาข้อมูลมีความไม่สมดุลกัน (Class Imbalanced Problems) หลักการจำแนกข้อมูลและขั้นตอนวิธีในการจำแนกข้อมูล อัลกอริทึมต่างๆที่นำมาใช้ในการสร้างแบบจำลองในการจำแนก รวมไปถึงการประเมินประสิทธิภาพของตัวจำแนก [2]

Leo Breiman (2001) ได้ทำการอธิบายเกี่ยวกับการทำงานของอัลกอริทึม Random Forest เพื่อใช้ในการจำแนกประเภทของข้อมูลตามหลักการของเทคนิคการเรียนรู้ของเครื่อง โดยได้ทำการเผยแพร่ไว้ในเว็บไซต์ http://scikit-learn.org ซึ่งได้รวบรวมคำสั่งต่างๆ (code) คำอธิบายความหมายของพารามิเตอร์ที่ใช้ในการทำนาย รวมไปถึงตัวอย่างของข้อมูลที่ทำนายด้วย Random Forest [11]

Pasapitch Chujai (2015) ได้ทำการศึกษาเกี่ยวกับเทคนิคการเรียนรู้ของเครื่องโดยเน้นศึกษากรณีของข้อมูลที่มีความไม่สมดุลกันซึ่งเป็นข้อมูลที่มีแนวโน้มไปยังทิศทางใดทิศทางหนึ่งมากเกินไป ผู้วิจัยใช้อัลกอริทึมใหม่ชื่อว่า Ensemble Learning with Decision Tree Visualization ร่วมกับอัลกอริทึมโครงสร้างต้นไม้ตัดสินใจ (Decision Tree) ซึ่งมีการเตรียมข้อมูลและปรับลดข้อมูลด้วย K-fold cross validation ซึ่งผลที่ได้พบว่าการใช้ K-fold cross validation สามารถช่วยแก้ปัญหาข้อมูลที่ไม่สมดุลได้ และพบว่าการใช้วิธีการ Ensemble Learning แบบ Boosting ให้ประสิทธิภาพประกอบดีกว่าแบบ Bagging และใช้ได้ดีในกรณีที่ข้อมูลมีความไม่สมดุล [14]

Dred Law (2017) ได้ศึกษาการทำนายสินค้าที่เป็น backorders จากโจทย์ของการแข่งขันในเว็บไซต์ https://www.kaggle.com ด้วยการใช้การเรียนรู้ของเครื่องได้ทำการจัดการกับข้อมูลที่ไม่สมบูรณ์หรือเป็นข้อมูลที่เป็นค่าว่างก่อน แล้วเขียนคำสั่งแสดงความสัมพันธ์ระหว่างคลาสต่างๆออกมาในรูปของกราฟรูปแบบต่างๆ จากนั้นทดสอบประสิทธิภาพในการจำแนกประเภทด้วยการสร้างแบบจำลองโดยใช้อัลกอริทึม 3 แบบ ได้แก่ K-nearest neighbor Random Forest และ Support Vector Machine มาเปรียบเทียบกัน โดยใช้ตัวชี้วัดคือ ค่าความถูกต้อง (Accuracy) เพียงอย่างเดียวผลที่ได้คือ Random Forest มีค่าความถูกต้องมากที่สุดคิดเป็นร้อยละ 98.89 รองลงมาเป็น Support Vector

Machine คิดเป็นร้อยละ 98.88 และลำดับสุดท้ายคือ K-nearest neighbor มีค่าความถูกต้องที่ร้อยละ 98.84 [15]

Kueipo Huang (2017) ได้ศึกษาการจำแนกประเภทของสินค้าที่เป็น backorders จากโจทย์ของการแข่งขันในเว็บไซต์ https://www.kaggle.com โดยใช้เทคนิคการเรียนรู้ของเครื่อง โดยเน้นศึกษาจากปัญหาชุดข้อมูลไม่สมดุลกัน กล่าวว่าในบางกรณีไม่สามารถวัดประสิทธิภาพการแบ่งประเภทของข้อมูลจากตัวชี้เพียงหนึ่งตัวได้ต้องเพิ่มขั้นตอนการแบ่งข้อมูล Training Data ให้อยู่ในอัตราส่วนที่เหมาะสมก่อนนำมาสร้างแบบจำลองด้วยการใช้ K-Fold Cross Validation แบบ Stratified Shuffle Split ซึ่งเหมาะสมกับข้อมูลที่มีแนวโน้มไปทางใดทางหนึ่งสูงกว่า และนอกจากนี้ตัวชี้วัดที่นำมาทดสอบอัลกอริทึม Random Forest คือ ค่าความถูกต้องโดยกำหนดค่าพารามิเตอร์ MaxDepth ที่แตกต่างกันนำมาเปรียบเทียบกันบน ROC Curve โดยผลการศึกษาพบว่า Random Forest เป็นอัลกอริทึมที่มีค่าความถูกต้องมากที่สุดคิดเป็นร้อยละ 94.70 และค่า MaxDepth ที่ทำให้ค่า ROC Curve ที่มากที่สุดคือ MaxDepth เท่ากับ 10 แสดงผลลัพธ์ของ ROC Curve (AUC) เท่ากับ 0.9475 [4]

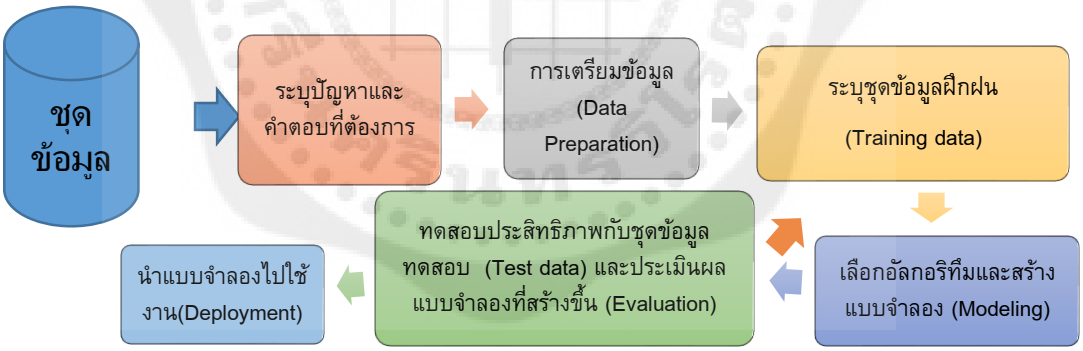
Rodrigo Santis และคณะ (2017) ได้ทำการศึกษางานวิจัยเรื่อง “Predicting Material backorders in Inventory Management using Machine Learning” ซึ่งเป็นงานวิจัยที่เกี่ยวข้องกับปัญหาสินค้า backorders ที่เป็นปัญหาที่เกิดขึ้นในธุรกิจส่งผลต่อระบบการจัดการสินค้าคงคลังและเป็นตัวระบุความขาดแคลนสินค้าและ การสร้างโอกาสทางธุรกิจ ในงานวิจัยนี้จะอธิบายถึงการใช้การเรียนรู้ของเครื่องโดยอาศัยจำแนกประเภทในการสร้างแบบจำลองในกรณีของข้อมูลที่ไม่สมดุลและทดสอบประสิทธิภาพด้วยตัวชี้วัด ROC Curve และ กราฟแสดง Precision-recall อีกทั้งยังกล่าวถึงการเรียนรู้ด้วยเทคนิค Ensemble ซึ่งเป็นการรวมการเรียนรู้หลายประเภทไว้ด้วยกันและเป็นอิสระต่อกันเพื่อเพิ่มประสิทธิภาพของโมเดลในการจำแนกอีกด้วยจากงานวิจัยนี้พบว่าประสิทธิภาพการจำแนกของอัลกอริทึม Logistic Regression มีค่าความถูกต้องต่ำที่สุดคิดเป็นร้อยละ 92.06 และอัลกอริทึมที่มีค่าความถูกต้องสูงที่สุดคือ Gradient Tree Boosting คิดเป็นร้อยละ 94.78 โดยงานวิจัยที่นำมาอ้างอิงนี้จะแตกต่างกับงานวิจัยของเราตรงที่งานวิจัยเรื่องนี้ไม่ได้นำอัลกอริทึม K-nearest neighbor และ Naive Bayes มาวัดผลด้วย [16]

Yuqi Li (2017) ได้ทำการวิจัยเรื่อง “Backorders Prediction Using Machine Learning for Danish Craft Beer Breweries” ได้ทำการศึกษาการใช้การเรียนรู้ของเครื่องในการทำนายข้อมูลของสินค้าประเภทเบียร์ประเทศเดนมาร์กที่จะเกิดสถานการณ์สินค้า backorders และต้องการจัดการกับปัญหาความขาดแคลนสินค้าซึ่งศึกษาจากข้อมูลการผลิตเบียร์ Danish ในอดีต นอกจากนี้ยังศึกษาการจัดการกับข้อมูลที่ไม่สมดุลโดยเริ่มจาก การเตรียมข้อมูล (Data preprocessing) ด้วยการทำข้อมูลให้เป็นกลาง (Data normalization)การกำจัดค่าว่างหรือข้อมูลที่หายไป (Missing Value) นำไปสู่ขั้นตอนการสร้างแบบจำลอง (Modeling) ผลการศึกษาพบว่า Support Vector Machine เป็นอัลกอริทึมที่ให้ผลดีที่สุดเมื่อเทียบกับการจำแนกแบบอื่น [9]

บทที่ 3
วิธีดำเนินการวิจัย

การวิจัยการทำนายสินค้าคงคลัง โดยใช้เทคนิคเหมืองข้อมูล (กรณีศึกษา: สินค้าค้างส่ง) เป็นงานวิจัยเชิงประยุกต์ (Applied Research) โดยประยุกต์ใช้กับข้อมูลตัวอย่างของสินค้าในคลังสินค้าแห่งหนึ่งเพื่อให้การตรวจสอบและจำแนกประเภทสินค้าให้มีความถูกต้องและแม่นยำยิ่งขึ้น ซึ่งผู้วิจัยเริ่มดำเนินการวิจัยด้วยการค้นคว้าบทความและงานวิจัยที่เกี่ยวข้องกับกระบวนการเรียนรู้ของเครื่องในรูปแบบการเรียนรู้แบบมีผู้สอน ที่อาศัยกฎการจำแนกประเภทเป็นหลัก โดยภาษาโปรแกรมที่เลือกใช้ในการศึกษาคือภาษา Python เวอร์ชั่น 3 และ ใช้เครื่องมือที่ใช้ในการประมวลคำสั่งที่ชื่อว่า Jupyter notebook ที่อยู่ในรูปแบบของ Browser ที่ชื่อว่า ‘https://csit.swu.ac.th” เพราะใช้งานง่าย และสามารถติดตั้งไลบารี (library) ที่สำคัญสำหรับแสดงข้อมูลได้หลากหลาย เช่น การสร้างกราฟรูปแบบต่างๆ นอกจากนี้ยังใช้ไลบารีที่มีฟังก์ชันใช้สำหรับวัดประสิทธิภาพของแบบจำลองที่ชื่อว่า Scikit-learn ซึ่งเป็น ไลบารีที่นิยมใช้ในการเรียนรู้ของเครื่องประกอบกับไลบารีที่ใช้ประมวลผลข้อมูล ได้แก่ Numpy Pandas Matplotlib Seaborn เป็นต้น

กระบวนการการเรียนรู้ของเครื่อง แบ่งออกเป็น 6 ส่วนสามารถแสดงเป็นแผนภาพได้ดังนี้



ภาพประกอบ 6 กระบวนการการเรียนรู้ของเครื่อง (Machine Learning process)

1. ส่วนที่ 1 ส่วนของชุดข้อมูล (Dataset)

1.1 ประเภทและคุณลักษณะของข้อมูล

ข้อมูลที่ใช้ในการศึกษาชุดนี้เป็นชุดข้อมูลที่ถูกเผยแพร่โดย www.kaggle.com ซึ่งเป็นข้อมูลการขายรายสัปดาห์ของสินค้าประเภทหนึ่งประกอบไปด้วย 23 แอตทริบิวต์ มีจำนวน 1,687,861 รายการ (entries) แสดงข้อมูล 2 ประเภท คือ ข้อมูลที่เป็นตัวเลขที่มีความยาว 64 บิต (float64) และ ข้อมูลที่เป็นออบเจ็กต์ (Object) มีลักษณะข้อมูลเป็นฐานสอง (Binary) 6

แอตทริบิวต์ ได้แก่ deck_risk oe_constraint potential_issue ppap_risk rev_stop stop_auto_buy และ went_on_backorders ที่มีคำตอบเพียงสองค่าคือ ใช่หรือไม่ใช่ มีแอตทริบิวต์ที่เป็นตัวเลข (Numeric) 16 แอตทริบิวต์ ซึ่งมีข้อมูลที่เป็นตัวเลขมีลักษณะเป็นแบบข้อมูลอัตราส่วนแบบต่อเนื่อง (Ratio/Continuous) ซึ่งได้แก่แอตทริบิวต์ดังต่อไปนี้ forecast_3_month forecast_6_month forecast_9_month in_transit_qty local_bo_qty min_bank national_inv perf_12_month_avg perf_6_month_avg pieces_past_due sales_1_month sales_3_month sales_6_month และ lead_time ส่วนของแอตทริบิวต์ที่ใช้แทนชื่อของสินค้า (ID) คือ sku ลักษณะข้อมูลเป็นเชิงลักษณะแบบไม่ต่อเนื่อง (Nominal/ Discrete) มีรายละเอียดตาม ตาราง 1 และ ตาราง 2 ดังนี้

ตาราง 1 แสดงข้อมูลประเภท float64

ชื่อคอลัมน์	ลักษณะของข้อมูล	ประเภทของข้อมูล	ความหมายของคอลัมน์
forecast_3_month	Ratio/Continuous	float64	ยอดขายที่พยากรณ์ช่วง 3 เดือน
forecast_6_month	Ratio/Continuous	float64	ยอดขายที่พยากรณ์ช่วง 6 เดือน
forecast_9_month	Ratio/Continuous	float64	ยอดขายที่พยากรณ์ช่วง 9 เดือน
in_transit_qty	Ratio/Continuous	float64	ปริมาณการขนส่งสินค้าจากแหล่งที่มา
lead_time	Ratio/Continuous	float64	ระยะเวลาในการขนส่งสินค้า
local_bo_qty	Ratio/Continuous	float64	จำนวนรวมของคำสั่งซื้อที่ค้างส่งลูกค้า
min_bank	Ratio/Continuous	float64	ระดับสต็อคขั้นต่ำที่ควรสำรองไว้
national_inv	Ratio/Continuous	float64	ระดับของสินค้าที่มี ณ ปัจจุบัน
perf_6_month_avg	Ratio/Continuous	float64	ประสิทธิภาพการส่งสินค้าจากผู้จัดหาย้อนหลัง 12 เดือน
perf_12_month_avg	Ratio/Continuous	float64	ประสิทธิภาพการส่งสินค้าจากผู้จัดหา ย้อนหลัง 6 เดือน
pieces_past_due	Ratio/Continuous	float64	จำนวนของสินค้าที่ค้างส่งจากผู้จัดหาสินค้า
sales_1_month	Ratio/Continuous	float64	ยอดขายย้อนหลัง 1 เดือน
sales_3_month	Ratio/Continuous	float64	ยอดขายย้อนหลัง 3 เดือน
sales_6_month	Ratio/Continuous	float64	ยอดขายย้อนหลัง 6 เดือน
sales_9_month	Ratio/Continuous	float64	ยอดขายย้อนหลัง 9 เดือน

ตาราง 2 แสดงข้อมูลประเภท object

ชื่อคอลัมน์	ลักษณะของข้อมูล	ประเภทของข้อมูล	ความหมายของคอลัมน์
deck_risk	Binary (0,1)	object	มี/ไม่มี ความเสี่ยงเรื่องการประกันสินค้า
oe_constraint	Binary (0,1)	object	มี/ไม่มี ความเสี่ยงในเรื่องข้อจำกัด
potential_issue	Binary (0,1)	object	มี/ไม่มี แหล่งที่มาของสินค้า
ppap_risk	Binary (0,1)	object	มี/ไม่มี ความเสี่ยงในกระบวนการผลิต (Production part approval process)
rev_stop	Binary (0,1)	object	มี/ไม่มี ความเสี่ยงต่อการหยุดการผลิต
sku	Nominal/Discrete	object	Random ID “ID รหัสของสินค้า"
stop_auto_buy	Binary (0,1)	object	มี/ไม่มี ความเสี่ยงต่อการหยุดขายสินค้า
went_on_backorders	Binary (0,1)	object	มี/ไม่มี สินค้าที่เป็น backorders (เป็น target value ที่ต้องการหา)

ในการศึกษาการเรียนรู้ของเครื่องตัวแปรที่มีความสำคัญของการที่จะใช้ในการสร้างแบบจำลองคือFeatures และ Labels ซึ่ง Features ที่เราใช้ในการศึกษาได้แก่ forecast_3_month forecast_6 _month forecast_9_month in_transit_qty local_bo_qty min_bank national_inv perf_6_month_avg perf_12_month_avg pieces_past_due sales_1_month sales_3_month sales_6_month sales_9_month lead_time deck_risk oe_constraint potential_issue ppap_risk rev_stop stop_auto_buy ส่วน sku เป็น Features ที่ไม่เกี่ยวข้องเพราะเป็น รหัสของสินค้าที่สุ่มขึ้นมาจึงไม่นำเข้ามาใช้เป็น Features ในการสร้างแบบจำลอง และให้ Labels ที่ใช้ในการทำนายเป็น went_on_backorders

1.2 การสำรวจค่าสูงสุด-ค่าต่ำสุด ค่ากลาง และค่าเบี่ยงเบนมาตรฐานของ Features (Features Exploration)

เมื่อทำการสำรวจ Features ด้วยคำสั่ง .describe() เพื่อสำรวจค่าสูงสุด (Max) ค่าต่ำสุด (Min) ค่ากลาง (Mean) รวมไปถึงค่าเบี่ยงเบนมาตรฐาน (Std) ของแต่ละ Features จากจำนวนของสินค้าทั้งหมด 1,687,860 หน่วย สามารถแสดงได้ดังภาพประกอบ 7

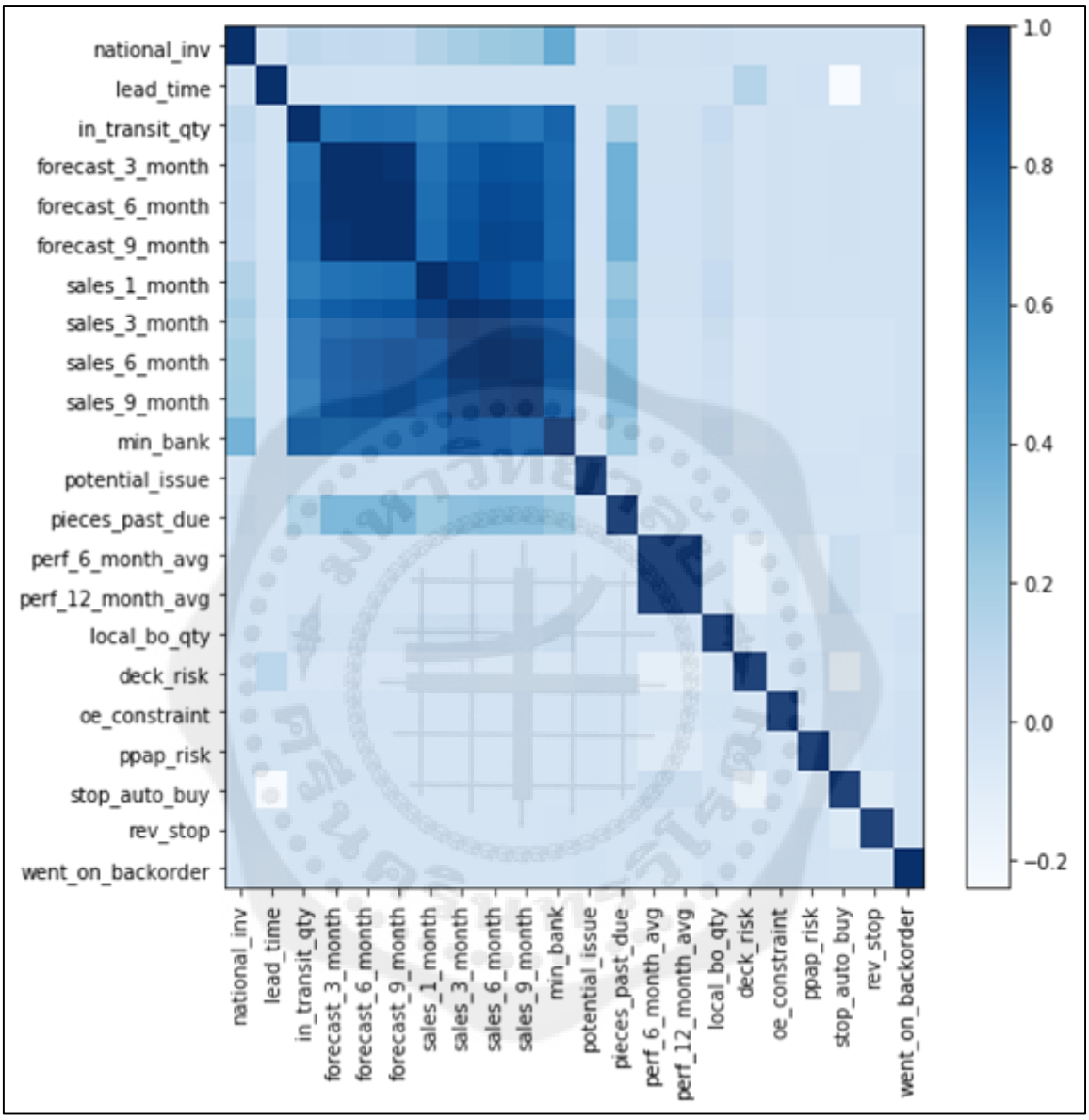
Features	count	mean	std	min	max
forecast_3_month	1,687,860.00	178.12	5,026.55	0.00	1,427,612.00
forecast_6_month	1,687,860.00	344.99	9,795.15	0.00	2,461,360.00
forecast_9_month	1,687,860.00	506.36	14,378.92	0.00	3,777,304.00
in_transit_qty	1,687,860.00	44.05	1,342.74	0.00	489,408.00
local_bo_qty	1,687,860.00	0.63	33.72	0.00	12,530.00
min_bank	1,687,860.00	52.77	1,254.98	0.00	313,319.00
national_inv	1,687,860.00	496.11	29,615.23	(27,256.00)	12,334,404.00
perf_12_month_avg	1,687,860.00	0.79	0.23	0.00	1.00
perf_6_month_avg	1,687,860.00	0.79	0.23	0.00	1.00
pieces_past_due	1,687,860.00	2.04	236.02	0.00	146,496.00
sales_1_month	1,687,860.00	55.93	1,928.20	0.00	741,774.00
sales_3_month	1,687,860.00	175.03	5,192.38	0.00	1,105,478.00
sales_6_month	1,687,860.00	341.73	9,613.17	0.00	2,146,625.00
sales_9_month	1,687,860.00	525.27	14,838.61	0.00	3,205,172.00
lead_time	1,687,860.00	7.88	6.84	0.00	52.00
deck_risk	1,687,860.00	0.23	0.42	0.00	1.00
oe_constraint	1,687,860.00	0.00	0.01	0.00	1.00
potential_issue	1,687,860.00	0.00	0.02	0.00	1.00
ppap_risk	1,687,860.00	0.12	0.33	0.00	1.00
rev_stop	1,687,860.00	0.00	0.02	0.00	1.00
stop_auto_buy	1,687,860.00	0.96	0.19	0.00	1.00

ภาพประกอบ 7 ผลลัพธ์ของการสำรวจ Features

1.3 การศึกษาความสัมพันธ์ระหว่าง Features

เนื่องจากผู้วิจัยต้องการตรวจสอบความสัมพันธ์ของแอตทริบิวต์ทั้งหมดที่ใช้ในการศึกษาว่ามีความสัมพันธ์ไปในทิศทางเดียวกันหรือไม่ โดยให้ทุกแอตทริบิวต์เป็นทั้ง Features และ Labels และแสดงออกมาในรูปของเมทริกซ์แสดงความสัมพันธ์ (Correlation matrix) ให้กราฟที่แสดงเป็นสีทึบแสดงถึงความสัมพันธ์ที่ไปในทิศทางเดียวกัน ส่วน กราฟที่แสดงสีอ่อนลงมามีความสัมพันธ์ไปในทิศทางตรงกันข้าม ดังนั้นถ้า ค่า Correlation มีค่าเป็นบวกมากแสดงว่ามีความสัมพันธ์กันมาก และมีค่าน้อยลงเมื่อมีความสัมพันธ์ต่อกันน้อยลงไปตามลำดับ และเมื่อค่า Correlation แสดงค่าที่ติดลบแสดงว่าความสัมพันธ์ของ Features นั้นๆ ไม่มีความสัมพันธ์กันเลยหรือมีความสัมพันธ์กันในทิศทางตรงกันข้าม ตัวอย่างเช่น ปริมาณการขนส่ง (in_transit_qty) ยอดขายที่พยากรณ์ในช่วง 3 เดือน 6 เดือนและ 9 เดือน (forecast) และยอดขายช่วง 1 เดือน 3 เดือน 6 เดือนและ 9 เดือน (sales) รวมไปถึงระดับสต๊อคขั้นต่ำที่ควรสำรองไว้ (min_bank) มีความสัมพันธ์กันมากเพราะค่าความสัมพันธ์มีค่าเป็นบวก และเมื่อพิจารณาจาก Features ระยะเวลาในการขนส่งสินค้า (lead_time) กับ ความเสี่ยงต่อการหยุดขายสินค้า (stop_auto_buy)

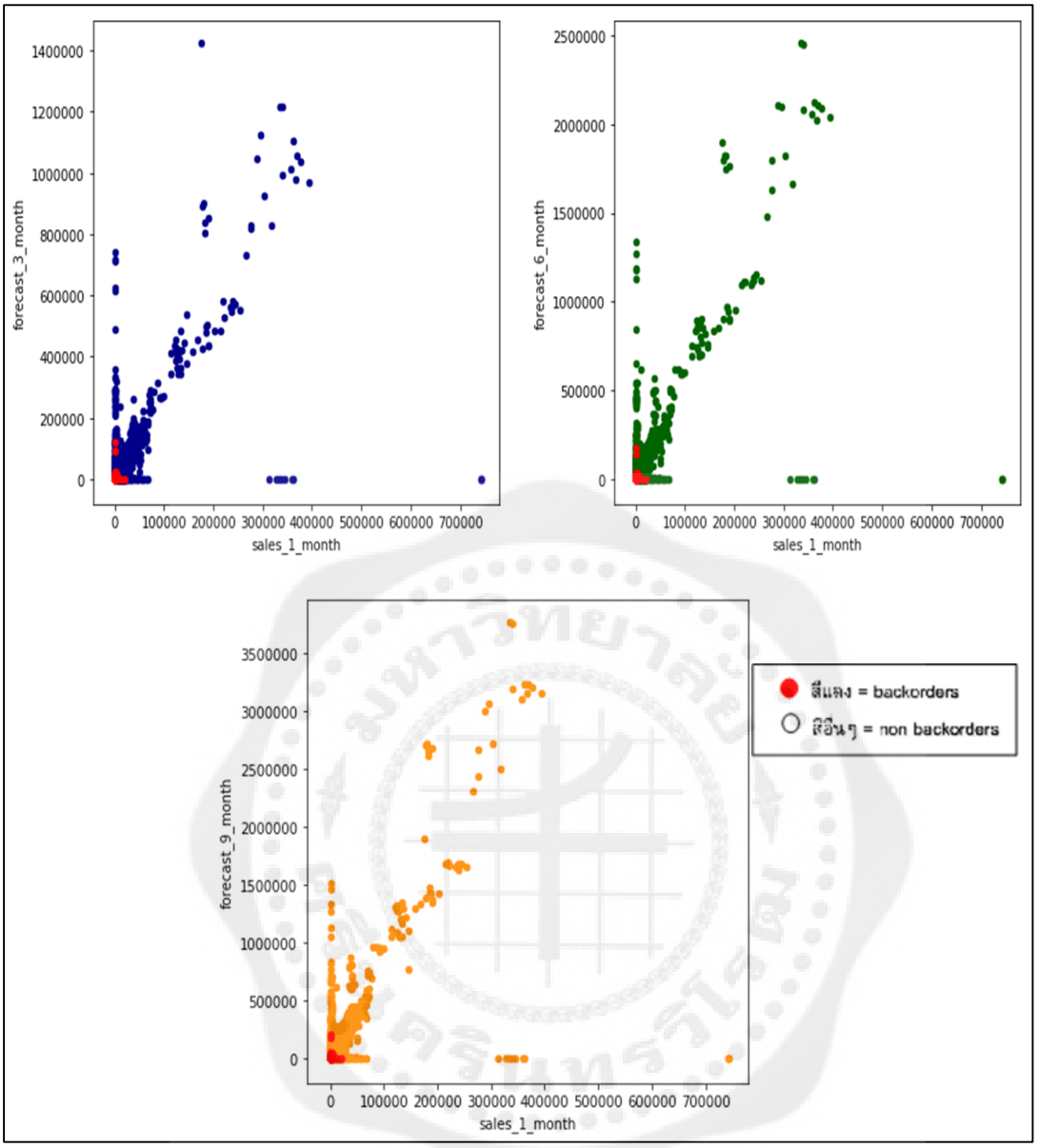
ค่าที่แสดงถึงมีความสัมพันธ์กันในเป็นลบ แสดงให้เห็นว่าทั้ง ระยะเวลาในการขนส่งสินค้ากับ ความเสี่ยงต่อการหยุดขายสินค้า มีความสัมพันธ์กันในทิศทางตรงกันข้าม ดังภาพประกอบ 9



ภาพประกอบ 8 เมทริกซ์แสดงความสัมพันธ์ของแอตทริบิวต์ในการทำนาย (Correlation matrix)

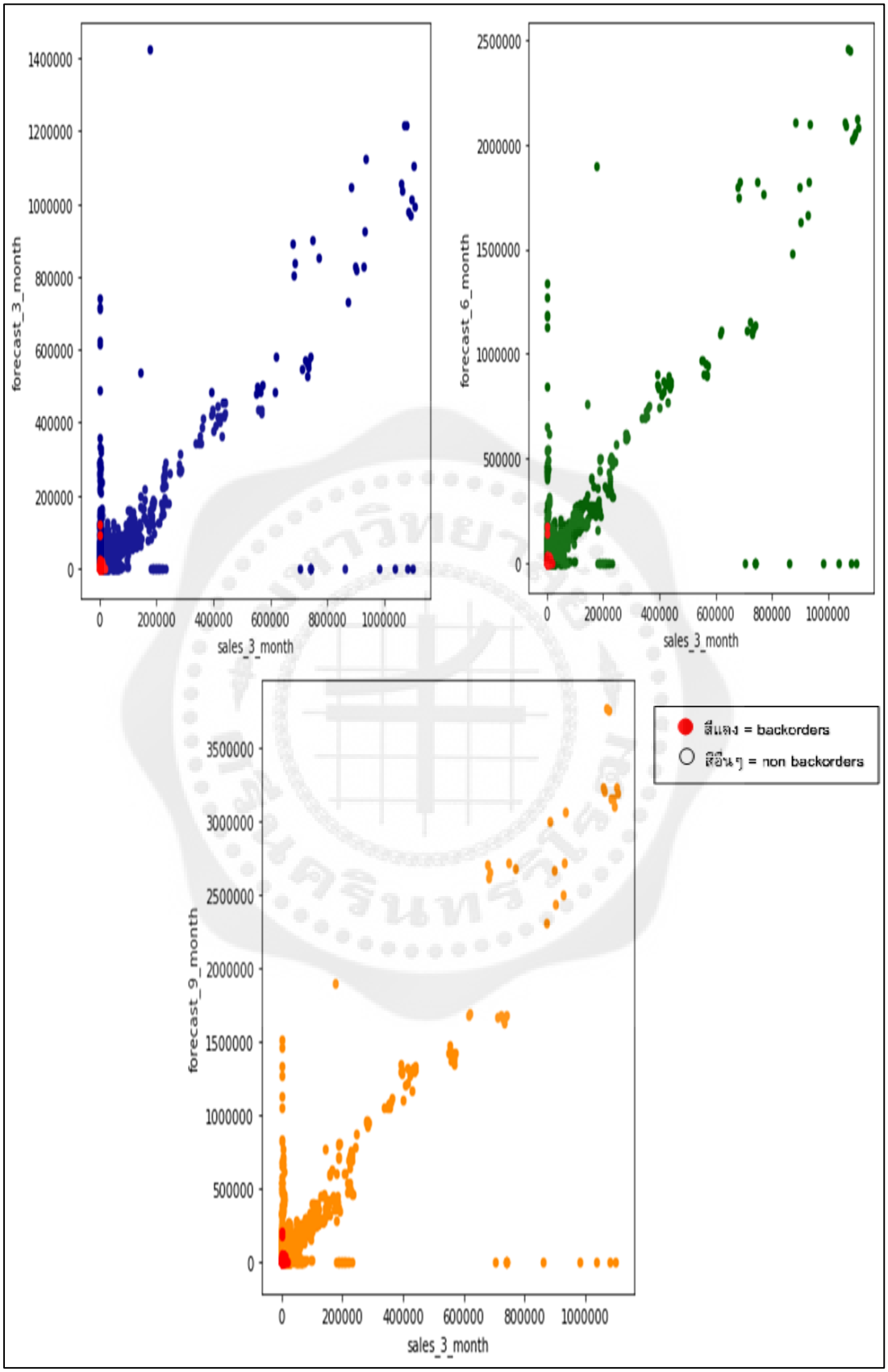
1.4 การศึกษาความสัมพันธ์ระหว่างสอง Features คือ ยอดขายจริง (sale) และยอดขายที่พยากรณ์ (forecast)

เมื่อทำการหาความสัมพันธ์ของกราฟในรูปแบบของแผนภาพการกระจาย (Scatter Plot) ระหว่าง Features forecast_3_month forecast_6_month และforecast_9_month เปรียบเทียบกับ sales_1_month sales_3_month sales_6_month และ sales_9_month โดยให้

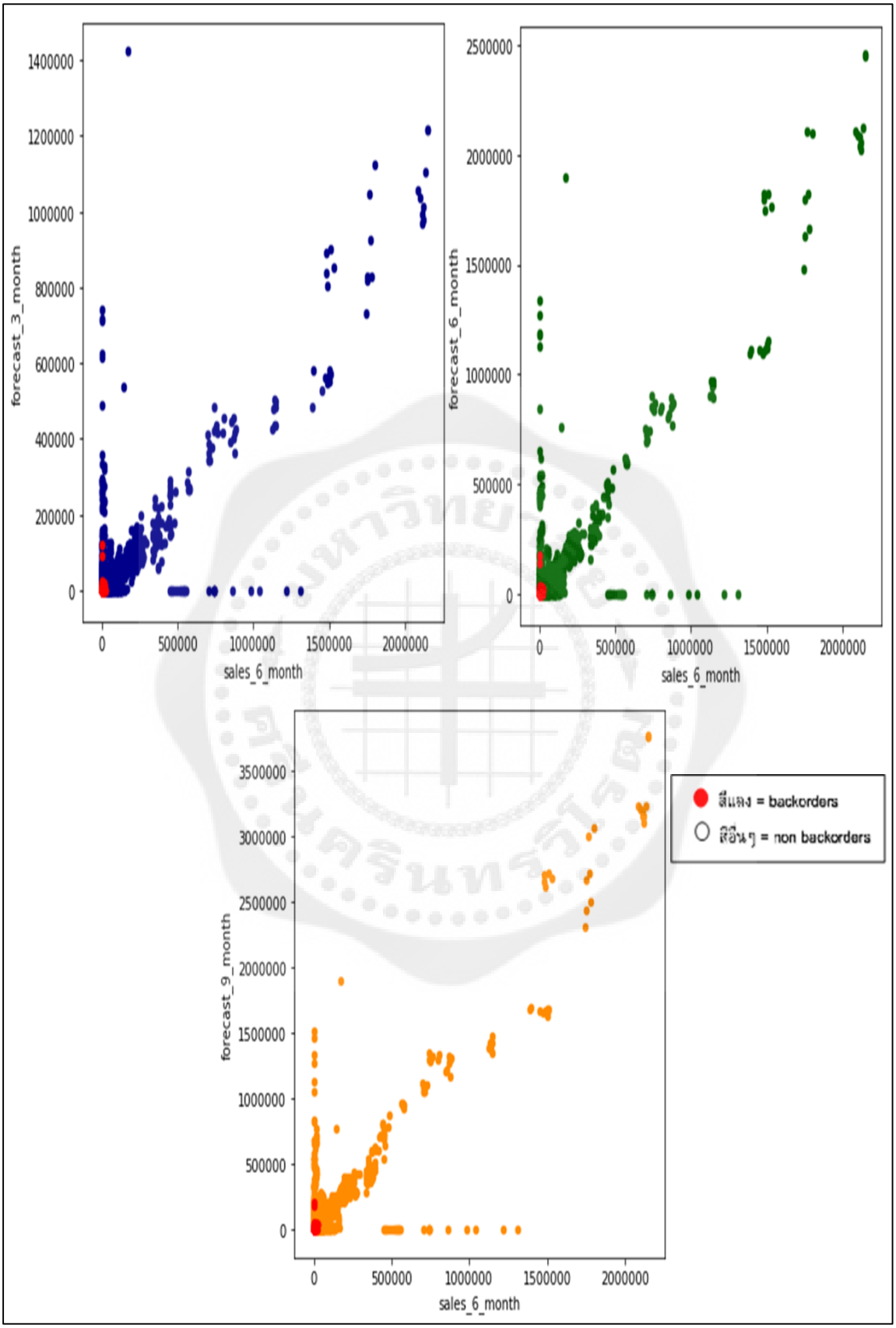


ภาพประกอบ 9 ความสัมพันธ์ระหว่าง Features sales_1_month และ forecast

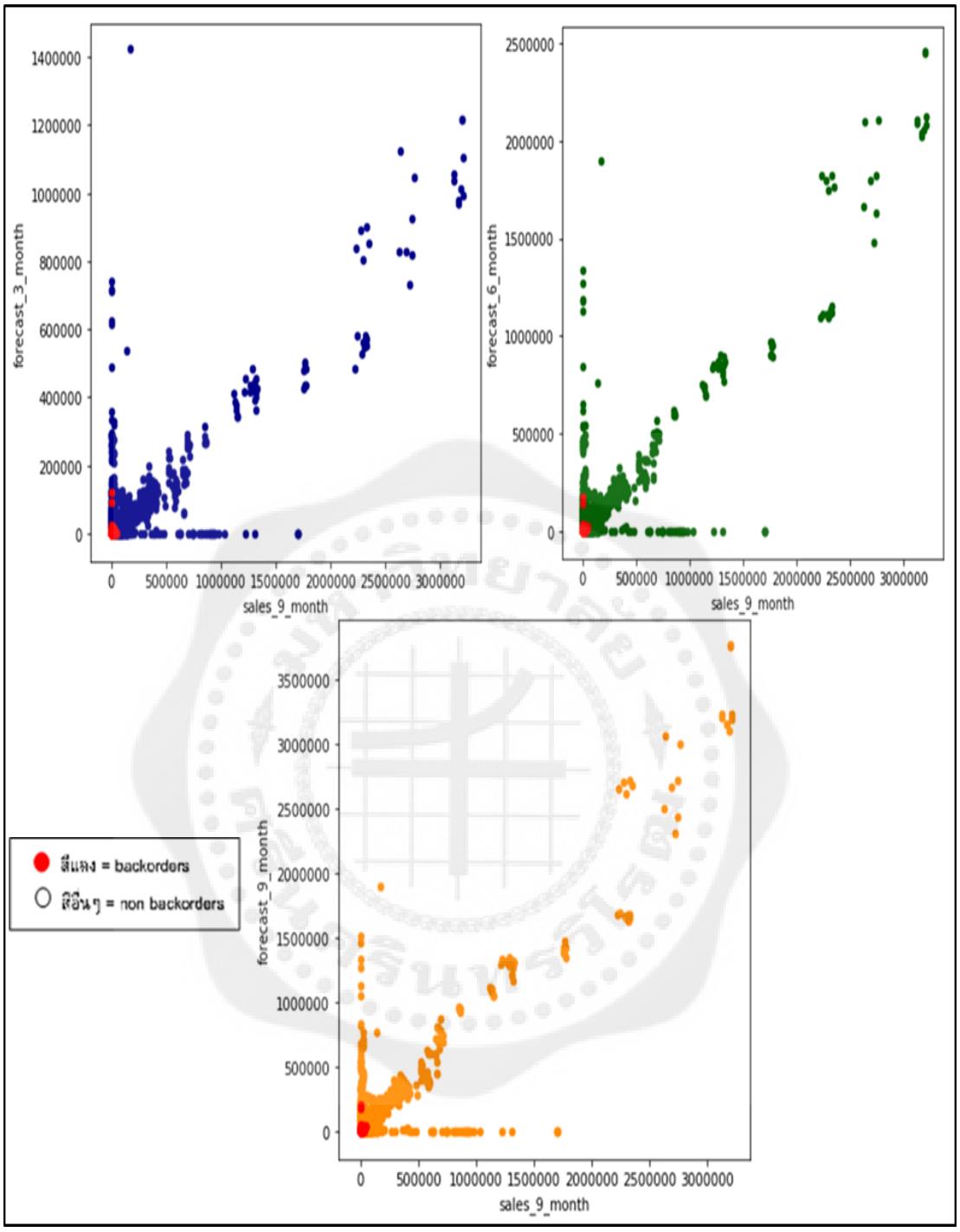
จุดสีแดงแทนยอดของการเกิดสินค้าที่เป็นสินค้า backorders และแทนค่าสินค้าที่ไม่เป็น backorders ด้วยสีอื่นๆ พบว่าเมื่อมีจำนวนยอดขายจริงในปริมาณที่น้อยจะทำให้การพยากรณ์ยอดขายในอนาคตน้อยตามไปด้วย และนอกจากนั้นยังพบว่า ปริมาณการเกิดสินค้าที่เป็นสินค้า backorders ส่วนใหญ่มักเกิดในช่วงที่มียอดขายน้อย และ ยอดขายที่พยากรณ์น้อยด้วยเช่นเดียวกัน จึงมีความสัมพันธ์ไปในทิศทางเดียวกันทำให้สามารถสรุปได้ว่า ในช่วงที่มีการขายสินค้าได้น้อย และการพยากรณ์ยอดขายน้อยมักเกิดสถานการณ์มีสินค้าที่เป็น backorders อยู่จำนวนมาก เพราะเมื่อมียอดขายน้อยจึงไม่มีการสำรองสินค้าไว้เพื่อจำหน่ายอย่างเพียงพอและเกิดความขาดแคลนสินค้าในกรณีที่มียอดขายเพิ่มขึ้นดังภาพประกอบ 9-12



ภาพประกอบ 10 ความสัมพันธ์ระหว่าง Features sales_3_month และ forecast



ภาพประกอบ 11 ความสัมพันธ์ระหว่าง Features sales_6_month และ forecast



ภาพประกอบ 12 ความสัมพันธ์ระหว่าง Features sales_9_month และ forecast

2. ส่วนที่ 2 การเตรียมข้อมูล

เนื่องจากแถวสุดท้ายในชุดข้อมูลนี้มีแถวที่ไม่มีข้อมูลอยู่จึงต้องกำจัดแถวนี้ออกไปและจัดการกับแถว (row) และคอลัมน์ (column) ที่มีข้อมูลที่เป็นค่าว่าง (NaN) (ดังภาพประกอบ 13 และภาพประกอบ 14) ให้มีค่าด้วยคำสั่ง .fillna ซึ่งได้แก่คอลัมน์ในแอตทริบิวต์ perf_6_month_avg

perf_12_month_avg และ lead_time และจาก ตาราง 1 และ ตาราง 2 จะเห็นได้ข้อมูลมีลักษณะที่ความแตกต่างกันมีทั้งข้อมูลที่เป็นตัวอักษร (String) และเป็นตัวเลข จึงต้องทำการแปลงข้อมูลที่เป็นตัวอักษร (String) ให้เป็นตัวเลขก่อน ได้แก่ แอตทริบิวต์ potential_issue deck_risk oe_constraint ppap_risk stop_auto_buy rev_stop และ went_on_backorders ซึ่งค่าของแอตทริบิวต์ที่กล่าวมานี้เป็นข้อมูลประเภท Binary มีค่าเป็น Yes และ No จึงแปลงข้อมูลที่เป็น Yes ให้มีค่าเท่ากับ 1 และ No ให้มีค่าเท่ากับ 0

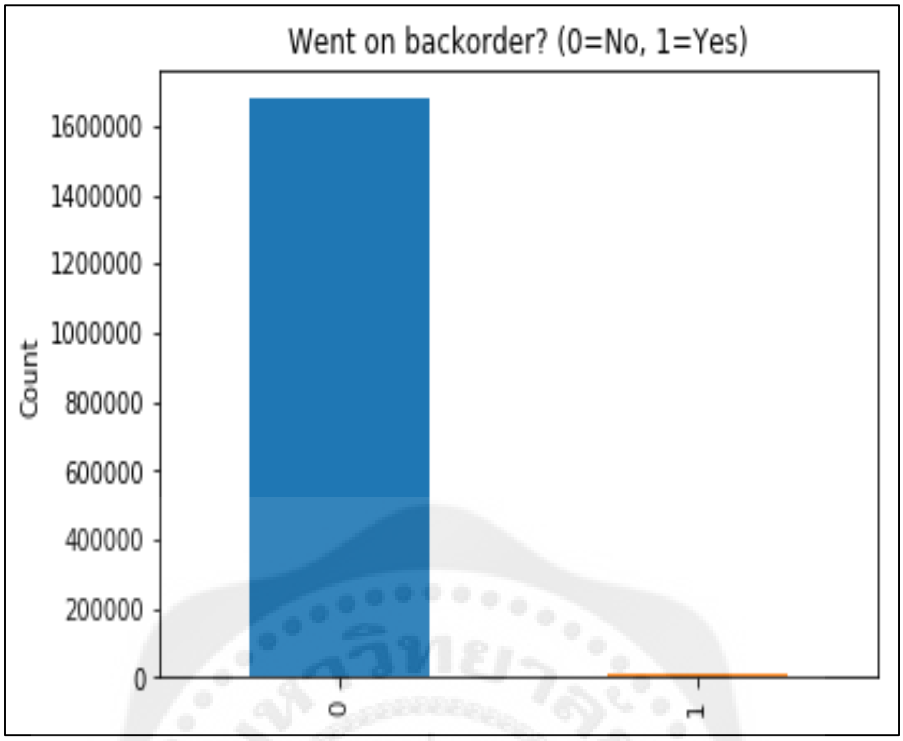
sku	national_inv	lead_time	in_transit_qty	forecast_3_month	forecast_6_month	forecast_9_month	sales_1_month	sales_3_month	sales_6_month
1373987	-1.0	NaN	0.0	5.0	7.0	9.0	1.0	3.0	3.0
1524346	-1.0	9.0	0.0	7.0	9.0	11.0	0.0	8.0	11.0
1439563	62.0	9.0	16.0	39.0	87.0	126.0	35.0	63.0	153.0
1502009	19.0	4.0	0.0	0.0	0.0	0.0	2.0	7.0	12.0
1687860 rows)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

ภาพประกอบ 13 แสดงตัวอย่างของชุดข้อมูล

pieces_past_due	perf_6_month_avg	perf_12_month_avg	local_bo_qty	deck_risk	oe_constraint	ppap_risk	stop_auto_buy	rev_stop	went_on_backorder
0.0	NaN	NaN	1.0	No	No	No	Yes	No	No
0.0	0.86	0.84	1.0	Yes	No	No	No	No	Yes
0.0	0.86	0.84	6.0	No	No	No	Yes	No	No
0.0	0.73	0.78	1.0	No	No	No	Yes	No	No
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

ภาพประกอบ 14 แสดงตัวอย่างของชุดข้อมูล (ต่อ)

เมื่อทำการแปลงค่าข้อมูลประเภท Binary จาก Yes และ No ให้เป็น 1 และ 0 ของแอตทริบิวต์ที่เราสนใจ นั่นคือ “went_on_backorders” จากรายการสินค้าทั้งหมด 1,687,860 รายการพบว่ามีสินค้าที่ไม่เป็น backorders 1,676,567 รายการ คิดเป็นร้อยละ 99.33 และสินค้าที่เป็น backorders เพียง 11,293 รายการ คิดเป็นร้อยละ 0.66 เท่านั้นซึ่งแตกต่างกันมากโดยมีข้อมูลที่เป็น backorders มีจำนวนน้อยกว่ามากจึงเกิดความไม่สมดุลของข้อมูล และอาจส่งผลต่อการสุ่มชุดข้อมูลฝึกฝน (Training dataset) ที่จะใช้ในการสร้างแบบจำลองและได้ผลลัพธ์ที่ไม่แม่นยำ ดังภาพประกอบ 15



ภาพประกอบ 15 แสดงสัดส่วนระหว่างสินค้าที่เป็น backorders (1) และไม่เป็น backorders (0)

3. ส่วนที่ 3 ระบุชุดข้อมูลฝึกฝน (Training data) และการสร้างแบบจำลอง (Modeling)

ก่อนที่เราจะทำการระบุชุดข้อมูลฝึกฝนขั้นตอนที่สำคัญก็คือการกำหนด Features และ Label ที่ใช้สำหรับชุดข้อมูลฝึกฝน (Training Data) และข้อมูลทดสอบ (Test Data) ซึ่งในการวิจัยนี้กำหนดให้ national_inv lead_time sales_1_month pieces_past_due perf_6_month_avg local_bo_qty deck_risk oe_constraint ppap_risk stop_auto_buy และ rev_stop เป็น Features และ Labels ซึ่งเป็นคำตอบของปัญหา โดยคำตอบที่ต้องการคือ went_on_backorders

จากสัดส่วนความแตกต่างของสินค้าที่เป็น backorders และไม่เป็น backorders มีความแตกต่างกันมาก ทำให้เกิดกรณีข้อมูลมีความไม่สมดุลกัน (Imbalanced Classes Problem) จึงทำให้ไม่สามารถใช้วิธีสุ่มข้อมูลแบบร้อยละ เช่นการแบ่งเป็นชุดข้อมูลฝึกฝน (Training data) เป็นร้อยละ 70 และ ข้อมูลชุดทดสอบ (Test data) เป็นร้อยละ 30 ได้ เพราะจะทำให้ผลลัพธ์ที่ได้มีแนวโน้มไปทางทิศทางใดทิศทางหนึ่งมากเกินไป (Kueipo Huang 2017) ผู้วิจัยจึงต้องใช้วิธีการสุ่มข้อมูลอีกวิธีหนึ่งคือ K-Fold Cross Validation แบบ Stratified Shuffle Split เพราะเหมาะสมสำหรับกรณีนี้มากที่สุด ซึ่งในการทดสอบในครั้งนี้ผู้วิจัยจะกำหนดค่า k เท่ากับ 10 คือจะทำการทดสอบการสุ่มข้อมูลจำนวน 10 ครั้ง โดยใช้คำสั่งโปรแกรมจาก Library ของ sklearn ดังนี้ StratifiedShuffleSplit (n_splits =10 random_state = 0) หมายถึง จำนวนค่า k หรือจำนวนครั้งของการสุ่มข้อมูล คือ 10

4. ส่วนที่ 4 การสร้างแบบจำลอง (Modeling)

เมื่อสุ่มข้อมูลด้วยวิธีการ K-Fold Cross Validation แบบ Stratified Shuffle Split แล้วจะได้ชุดข้อมูลในสัดส่วนที่เหมาะสมสำหรับการสร้างแบบจำลอง ซึ่งขั้นตอนการสร้างแบบจำลองด้วยการใช้เทคนิคการเรียนรู้ของเครื่อง โดยอัลกอริทึมที่จะทดสอบประสิทธิภาพในการสร้างแบบจำลองมีวิธีการและคำสั่งที่ใช้ในการสร้างแบบจำลองเพื่อจำแนกข้อมูลมีดังต่อไปนี้

4.1 อัลกอริทึมการเรียนรู้แบบเบย์ (Naive Bayes)

4.1.1 นำเข้าไลบารีของ sklearn สำหรับการจำแนกโดยใช้ความน่าจะเป็น ด้วยคำสั่งดังนี้ sklearn.naive_bayes import GaussianNB

4.1.2 กำหนดชื่อตัวแปรสำหรับสร้างแบบจำลองของ Naive Bayes ว่า nbmodel

4.1.3 กำหนดค่าพารามิเตอร์เพื่อให้ค่าสำหรับทำนายมีความถูกต้อง โดยผู้วิจัยได้กำหนดพารามิเตอร์ไว้ตามค่าตั้งต้น คือ GaussianNB (priors=None)

4.2 อัลกอริทึมต้นไม้ตัดสินใจ (Decision Tree)

4.2.1 นำเข้าไลบารีของ sklearn สำหรับสร้างต้นไม้การตัดสินใจ คือ sklearn.tree import DecisionTreeClassifier

4.2.2 กำหนดชื่อตัวแปรสำหรับสร้างแบบจำลองของ DecisionTree ว่า dtmodel

4.2.3 กำหนดค่าพารามิเตอร์เพื่อให้ค่าสำหรับทำนายมีความถูกต้อง โดยผู้วิจัยได้กำหนดพารามิเตอร์ตามค่าตั้งต้น

4.3 อัลกอริทึมเพื่อนบ้านใกล้เคียงกันมากที่สุด (K-nearest neighbor)

4.3.1 นำเข้าไลบารีของ sklearn คือ sklearn.neighbors import KNeighborsClassifier

4.3.2 กำหนดตัวแปรสำหรับสร้างแบบจำลองของ K-nearest neighbor ว่า knnmodel

4.3.3 กำหนดค่าพารามิเตอร์สำหรับทำนายมีความถูกต้องตามค่าตั้งต้น

4.4 อัลกอริทึมการจำแนกป่าแบบสุ่ม (Random Forest)

4.4.1 นำเข้าไลบารีของ sklearn ในกรณีของการเรียนรู้ร่วมกัน และคำสั่งสำหรับจำแนกคือ RandomForestClassifier

4.4.2 กำหนดชื่อตัวแปรสำหรับสร้างแบบจำลองของ Random Forest ว่า rfmodel

4.4.3 กำหนดค่าพารามิเตอร์สำหรับทำนายความถูกต้อง ดังนี้

- n_estimators = 10 คือ จำนวนต้นไม้ทั้งหมดที่ใช้ในตัวทำนายจำนวน 10 ต้น
- min_samples_split = 2 คือ การแบ่งจำนวนตัวอย่างที่น้อยที่สุดเท่ากับ 2
- criterion = 'gini' คือ เป็นเกณฑ์ในการวัดการกระจายของข้อมูล ซึ่งในค่าตั้งต้น กำหนดให้ใช้ Gini เป็นเกณฑ์ นอกจากนั้นใช้ค่าตั้งต้นทั้งหมด

4.5 การจำแนกแบบ Adaptive Boosting Algorithm หรือ AdaBoost

4.5.1 นำเข้าไลบารีของ sklearn ในกรณีของการเรียนรู้ร่วมกัน คำสั่งสำหรับการจำแนกคือ AdaBoostClassifier

4.5.2 กำหนดชื่อตัวแปรสำหรับสร้างแบบจำลองของ AdaBoost ไว้ว่า adbmodel

4.5.3 กำหนดค่าพารามิเตอร์สำหรับทำนายความถูกต้อง ด้วยพารามิเตอร์ดังนี้

- n_estimators = 50 คือ จำนวนที่ใช้ในการ Boost จำนวน 50 ตัว
- Algorithm='SAMME.R' คือ Algorithm ที่ใช้ในการ boost
- learning_rate=1.0 คือ อัตราในการเรียนรู้

4.6 การจำแนกแบบ Extreme Gradient Boosting หรือ XGBoost

4.6.1 นำเข้าไลบารีของ sklearn ที่สำคัญ ได้แก่ from sklearn.preprocessing import LabelEncoder และ import xgboost as xgb

4.6.2 กำหนดชื่อตัวแปรสำหรับสร้างแบบจำลองของ XGBoost ไว้ว่า xgbmodel

4.6.3 กำหนดค่าพารามิเตอร์สำหรับทำนายมีความถูกต้อง ด้วยพารามิเตอร์ดังนี้

- max_depth คือ พารามิเตอร์ที่กำหนดความลึกในการสร้างต้นไม้ให้เท่ากับ 3
- n_estimators คือจำนวนต้นไม้ของปัญหาให้เท่ากับ 100 ต้น
- learning_rate คือ อัตราในการเรียนรู้ โดยค่าตั้งต้นปกติจะมีค่า 0.1 มักกำหนดค่า Learning Rate ไม่เกิน 1

5. ส่วนที่ 5 ทดสอบประสิทธิภาพ และประเมินผลแบบจำลองที่สร้างขึ้น (Evaluation)

ในการทดสอบประสิทธิภาพของแบบจำลองในงานวิจัยนี้ประกอบไปด้วยเกณฑ์ที่ใช้ในการประเมินผลแบบจำลองดังต่อไปนี้ [16]

5.1 Confusion Matrix

เป็นเกณฑ์ในการทดสอบประสิทธิภาพซึ่งจะทำการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายได้ในรูปแบบของตารางเป็นเมตริกซ์ เรียกว่า “Confusion Matrix” [3] ดังตาราง 3

ตาราง 3 Confusion Matrix

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	TP (True Positive)	FN (False Negative)
ไม่เป็น backorders	FP (False Positive)	TN (True Negative)

โดยที่

True Positive (TP) คือ จำนวนข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders

True Negative (TN) คือจำนวนข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น backorders

False Positive (FP) คือจำนวนข้อมูลที่ไม่เป็น backorders แต่ทำนายว่าเป็น backorders

False Negative (FN) คือจำนวนข้อมูลที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders

5.2 ค่าความถูกต้อง (Accuracy)

เป็นการวัดความถูกต้องคำตอบของข้อมูลทั้งหมดจากแบบจำลองที่สร้างขึ้นว่ามีความใกล้เคียงกับค่าจริงมากน้อยเพียงใดโดยรวมสามารถคำนวณเป็นสมการได้ดังนี้

$$\text{Accuracy} = [(TP+TN)/ (TP+TN+FP+FN)] \times 100 \quad (2)$$

5.3 ค่า Precision

เป็นค่าที่ใช้วัดความแม่นยำของแบบจำลองโดยจะพิจารณาจากคลาสที่มีคำตอบที่ทำนายเป็น backorders ทั้งหมด คำนวณเป็นสมการได้ดังนี้

$$\text{Precision} = TP/ (TP+FP) \quad (3)$$

5.4 ค่า Recall

เป็นค่าที่พิจารณาจากคลาสที่ทำนายถูกจากค่าจริงทั้งหมดซึ่งเป็นสูตรของคลาสที่มีคำตอบเป็นbackorders สามารถคำนวณเป็นสมการได้ดังนี้

$$\text{Recall} = TP/ (TP+FN) \quad (4)$$

5.5 ค่า F-Measure หรือ F1 score

เป็นค่ากลางระหว่าง ค่า Precision และค่า Recall ดังสมการที่แสดงดังต่อไปนี้

$$\text{F1 score} = 2 \times (\text{Precision} \times \text{Recall} / \text{Precision} + \text{Recall}) \quad (5)$$

5.6 ค่าจากกราฟ ROC (Receiver Operator Characteristic)

เป็นกราฟที่แสดงความสัมพันธ์ระหว่างสัดส่วนของข้อมูลที่ทำนายถูก

(True Positive Rate) กำหนดให้เป็นแกน Y และค่าที่ทำนายไม่ถูก (False Positive Rate) เป็นแกน X ค่าที่บ่งชี้ประสิทธิภาพในการทำนายที่ดีจะต้องมีค่าพื้นที่ใต้กราฟ หรือ Area Under the Curve (AUC) อยู่ที่ร้อยละ 80 ถึงร้อยละ 95

5.7 ค่าความแม่นยำเฉลี่ย (Average Precision) และค่ากราฟ Precision-Recall

5.7.1 Average precision (AP)

คือ เกณฑ์ที่ใช้ในการประเมินประสิทธิภาพการจำแนกของแบบจำลองจากค่าความแม่นยำเฉลี่ยซึ่งค่า Average precision ที่ดีจะต้องเป็นค่าที่เข้าใกล้ 1 จึงจะแสดงถึงความแม่นยำของประสิทธิภาพมาก

5.7.2 ค่ากราฟ Precision-Recall (Precision-Recall Curve)

เป็นกราฟที่บ่งบอกถึงความสัมพันธ์ระหว่าง Precision และ Recall แสดงในรูปแบบของกราฟที่เป็นเส้นโค้ง ซึ่งจะแสดงถึงการเปลี่ยนแปลงของค่า Precision เมื่อค่า Recall เพิ่มขึ้นหรือเปลี่ยนไป โดยปกติจะมีคะแนนความเชื่อมั่นที่แตกต่างกันออกไป ดังนั้นจึงไม่อาจมีเพียงค่าใดค่าหนึ่งได้ต้องใช้การวัดประสิทธิภาพจากทั้งสองค่า โดยพิจารณาจากค่าที่อยู่ใต้เส้นโค้ง หากค่าเข้าใกล้ 1 หรือมีพื้นที่เส้นโค้งมากแสดงว่าค่านั้นบ่งบอกถึงความมีประสิทธิภาพดีของแบบจำลอง

บทที่ 4
ผลการศึกษา

ผลของการศึกษาในงานวิจัยครั้งนี้แสดงผลการสร้างแบบจำลองอัลกอริทึมและตัวชี้วัดที่ใช้ในการวัดประสิทธิภาพของแบบจำลองซึ่งก่อนทำการสร้างแบบจำลองทุกครั้งต้องมีการสุ่มข้อมูลให้เกิดความสมดุลด้วยฟังก์ชันการทำ K-Fold Cross Validation แบบ Stratified Shuffle Split ด้วยการกำหนดตัวแปรที่จะใช้แทนข้อมูลฝึกฝน และข้อมูลทดสอบ รวมถึง Parameter ที่ใช้ในการสุ่มข้อมูล ซึ่งค่า K ที่เราได้จากคำสั่งนี้คือมีค่าเท่ากับ 10 นั้นหมายถึงเราจะทำการสุ่มข้อมูลทั้งหมด 10 ครั้ง

```
In [74]: from sklearn.model_selection import StratifiedShuffleSplit
         X = train_df.drop('went_on_backorder', axis=1).values
         y = train_df['went_on_backorder'].values
         sss = StratifiedShuffleSplit(n_splits=10, test_size=0.3, random_state=0)
         sss.get_n_splits(X, y)

Out[74]: 10

In [75]: for train_index, test_index in sss.split(X, y):
             X_features_train, X_features_test = X[train_index], X[test_index]
             y_labels_train,  y_labels_test = y[train_index], y[test_index]
```

ภาพประกอบ 16 คำสั่งและการกำหนดพารามิเตอร์สำหรับ Stratified Shuffle Split

ผลที่ได้จากการวัดประสิทธิภาพของแบบจำลอง สามารถแยกตามอัลกอริทึมจะพิจารณาจากจำนวน ข้อมูลทั้งหมดที่ได้จากการแบ่งด้วย K-Fold Cross Validation แบบ Stratified Shuffle Split รวมทั้งสิ้น 506,358 รายการ มีสินค้าที่ไม่เป็น backorders ทั้งหมด 502,970 และเป็นสินค้าที่เป็น backorders 3,388 รายการ ได้ดังนี้

1. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม Naive Bayes

1.1 ค่า Accuracy ของอัลกอริทึม Naive Bayes

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม Naive Bayes เป็นค่าตั้งต้น คือ (None) ดังภาพประกอบ 17 เพื่อใช้ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้องเพียง 0.0601 หรือคิดเป็นร้อยละ 6.01 เท่านั้นซึ่งค่าถูกต้องที่ได้นั้นน้อยมากอัลกอริทึมนี้จึงอาจไม่เหมาะสมต่อการนำไปใช้งานต่อไปหากวัดประสิทธิภาพจากค่า Accuracy เพราะค่า Accuracy ที่ดีจะต้องมีค่าเข้าใกล้ 1 หรือ คิดเป็นร้อยละ 100 แต่อัลกอริทึมนี้ใช้เวลาในการประมวลผลไม่นานเพียง 2.26 วินาทีเท่านั้น

```
In [78]: from sklearn.naive_bayes import GaussianNB
         nbmodel = GaussianNB()
         nbmodel.fit(X_features_train,y_labels_train)

Out[78]: GaussianNB(priors=None)
```

ภาพประกอบ 17 การกำหนดค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ Naive Bayes

1.2 ค่า Precision ของอัลกอริทึม Naive Bayes

ผลจากการสรุปรายงานการจำแนก (Classification report) ระหว่างค่าจริงที่ผ่านการสุ่มข้อมูลแบบ Stratified Shuffle Split และค่าที่ทำนายได้ของอัลกอริทึม Naive Bayes พบว่าผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision ถึง 0.992 คิดเป็นร้อยละ 99.20 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่เป็น backorders มีค่า 0.007 คิดเป็นร้อยละ 0.70 ในขณะที่ ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.998 คิดเป็นร้อยละ 99.80 เนื่องจากค่า Precision ที่ใช้วัดความแม่นยำของแบบจำลองโดยจะพิจารณาจากคลาสที่มีคำตอบที่ทำนายเป็น backorders ทั้งหมด ค่า Precision ของคลาสที่เป็น backorders จึงสูงกว่า แสดงว่ามีความแม่นยำ ดังภาพประกอบ 18

1.3 ค่า Recall ของอัลกอริทึม Naive Bayes

เมื่อพิจารณาจากผลลัพธ์แสดงในภาพประกอบ 18 พบว่า ค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม Naive Bayes มีค่า 0.060 คิดเป็นร้อยละ 6.00% ซึ่งค่า Recall นี้มีค่าน้อยมาก สามารถอธิบายได้ว่า Naive Bayes มีความแม่นยำน้อย แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.054 คิดเป็นร้อยละ 5.40 ในขณะที่ คลาสของสินค้าที่เป็นสินค้า backorders มีค่า Recall 0.986 คิดเป็นร้อยละ 98.60 แสดงให้เห็นว่า คลาสที่ทำนายถูกจากค่าจริงทั้งหมดของคำตอบที่เป็นสินค้า backorders มีความแม่นยำมาก

1.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม Naive Bayes

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 18 พบว่า ค่า F1 score ที่เฉลี่ยรวมของอัลกอริทึม Naive Bayes ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.102 คิดเป็นร้อยละ 10.20 ซึ่งค่า F1 score เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders และ ไม่เป็น backorders พบว่า คลาสที่เป็นสินค้า backorders มีค่า F1 score เท่ากับ 0.014 คิดเป็นร้อยละ 1.40 และค่า F1 score ของสินค้าที่ ไม่เป็น backorders มีค่าเท่ากับ 0.102 คิดเป็นร้อยละ 10.20 ดังนั้น ค่า F1 score ที่แสดงถึงความแม่นยำที่ดี จะต้องมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 สามารถนี้อธิบายผลลัพธ์ได้ว่า ค่า F1 score ของอัลกอริทึม Naive Bayes โดยเฉลี่ยรวมมีความแม่นยำน้อย

	precision	recall	f1-score	support
0	0.998	0.054	0.102	502970
1	0.007	0.986	0.014	3388
avg / total	0.992	0.060	0.102	506358

ภาพประกอบ 18 ผลลัพธ์จาก classification report ของ Naive Bayes

1.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม Naive Bayes

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

1.5.1 True Positive (TP) ของอัลกอริทึม Naive Bayes

คือ จำนวนข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders มีค่าเท่ากับ 3,340 แสดงว่าค่าที่ทำนายถูกเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีค่าน้อยแสดงว่ามีความแม่นยำน้อย

1.5.2 True Negative (TN) ของอัลกอริทึม Naive Bayes

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 27,104 แสดงว่าความแม่นยำของอัลกอริทึมเมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders ค่าที่ทำนายถูกมีค่าน้อย แสดงว่ามีความถูกต้องในการทำนายน้อยและมีความแม่นยำน้อย

1.5.3 False Positive (FP) ของอัลกอริทึม Naive Bayes

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders มีค่าเท่ากับ 475,866 แสดงว่ามีความผิดพลาดในการทำนายมาก ดังนั้นจึงมีความแม่นยำน้อย

1.5.4 False Negative (FN) ของอัลกอริทึม Naive Bayes

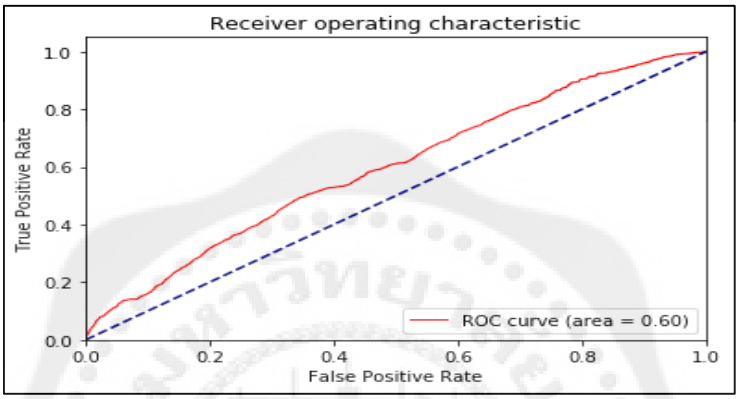
คือ จำนวนข้อมูลที่ค่าจริงเป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 48 ซึ่งแสดงว่า ค่าที่แสดงความแม่นยำของอัลกอริทึมเมื่อพิจารณาจากคลาสที่เป็น backorders มีความผิดพลาดในทำนายไม่ถูกต้องเพียงเล็กน้อย

ตาราง 4 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ Naive Bayes

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	3,340 (TP)	48 (FN)
ไม่เป็น backorders	475,866 (FP)	27,104 (TN)

1.6 ค่าจากกราฟ ROC Curve ของอัลกอริทึม Naive Bayes

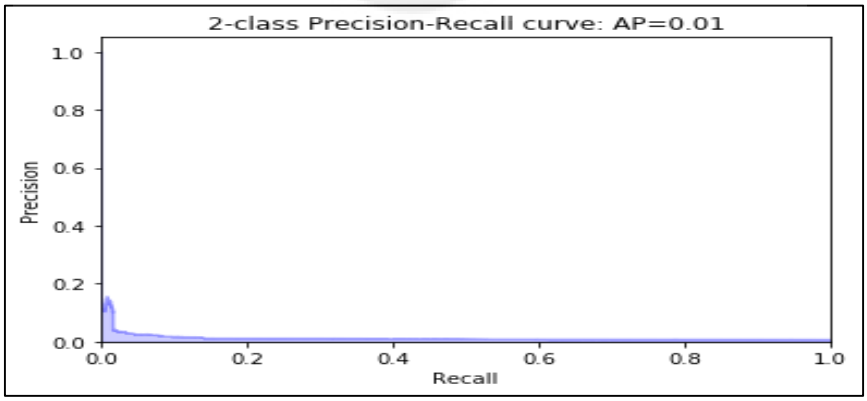
จากกราฟ ROC Curve ของอัลกอริทึม Naive Bayes ได้ผลลัพธ์ของค่าพื้นที่ใต้กราฟ (AUC) เท่ากับ 0.60 โดยค่านี้จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายซึ่งค่าต้องต้องอยู่ระหว่างช่วงร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่าพื้นที่ใต้กราฟของอัลกอริทึม Naive Bayes มีค่าน้อยกว่าช่วงที่แสดงถึงประสิทธิภาพประกอบดี ดังนั้นสรุปได้ว่า Naive Bayes มีความแม่นยำน้อยและไม่เหมาะสมต่อการนำแบบจำลองไปใช้ในการตัดสินใจ ดังภาพประกอบ 19



ภาพประกอบ 19 กราฟ ROC Curve ของ Naive Bayes

1.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม Naive Bayes

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม Naive Bayes พบว่ามีค่าเท่ากับ 0.01 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อยมากซึ่งสอดคล้องกับ กราฟ Precision-Recall ที่มีพื้นที่ใต้กราฟค่อนข้างที่จะน้อย และสามารถแสดงเป็นกราฟ Precision-Recall ได้ดังนี้



ภาพประกอบ 20 กราฟ Precision-Recall ของ Naive Bayes

2. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม Decision Tree

2.1 ค่า Accuracy ของอัลกอริทึม Decision Tree

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม Decision Tree กำหนดพารามิเตอร์ตามค่าตั้งต้นของไลบารี sklearn ดังภาพประกอบ 21 ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้อง 0.9915 หรือคิดเป็นร้อยละ 99.15 ซึ่งค่าถูกต้องอยู่ในช่วงค่าความถูกต้องที่เข้าใกล้ 1 คิดเป็นร้อยละ 100 แสดงว่าอัลกอริทึมนี้จึงมีค่าความถูกต้องสูงใช้เวลาในการประมวลผล 39.43 วินาที

```
In [52]: from sklearn.tree import DecisionTreeClassifier
         dtmodel = DecisionTreeClassifier()
         dtmodel.fit(X_features_train,y_labels_train)

Out[52]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_split=1e-07, min_samples_leaf=1,
                                min_samples_split=2, min_weight_fraction_leaf=0.0,
                                presort=False, random_state=None, splitter='best')
```

ภาพประกอบ 21 ค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ Decision Tree

2.2 ค่า Precision ของอัลกอริทึม Decision Tree

ผลจากการสรุปรายงานการจำแนก (Classification report) ระหว่างค่าจริงที่ผ่านการสุ่มข้อมูลแบบ Stratified Shuffle Split และค่าที่ทำนายได้ของอัลกอริทึม Decision Tree พบว่า ผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision ถึง 0.992 คิดเป็นร้อยละ 99.20 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่เป็น backorders มีค่า 0.366 คิดเป็นร้อยละ 36.60 ในขณะที่ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.996 คิดเป็นร้อยละ 99.60 เนื่องจากค่า Precision ที่ใช้วัดความแม่นยำของแบบจำลองโดยจะพิจารณาจากคลาสที่มีคำตอบที่ทำนายเป็น backorders ทั้งหมดแต่เนื่องจากคลาสของสินค้าที่เป็น backorders มีจำนวนน้อยกว่าคลาสของสินค้าที่ไม่เป็น backorders ดังนั้นจากข้อมูลจำนวนของสินค้าที่เป็น backorders จึงมีค่าความแม่นยำน้อย

2.3 ค่า Recall ของอัลกอริทึม Decision Tree

เมื่อพิจารณาจากค่าผลลัพธ์แสดงในภาพประกอบ 22 พบว่าค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม Decision Tree มีค่า 0.991 คิดเป็นร้อยละ 99.10 ซึ่งค่า Recall นี้มีค่าสูง สามารถอธิบายได้ว่า Decision Tree มีความแม่นยำมาก เนื่องจากค่าที่ได้มีความเข้าใกล้ 1 หรือ ร้อยละ 100 แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0)

พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.996 คิดเป็นร้อยละ 99.60 ในขณะที่ คลาสของสินค้าที่เป็นสินค้า backorders มีค่า Recall เพียง 0.375 คิดเป็นร้อยละ 37.50 แสดงให้เห็นว่า คลาสที่ทำนายถูกจากค่าจริงทั้งหมดของคำตอบที่เป็นสินค้า backorders มีความแม่นยำน้อย

2.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม Decision Tree

เมื่อพิจารณาจาผลลัพธ์ที่แสดงในภาพประกอบ 22 พบว่า ค่า F1 score ที่เฉลี่ยรวมของอัลกอริทึม Decision Tree ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.992คิดเป็นร้อยละ 99.20 ซึ่ง ดังนั้น ค่า F1 score ที่แสดงถึงความแม่นยำสูง เพราะมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 แต่เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า F1 score ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.996 คิดเป็น 99.60 ส่วนคลาสของสินค้าที่เป็นสินค้า backorders มีค่า 0.375 คิดเป็นร้อยละ 37.50 แสดงให้เห็นว่า ค่ากลางของค่า Precision และ ค่า Recall มีความแม่นยำโดยเฉลี่ยน้อย

	precision	recall	f1-score	support
0	0.996	0.996	0.996	502970
1	0.366	0.375	0.371	3388
avg / total	0.992	0.991	0.992	506358

ภาพประกอบ 22 ผลลัพธ์จาก classification report ของ Decision Tree

2.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม Decision Tree

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

2.5.1 True Positive (TP) ของอัลกอริทึม Decision Tree

คือ จำนวนของข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders มีค่าเท่ากับ 1,272 แสดงว่าค่าที่ทำนายเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีค่ามากแสดงว่าทำนายถูกน้อยเมื่อเทียบจากจำนวนข้อมูลทั้งหมดแสดงว่ามีความแม่นยำในการจำแนกน้อย

2.5.2 True Negative (TN) ของอัลกอริทึม Decision Tree

คือ จำนวนของข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 500,769 แสดงว่า ความแม่นยำของอัลกอริทึมเมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders มีค่าที่ทำนายถูกมากเพราะจากจำนวนสินค้าทั้งหมดมีสินค้าที่ไม่เป็น backorders จำนวนมากซึ่งสมเหตุสมผลกันกับค่าจริง ดังนั้นจึงมีความแม่นยำของการจำแนกมาก

2.5.3 False Positive (FP) ของอัลกอริทึม Decision Tree

คือ จำนวนของข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders มีค่าเท่ากับ 2,201 แสดงว่า ความแม่นยำของอัลกอริทึม เมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders มีค่าที่ทำนายที่ไม่ถูกต้องเป็นส่วนน้อย แสดงว่ายังมีความแม่นยำในการจำแนกมาก

2.5.4 False Negative (FN) ของอัลกอริทึม Decision Tree

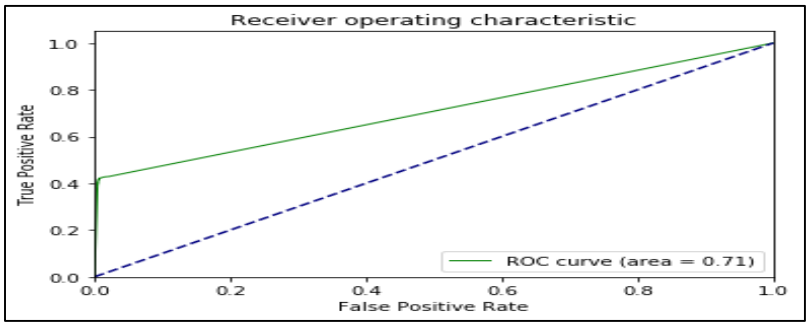
คือ จำนวนข้อมูลค่าจริงที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 2,116 ซึ่งแสดงว่า ค่าที่แสดงความแม่นยำของอัลกอริทึมเมื่อพิจารณาจากคลาสที่เป็น backorders มีค่าที่ทำนายไม่ถูกต้องมีจำนวนมากแสดงว่ามีความแม่นยำในการจำแนกน้อย

ตาราง 5 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ Decision Tree

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	1,272 (TP)	2,116 (FN)
ไม่เป็น backorders	2,201 (FP)	500,769 (TN)

2.6 ค่าจากกราฟ ROC Curve ของอัลกอริทึม Decision Tree

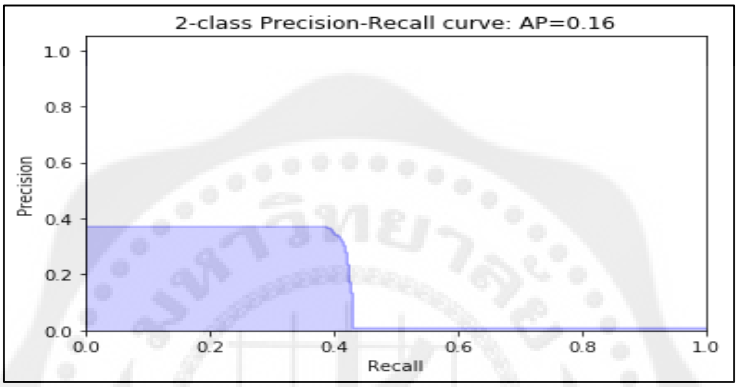
จากการประมวลผลคำสั่งในการเปรียบเทียบค่าของพื้นที่ใต้กราฟ ROC Curve (AUC) ของอัลกอริทึม Decision Tree เท่ากับ 0.71 โดยค่าที่จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายต้องมีค่าระหว่างช่วงร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่าพื้นที่ใต้กราฟของอัลกอริทึม Decision Tree มีค่าน้อยกว่าช่วงที่แสดงถึงประสิทธิภาพประกอบดี ดังนั้นสรุปได้ว่า Decision Tree มีความแม่นยำอยู่ในเกณฑ์พอใช้และไม่อาจเหมาะสมต่อการนำแบบจำลองไปใช้ในการตัดสินใจ ดังภาพประกอบ 23



ภาพประกอบ 23 กราฟ ROC Curve ของ Decision Tree

2.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม Decision Tree

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม Decision Tree พบว่ามีค่าเท่ากับ 0.16 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อยและ กราฟ Precision-Recall ก็เอียงไปทางด้านซ้ายน้อยด้วย ซึ่งแตกต่างจากค่าของ ROC Curve ที่มีค่าของพื้นที่ใต้กราฟค่อนข้างที่จะสูง ทำให้สรุปได้ว่าผลการทดสอบประสิทธิภาพของแบบจำลองไม่สามารถใช้กราฟ ROC Curve เพียงกราฟเดียวในการใช้เป็นตัวชี้วัด สามารถแสดงเป็นกราฟได้ดังนี้



ภาพประกอบ 24 กราฟ Precision-Recall ของ Decision Tree

3. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม K-nearest neighbor

3.1 ค่า Accuracy ของอัลกอริทึม K-nearest neighbor

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม K-nearest neighbor กำหนดพารามิเตอร์ตามค่าตั้งต้นของไลบารี sklearn ดังภาพประกอบ 25 ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้อง 0.9915 หรือคิดเป็นร้อยละ 99.15 ซึ่งค่าถูกต้องอยู่ในช่วงค่าความถูกต้องที่เข้าใกล้ 1 คิดเป็นร้อยละ 100 แสดงว่าอัลกอริทึมนี้จึงมีค่าความถูกต้องสูงใช้เวลาในการประมวลผล 3,048.89 วินาที

```
In [123]: from sklearn.neighbors import KNeighborsClassifier
          knnmodel = KNeighborsClassifier()
          knnmodel.fit(X_features_train, y_labels_train)

Out[123]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                    metric_params=None, n_jobs=1, n_neighbors=5, p=2,
                    weights='uniform')
```

ภาพประกอบ 25 การกำหนดค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ K-nearest neighbor

3.2 ค่า Precision ของอัลกอริทึม K-nearest neighbor

เมื่อทำการประมวลผลคำสั่งในการสรุปรายงานการจำแนก (Classification report) ของอัลกอริทึม K-nearest neighbor พบว่า ผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision เท่ากับ 0.991 คิดเป็นร้อยละ 99.10 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.994 คิดเป็นร้อยละ 99.40 ในขณะที่ ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.514 คิดเป็นร้อยละ 51.40 ในการวัดประสิทธิภาพของจะพิจารณาจากคลาสของสินค้าที่เป็น backorders มีจำนวนน้อยกว่า คลาสของสินค้าที่ไม่เป็น backorders ซึ่งเราจะใช้ค่าของคลาสที่เป็น backorders ในการวัดผลจึงสรุปได้ว่ามีความแม่นยำปานกลาง

3.3 ค่า Recall ของอัลกอริทึม K-nearest neighbor

เมื่อพิจารณาจากผลลัพธ์แสดงในภาพประกอบ 26 พบว่า ค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม K-nearest neighbor มีค่า 0.993 คิดเป็นร้อยละ 99.30 ซึ่งค่า Recall นี้มีค่าสูง สามารถอธิบายได้ว่า อัลกอริทึม K-nearest neighbor มีความแม่นยำมาก เพราะมีค่าของผลลัพธ์เนื่องจากความเข้าใกล้ 1 หรือ ร้อยละ 100 ซึ่งเป็นค่าที่แสดงถึงประสิทธิภาพประกอบดีในการทำนาย แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.999 คิดเป็นร้อยละ 99.90 ในขณะที่คลาสของสินค้าที่เป็นสินค้า backorders มีค่า Recall เท่ากับ 0.153 คิดเป็นร้อยละ 15.30 แสดงว่ามีความแม่นยำน้อย

3.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม K-nearest neighbor

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 26 พบว่า ค่า F1 score ที่เฉลี่ยรวมของอัลกอริทึม K-nearest neighbor ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.992 คิดเป็นร้อยละ 99.20 ซึ่ง ดังนั้น ค่า F1 score ที่แสดงถึงความแม่นยำสูง เพราะมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 แต่เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่าค่า F1 score ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.997 คิดเป็น 99.70 ส่วนคลาสของสินค้าที่เป็นสินค้า backorders มีค่า 0.236 คิดเป็นร้อยละ 23.60 แสดงให้เห็นว่า ค่ากลางของค่า Precision และ ค่า Recall ของคลาสที่เป็นสินค้าที่เป็น backorders มีความแม่นยำเฉลี่ยน้อย

	precision	recall	f1-score	support
0	0.994	0.999	0.997	502970
1	0.514	0.153	0.236	3388
avg / total	0.991	0.993	0.992	506358

ภาพประกอบ 26 ผลลัพธ์จาก classification report ของ K-nearest neighbor

3.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม K-nearest neighbor

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

3.5.1 True Positive (TP) ของอัลกอริทึม K-nearest neighbor

คือ จำนวนของข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders มีค่าเท่ากับ 520 แสดงว่าค่าที่ทำนายเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีค่าน้อยจากจำนวนของสินค้าที่ค่าจริงเป็น backorders ทั้งหมดแสดงว่าทำนายถูกน้อย และมีความแม่นยำน้อย

3.5.2 True Negative (TN) ของอัลกอริทึม K-nearest neighbor

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 502,479 ซึ่งสอดคล้องกับความเป็นจริงข้อมูลนี้มีสินค้าที่ไม่เป็น backorders มากกว่า จึงสมเหตุสมผลและสรุปได้ว่า ค่าความแม่นยำของอัลกอริทึมนี้เมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders ส่วนใหญ่ทำนายถูกต้อง ดังนั้นจึงมีความแม่นยำมาก

3.5.3 False Positive (FP) ของอัลกอริทึม K-nearest neighbor

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders มีค่าเท่ากับ 491 แสดงว่า เมื่อพิจารณาจากคลาสที่ไม่เป็น backorders พบว่ามีความไม่ถูกต้องในการทำนายมีน้อยมาก จากข้อมูลส่วนใหญ่แสดงว่าความแม่นยำในการจำแนกของอัลกอริทึมเพราะมีการทำนายไม่ถูกต้องเพียงเล็กน้อย

3.5.4 False Negative (FN) ของอัลกอริทึม K-nearest neighbor

คือ จำนวนของข้อมูลที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 2,868 หมายความว่า มีค่าจากการทำนายไม่ถูกต้องอยู่ในเกณฑ์พอใช้เมื่อเทียบกับจำนวนของสินค้าที่เป็น backorders ทั้งหมดแสดงว่ายังอยู่ในเกณฑ์ที่มีความแม่นยำพอใช้

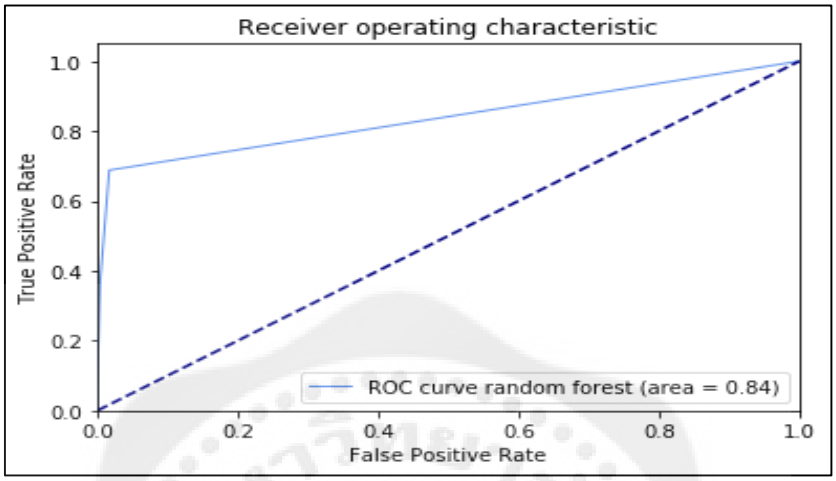
ตาราง 6 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ K-nearest neighbor

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	520 (TP)	2,868 (FN)
ไม่เป็น backorders	491 (FP)	502,479 (TN)

3.6 ค่าจากกราฟ ROC Curve ของอัลกอริทึม K-nearest neighbor

จากการประมวลผลคำสั่งในการเปรียบเทียบค่าของพื้นที่ใต้กราฟ ROC Curve (AUC) ของอัลกอริทึม K-nearest neighbor เท่ากับ 0.84 ดังภาพประกอบ 27 โดยค่าที่จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายต้องมีค่าระหว่างช่วงร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่า

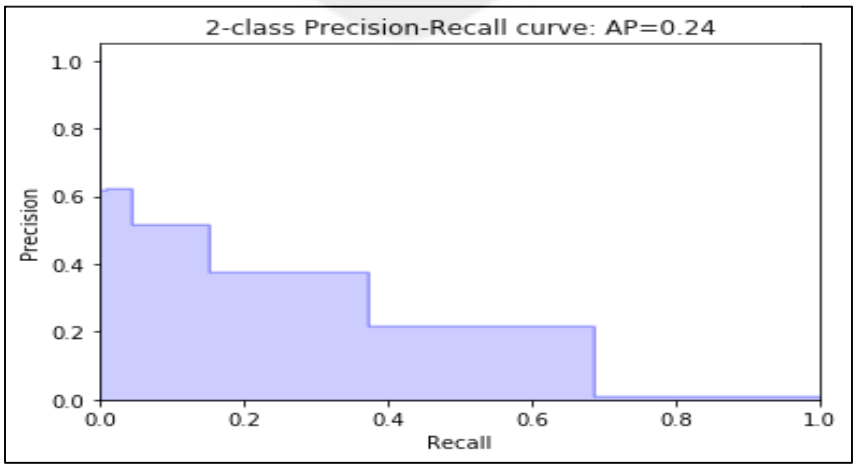
พื้นที่ใต้กราฟของอัลกอริทึม K-nearest neighbor มีค่าอยู่ในเกณฑ์ที่แสดงถึงประสิทธิภาพการทำนายดี ดังภาพประกอบ 27 ดังนั้นสรุปได้ว่า K-nearest neighbor มีความแม่นยำสูงกว่าอัลกอริทึม Decision Tree และ Naive Bayes



ภาพประกอบ 27 กราฟ ROC Curve ของ K-nearest neighbor

3.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม K-nearest neighbor

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม K-nearest neighbor พบว่ามีค่าเท่ากับ 0.24 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อย แต่เมื่อเปรียบเทียบกับ Naive bays และ Decision tree พบว่าให้ค่า Average Precision ของ K-nearest neighbor มีค่าสูงกว่า แสดงเป็นกราฟ Precision-Recall ได้ดังภาพประกอบ 28



ภาพประกอบ 28 กราฟ Precision- Recall ของ K-nearest neighbor

4. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม Random Forest

4.1 ค่า Accuracy ของอัลกอริทึม Random Forest

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม Random Forest กำหนดพารามิเตอร์ตามค่าตั้งต้นของไลบารี sklearn ดังภาพประกอบ 29 ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้อง 0.9943 หรือคิดเป็นร้อยละ 99.43 ซึ่งค่าถูกต้องอยู่ในช่วงค่าความถูกต้องที่เข้าใกล้ 1 คิดเป็นร้อยละ 100 แสดงว่าอัลกอริทึมนี้จึงมีค่าความถูกต้องสูง และใช้เวลาในการประมวลผล 96.16 วินาที ซึ่งในเวลาในการประมวลผลเร็วกว่า อัลกอริทึม K-nearest neighbor มาก

```
In [38]: from sklearn.ensemble import RandomForestClassifier
         rfmodel = RandomForestClassifier() #Random state to get a consistent score
         rfmodel.fit(X_features_train, y_labels_train)

Out[38]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                     max_depth=None, max_features='auto', max_leaf_nodes=None,
                     min_impurity_split=1e-07, min_samples_leaf=1,
                     min_samples_split=2, min_weight_fraction_leaf=0.0,
                     n_estimators=10, n_jobs=1, oob_score=False, random_state=None,
                     verbose=0, warm_start=False)
```

ภาพประกอบ 29 ค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ Random Forest

4.2 ค่า Precision ของอัลกอริทึม Random Forest

ผลจากการสรุปรายงานการจำแนก (Classification report) พบว่า ผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision เท่ากับ 0.9934 คิดเป็นร้อยละ 99.34 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่เป็น backorders มีค่า 0.7977 คิดเป็นร้อยละ 79.77 ในขณะที่ ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.9947 คิดเป็นร้อยละ 99.47 เนื่องจากค่า Precision ที่ใช้วัดความแม่นยำของแบบจำลองโดยจะพิจารณาคลาสจากสินค้าที่เป็น backorders เป็นตัวชี้วัดประสิทธิภาพ จึงสรุปได้ว่าค่าของความแม่นยำของอัลกอริทึม Random Forest มีค่าความแม่นยำที่อยู่ในเกณฑ์ที่สูง มากกว่าอัลกอริทึม Naïve Bays Decision tree และ K-nearest neighbor

4.3 ค่า Recall ของอัลกอริทึม Random Forest

เมื่อพิจารณาจากผลลัพธ์แสดงในภาพประกอบ 30 พบว่า ค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม Random Forest มีค่า 0.9943 คิดเป็นร้อยละ 99.43 ซึ่งค่า Recall นี้มีค่าสูง สามารถอธิบายได้ว่า อัลกอริทึม Random Forest มีความแม่นยำมาก เพราะมีค่าของผลลัพธ์เนื่องจากความ

เข้าใกล้ 1 หรือ ร้อยละ 100 และเป็นค่าที่แสดงถึงประสิทธิภาพประกอบดีในการทำนาย แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.9996 คิดเป็นร้อยละ 99.96 ในขณะที่คลาสของสินค้าที่เป็นสินค้า backorders มีค่า Recall 0.2060 คิดเป็นร้อยละ 20.60 จึงสรุปผลได้ว่าค่าความแม่นยำของอัลกอริทึม Random Forest โดยพิจารณาจากค่า Recall ที่ใช้ค่าจากคลาสของสินค้าที่เป็น backorders เป็นตัวชี้วัดพบว่า ค่า Recall มีค่าน้อยเนื่องจากข้อมูลมีคลาส ของสินค้าที่ไม่เป็น backorders มากกว่าทำให้ยากที่จะทำนายว่าเป็นสินค้าที่เป็น backorders แต่เมื่อเปรียบเทียบกับอัลกอริทึมอื่นๆ ทำให้ทราบ อัลกอริทึม Random Forest มีค่า Recall มากกว่าอัลกอริทึม K-nearest neighbor และ Adaboost

4.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม Random Forest

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 30 พบว่า ค่า F1 score ที่เฉลี่ยรวมของอัลกอริทึม Random Forest ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.9927 คิดเป็นร้อยละ 99.27 ซึ่ง ดังนั้น จึงอยู่ในเกณฑ์ที่ ค่า F1 score ที่แสดงถึงความแม่นยำสูง เพราะมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 แต่เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่าค่า F1 score ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.9972 คิดเป็น 99.72 ส่วนคลาสของสินค้าที่เป็นสินค้า backorders มีค่า 0.3275 คิดเป็นร้อยละ 32.75 แสดงให้เห็นว่าค่า F1 score ของคลาสที่เป็นสินค้าที่เป็น backorders มีความแม่นยำโดยเฉลี่ยมีค่าน้อย

	precision	recall	f1-score	support
0	0.9947	0.9996	0.9972	502970
1	0.7977	0.2060	0.3275	3388
avg / total	0.9934	0.9943	0.9927	506358

ภาพประกอบ 30 ผลลัพธ์จาก classification report ของ Random Forest

4.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม Random Forest

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

4.5.1 True Positive (TP) ของอัลกอริทึม Random Forest

คือ จำนวนข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders (TP) มีค่าเท่ากับ 698 แสดงว่าค่าที่ทำนายเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีค่าน้อย แสดงว่ามีจำนวนของข้อมูลที่ทำนายถูกจำนวนน้อยและแสดงถึงความแม่นยำน้อยแต่เมื่อ

เปรียบเทียบกับอัลกอริทึมอื่นๆ เช่น อัลกอริทึม K-nearest neighbor พบว่า มีค่าการทำนายถูกน้อยกว่า อัลกอริทึม Random Forest

4.5.2 True Negative (TN) ของอัลกอริทึม Random Forest

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น Backorders มีค่า 502,793 ซึ่งสมเหตุสมผลกับค่าจริงที่มีข้อมูลนี้มีสินค้าที่ไม่เป็น backorders จำนวนมากจากจำนวนสินค้าที่ไม่เป็น backorders ทั้งหมด ดังนั้นความแม่นยำของอัลกอริทึมนี้เมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders ส่วนใหญ่ทำนายได้ถูกต้องและแสดงถึงประสิทธิภาพประกอบดีมากในการจำแนก

4.5.3 False Positive (FP) ของอัลกอริทึม Random Forest

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders มีค่าเท่ากับ 177 แสดงว่า เมื่อพิจารณาจากคลาสที่ไม่เป็น backorders พบว่าในความเป็นจริงแล้วควรจะต้องทำนายว่าไม่เป็น backorders แต่ทำนายไม่ถูกต้อง แต่ค่าที่ทำนายไม่ถูกต้องเป็นเพียงส่วนน้อย ซึ่งค่าที่ทำนายไม่ถูกต้องน้อยที่สุดเมื่อเปรียบเทียบกับอัลกอริทึมทั้งหมด แต่มากกว่า Decision Tree

4.5.4 False Negative (FN) ของอัลกอริทึม Random Forest

คือ จำนวนข้อมูลที่ค่าจริงเป็นสินค้าที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 2,690 หมายความว่า แสดงว่ามีจำนวนของสินค้าที่เป็น backorders ที่มีการทำนายที่ไม่ถูกต้องมีจำนวนมาก แต่เมื่อเปรียบเทียบกับอัลกอริทึม AdaBoost พบว่า AdaBoost มีจำนวนของการทำนายไม่ถูกต้องมากกว่า ดังนั้นความแม่นยำอยู่ในเกณฑ์ที่พอใช้

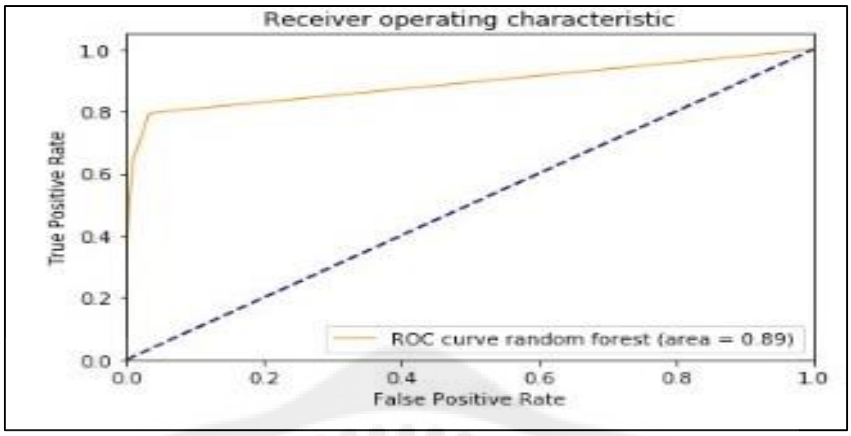
ตาราง 7 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ Random Forest

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	698 (TP)	2,690 (FN)
ไม่เป็น backorders	177 (FP)	502,793 (TN)

4.6 ค่ากราฟ ROC Curve ของอัลกอริทึม Random Forest

จากการประมวลผลคำสั่งในการเปรียบเทียบค่าของพื้นที่ใต้กราฟ ROC Curve (AUC) ของอัลกอริทึม Random Forest เท่ากับ 0.89 ดังภาพประกอบ 31 โดยค่าที่จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายต้องมีค่าระหว่างร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่าพื้นที่ใต้กราฟของอัลกอริทึม Random Forest มีค่าอยู่ในเกณฑ์ที่แสดงถึงประสิทธิภาพประกอบดีในการ

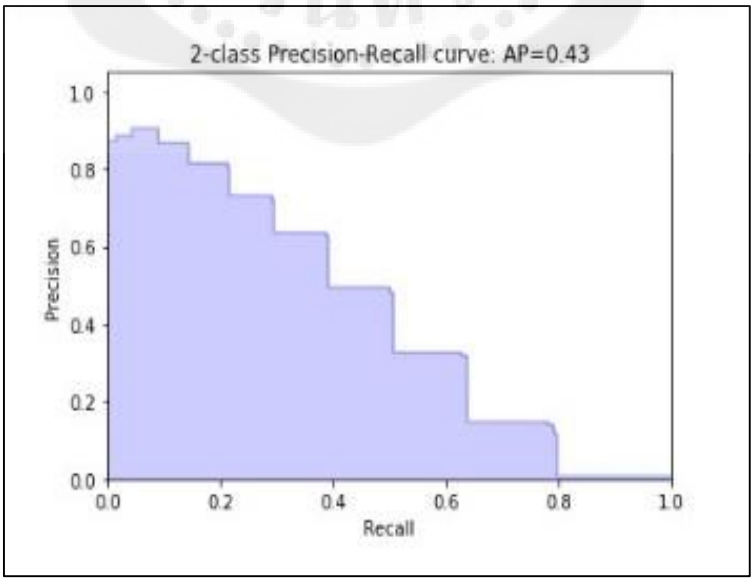
ทำนาย ดังนั้นเมื่อทำการเปรียบเทียบค่าพื้นที่ใต้กราฟ จะเห็นว่า Random Forest มีความแม่นยำสูงกว่าอัลกอริทึม Decision Tree และ Naive Bayes



ภาพประกอบ 31 กราฟ ROC Curve ของ Random Forest

4.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม Random forest

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม Random Forest พบว่ามีค่าเท่ากับ 0.43 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อยแต่เมื่อเปรียบเทียบกับ Naive Bayes Decision tree และ K-nearest neighbors พบว่าให้ค่า Average Precision ของ Random Forest มีค่าสูงกว่า สามารถแสดงเป็นกราฟได้ดังภาพประกอบ 32



ภาพประกอบ 32 กราฟ Precision-Recall ของ Random Forest

5. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม Adaptive Boosting (AdaBoost)

5.1 ค่า Accuracy ของอัลกอริทึม AdaBoost

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม AdaBoost กำหนดพารามิเตอร์ตามค่าตั้งต้นของไลบารี sklearn ดังภาพประกอบ 33 ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้อง 0.9931 หรือคิดเป็นร้อยละ 99.31 ซึ่งค่าที่แสดงถึงประสิทธิภาพประกอบดีในการทำนายต้องเป็นค่าที่เข้าใกล้ 1 คิดหรือคิดเป็นร้อยละ 100 แสดงว่าอัลกอริทึมนี้จึงมีค่าความถูกต้องสูง และใช้เวลาในการประมวลผล 308.48 วินาที ซึ่งในเวลาในการประมวลผลนานแต่ให้ค่าความถูกต้องที่สูง

```
In [65]: from sklearn.ensemble import AdaBoostClassifier
         adbmodel = AdaBoostClassifier()
         adbmodel.fit(X_features_train, y_labels_train)

Out[65]: AdaBoostClassifier(algorithm='SAMME.R', base_estimator=None,
                    learning_rate=1.0, n_estimators=50, random_state=None)
```

ภาพประกอบ 33 ค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ AdaBoost

5.2 ค่า Precision ของอัลกอริทึม AdaBoost

ผลจากการสรุปรายงานการจำแนก (Classification report) พบว่า ผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision เท่ากับ 0.989 คิดเป็นร้อยละ 98.90 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่เป็น backorders มีค่าเท่ากับ 0.334 คิดเป็นร้อยละ 33.40 ในขณะที่ ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่าเท่ากับ 0.994 คิดเป็นร้อยละ 99.40 ดังนั้นเนื่องจากจำนวนของสินค้าที่ไม่เป็น backorders มีจำนวนมาก ถ้าใช้ค่า Precision ของสินค้าที่ไม่เป็น backorders เป็นตัวชี้วัดอาจให้ผลลัพธ์ไม่ชัดเจนจึงใช้ค่า Precision ของสินค้าที่เป็น backorders เป็นตัวชี้วัดและพบว่าค่า Precision ของสินค้าที่เป็น backorders แสดงถึงความแม่นยำน้อย แต่หากเปรียบเทียบกับอัลกอริทึมอื่น เช่น Naive Bayes พบว่ามีค่าความแม่นยำจากคลาสที่พิจารณาจากคลาสของสินค้าที่เป็น backorders มากกว่าแต่ยังน้อยกว่าอัลกอริทึม Random Forest

5.3 ค่า Recall ของอัลกอริทึม AdaBoost

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 34 พบว่า ค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม AdaBoost มีค่า 0.993 คิดเป็นร้อยละ 99.30 ซึ่งค่า Recall นี้มีค่าสูง สามารถอธิบายได้ว่า อัลกอริทึม AdaBoost มีความแม่นยำมาก เพราะมีค่าของผลลัพธ์เนื่องจากความเข้าใกล้ 1 หรือร้อยละ 100 ซึ่งเป็นค่าที่แสดงถึงประสิทธิภาพประกอบดีในการทำนาย แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 1.000 คิดเป็นร้อยละ 100 ซึ่งเป็นค่าที่มีค่าสูง ในขณะที่ คลาส

ของสินค้าที่เป็นสินค้า backorders มีค่า Recall 0.032 คิดเป็นร้อยละ 3.20 แสดงให้เห็นว่าผลลัพธ์ที่ได้จากการทำนายที่พิจารณาจากคลาสที่เป็นสินค้า backorders จึงมีความแม่นยำน้อย

5.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม AdaBoost

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 34 พบว่า ค่า F1 score ที่เฉลี่ยรวมของอัลกอริทึม Random Forest ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.990 คิดเป็นร้อยละ 99 ซึ่ง ดังนั้น จึงอยู่ในเกณฑ์ที่แสดงถึงความแม่นยำในการทำนายสูง เพราะมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 แต่เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1)และ ไม่เป็น backorders (0) พบว่าค่า F1 score ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.997 คิดเป็น 99.70 ส่วนคลาสของสินค้าที่เป็นสินค้า backorders มีค่า 0.059 คิดเป็นร้อยละ 5.90 แสดงให้เห็นว่า ค่า F1 score ของคลาสที่เป็นสินค้าที่เป็น backorders มีความแม่นยำโดยเฉลี่ยมีค่าน้อย

	precision	recall	f1-score	support
0	0.994	1.000	0.997	502970
1	0.334	0.032	0.059	3388
avg / total	0.989	0.993	0.990	506358

ภาพประกอบ 34 ผลลัพธ์จาก classification report ของ AdaBoost

5.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม AdaBoost

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

5.5.1 True Positive (TP) ของอัลกอริทึม AdaBoost

คือ จำนวนของข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders มีค่าเท่ากับ 110 จากจำนวนของข้อมูลที่ไม่เป็น แสดงว่าค่าที่ทำนายเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีจำนวนของข้อมูลที่ทำนายถูกจำนวนน้อยและมีประสิทธิภาพในการทำนายที่แม่นยำน้อย

5.5.2 True Negative (TN) ของอัลกอริทึม AdaBoost

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders และทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 502,751 ซึ่งเมื่อพิจารณาค่าจริงกับค่าที่ทำนายจากคลาสของสินค้าที่ไม่เป็น backorders มีค่าสูงและสมเหตุสมผลเพราะมีสินค้าที่ไม่เป็น backorders จำนวนมากกว่าสินค้าที่เป็น backorders แสดงว่ามีการทำนายผิดมีจำนวนน้อยแสดงว่ามีความแม่นยำของอัลกอริทึมจำนวนมาก

5.5.3 False Positive (FP) ของอัลกอริทึม AdaBoost

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders

มีค่าเท่ากับ 219 แสดงว่า เมื่อพิจารณาจากคลาสที่ไม่เป็น backorders พบว่าในความเป็นจริงแล้วควรจะต้องทำนายว่าไม่เป็น backorders แต่ทำนายไม่ถูกต้อง แต่ค่าที่ทำนายไม่ถูกต้องเป็นส่วนน้อย แต่เมื่อเปรียบเทียบกับ อัลกอริทึม Decision Tree และ Naive Bayes พบว่า Decision Tree และ Naive Bayes ทำนายผิดมากกว่า

5.5.4 False Negative (FN) ของอัลกอริทึม AdaBoost

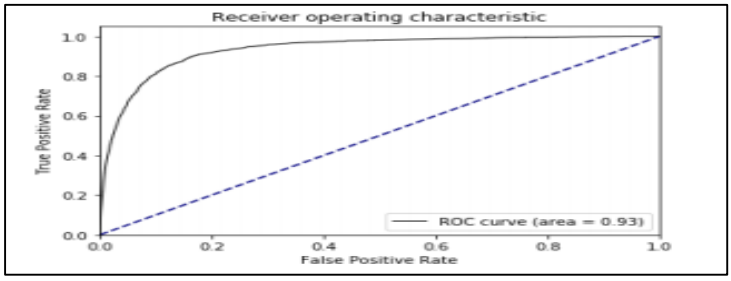
คือ จำนวนข้อมูลที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 3,278 หมายความว่า ในความเป็นจริงแล้วค่าจริงเป็นสินค้า backorders ควรจะทำนายว่าเป็นสินค้า backorders แต่กลับทำนายว่าไม่เป็นสินค้า backorders ซึ่งไม่ถูกต้อง แต่ค่าการทำนายที่ไม่ถูกต้องมีจำนวนปานกลาง แต่เมื่อเทียบกับอัลกอริทึม Naive Bayes Decision tree K-nearest neighbor และ Random Forest พบว่ามีการทำนายผิดมากกว่า

ตาราง 8 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ AdaBoost

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	110 (TP)	3,278 (FN)
ไม่เป็น backorders	219 (FP)	502,751 (TN)

5.6 กราฟ ROC Curve ของอัลกอริทึม AdaBoost

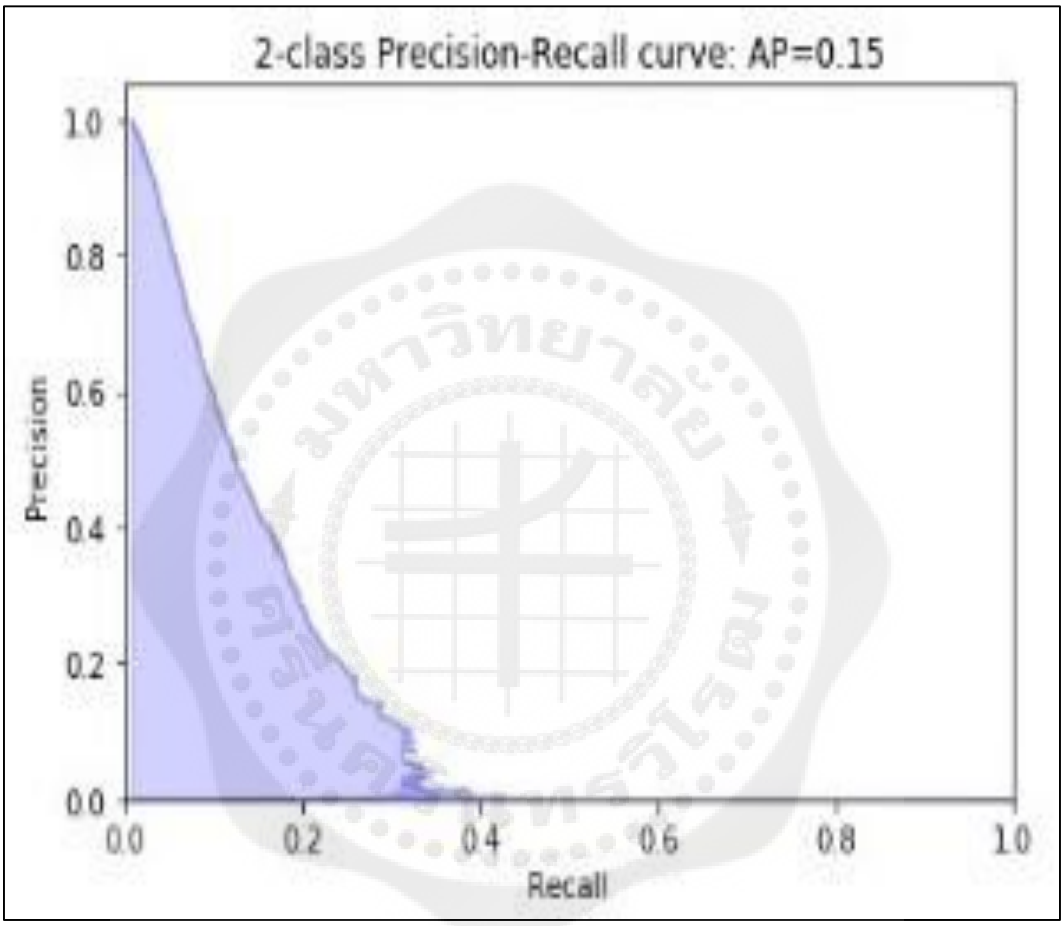
จากการประมวลผลคำสั่งในการเปรียบเทียบค่าของพื้นที่ใต้กราฟ ROC Curve (AUC) ของอัลกอริทึม AdaBoost เท่ากับ 0.93 จากภาพประกอบ 35 โดยค่าที่จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายที่ดีต้องมีค่าระหว่างช่วงร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่าพื้นที่ใต้กราฟของอัลกอริทึม AdaBoost มีค่าอยู่ในเกณฑ์ที่แสดงถึงประสิทธิภาพประกอบในการทำนายดี ดังนั้นเมื่อเปรียบเทียบค่าพื้นที่ใต้กราฟ จะเห็นว่า AdaBoost มีความแม่นยำสูงกว่าทุกอัลกอริทึม ยกเว้น XGBoost



ภาพประกอบ 35 กราฟ ROC Curve ของ AdaBoost

5.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม AdaBoost

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม AdaBoost พบว่า มีค่าเท่ากับ 0.15 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อย เมื่อเปรียบเทียบกับ ของ Random Forest พบว่าให้ค่า Average Precision ของ Random Forest มีค่าสูงกว่า สามารถแสดงเป็นกราฟ Precision-Recall ดังภาพประกอบ 36



ภาพประกอบ 36 กราฟ Precision-Recall ของ AdaBoost

6. ผลลัพธ์ของการวัดประสิทธิภาพของแบบจำลองของอัลกอริทึม Extreme Gradient Boosting (XGBoost)

6.1 ค่า Accuracy ของอัลกอริทึม XGBoost

จากการกำหนดค่าพารามิเตอร์ของอัลกอริทึม XGBoost กำหนดพารามิเตอร์ตามค่าตั้งต้นของไลบารี sklearn ดังภาพประกอบ 37 ในการสร้างแบบจำลอง พบว่าได้ค่าความถูกต้อง 0.9933 หรือคิดเป็นร้อยละ 99.33 ซึ่งค่าถูกต้องอยู่ในช่วงค่าความถูกต้องที่เข้าใกล้ 1 คิดเป็นร้อยละ 100 แสดงว่าอัลกอริทึมนี้จึงมีค่าความถูกต้องสูง และใช้เวลาในการประมวลผล 209.64 วินาที

```
import xgboost as xgb
from sklearn.preprocessing import LabelEncoder
import numpy as np
xgbmodel = xgb.XGBClassifier()

xgbmodel = xgbmodel.fit(X_features_train, y_labels_train)

xgbmodel

XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,
       max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
       n_jobs=1, nthread=None, objective='binary:logistic', random_state=0,
       reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
       silent=True, subsample=1)
```

ภาพประกอบ 37 ค่าพารามิเตอร์สำหรับสร้างแบบจำลองของ XGBoost

6.2 ค่า Precision ของอัลกอริทึม XGBoost

ผลจากการสรุปรายงานการจำแนก (Classification report) ระหว่างค่าจริงที่ผ่านการสุ่มข้อมูลแบบ Stratified Shuffle Split และค่าที่ทำนายได้ของอัลกอริทึม XGBoost พบว่า ผลเฉลี่ยรวมของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่า Precision เท่ากับ 0.989 คิดเป็นร้อยละ 98.90 แต่หากพิจารณาค่า Precision เฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และไม่เป็น backorders (0) พบว่า ค่า Precision ของสินค้าที่เป็น backorders มีค่า 0.394 คิดเป็นร้อยละ 39.40 ในขณะที่ ค่า Precision ของสินค้าที่ไม่เป็น backorders มีค่า 0.993 คิดเป็นร้อยละ 99.30 เนื่องจากค่า Precision ที่ใช้วัดความแม่นยำของแบบจำลองโดยจะพิจารณาจากคลาสที่ของสินค้าที่เป็น backorders แสดงค่าที่มีค่าน้อย จึงมีความแม่นยำน้อย ดังภาพประกอบ 38

6.3 ค่า Recall ของอัลกอริทึม XGBoost

เมื่อพิจารณาจากผลลัพธ์แสดงในภาพประกอบ 38 พบว่า ค่า Recall ที่เฉลี่ยรวมของอัลกอริทึม XGBoost มีค่า 0.993 คิดเป็นร้อยละ 99.30 ซึ่งค่า Recall นี้มีค่าสูง สามารถอธิบายได้ว่าอัลกอริทึม XGBoost มีความแม่นยำมาก เพราะมีค่าของผลลัพธ์เนื่องจากความเข้าใกล้ 1 หรือร้อยละ 100 ซึ่งเป็นค่าที่แสดงถึงประสิทธิภาพประกอบดีในการทำนาย แต่หากพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่า ค่า Recall ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 1.000 คิดเป็นร้อยละ 100 ในขณะที่ คลาสของสินค้าที่เป็นสินค้า backorders มีค่า Recall 0.008 คิดเป็นร้อยละ 8 ซึ่งเกณฑ์ที่ใช้เป็นตัวชี้วัดสำหรับการวัดประสิทธิภาพของอัลกอริทึมด้วยค่า Recall สำหรับอัลกอริทึม XGBoost แสดงให้เห็นว่ามีความแม่นยำที่น้อยมาก

6.4 ค่า F-Measure หรือ F1 score ของอัลกอริทึม XGBoost

เมื่อพิจารณาจากผลลัพธ์ที่แสดงในภาพประกอบ 38 พบว่า ค่า F1 score ที่เฉลี่ยรวม

ของอัลกอริทึม XGBoost ของสินค้าที่เป็น backorders และ ไม่เป็น backorders มีค่าเท่ากับ 0.990 คิดเป็นร้อยละ 99 ซึ่ง อยู่ในเกณฑ์มีความแม่นยำสูง เพราะมีค่าเข้าใกล้ 1 หรือเข้าใกล้คิดเป็นร้อยละ 100 แต่เมื่อพิจารณาเฉพาะคลาสระหว่างสินค้าที่เป็น backorders (1) และ ไม่เป็น backorders (0) พบว่าค่า F1 score ของสินค้าที่ไม่เป็น backorders มีค่าความแม่นยำ 0.9970 คิดเป็น 99.70 ส่วนคลาสของสินค้าที่เป็นสินค้า backorders มีค่า 0.016 คิดเป็นร้อยละ 1.60 แสดงให้เห็นว่า ค่า F1 score ของคลาสที่เป็นสินค้าที่เป็น backorders แสดงถึงความแม่นยำน้อย

	precision	recall	f1-score	support
0	0.993	1.000	0.997	502970
1	0.394	0.008	0.016	3388
avg / total	0.989	0.993	0.990	506358

ภาพประกอบ 38 ผลลัพธ์จาก classification report ของ XGBoost

6.5 ค่าจากตาราง confusion matrix ของอัลกอริทึม XGBoost

พิจารณาตารางเมตริกซ์ เพื่อเปรียบเทียบระหว่างค่าจริงและค่าที่ทำนายแสดงผลดังนี้

6.5.1 True Positive (TP) ของอัลกอริทึม XGBoost

คือ จำนวนข้อมูลที่ค่าจริงเป็น backorders และทำนายว่าเป็น backorders มีค่าเท่ากับ 28 แสดงว่าค่าที่ทำนายเมื่อเปรียบเทียบกับค่าจริงของคลาสที่เป็น backorders มีค่าที่ทำนายถูกน้อยมาก จากจำนวนข้อมูลทั้งหมด แสดงว่ามีอัลกอริทึมนี้มีประสิทธิภาพและความแม่นยำน้อย

6.5.2 True Negative (TN) ของอัลกอริทึม XGBoost

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็นสินค้า backorders และค่าที่ทำนายไม่เป็น backorders มีค่าเท่ากับ 502,927 ซึ่งเมื่อเปรียบเทียบค่าที่ทำนายกับค่าจริงของคลาสที่ไม่เป็น backorders ที่มีมากกว่าสินค้าที่เป็น backorders พบว่าผลลัพธ์มีความสมเหตุสมผล และแสดงถึงความแม่นยำที่มีค่าสูง

6.5.3 False Positive (FP) ของอัลกอริทึม XGBoost

คือ จำนวนข้อมูลที่ค่าจริงไม่เป็น backorders แต่ทำนายว่าเป็น backorders มีค่า
เท่ากับ 43 แสดงว่า เมื่อพิจารณาจากคลาสของสินค้าที่ไม่เป็น backorders พบว่าในความเป็นจริงแล้วควรจะต้องทำนายว่าไม่เป็น backorders แต่ทำนายไม่ถูกต้องเป็นส่วนน้อยแสดงว่าส่วนมากทำนายถูกต้องแสดงว่ามีความแม่นยำ

6.5.4 False Negative (FN) ของอัลกอริทึม XGBoost

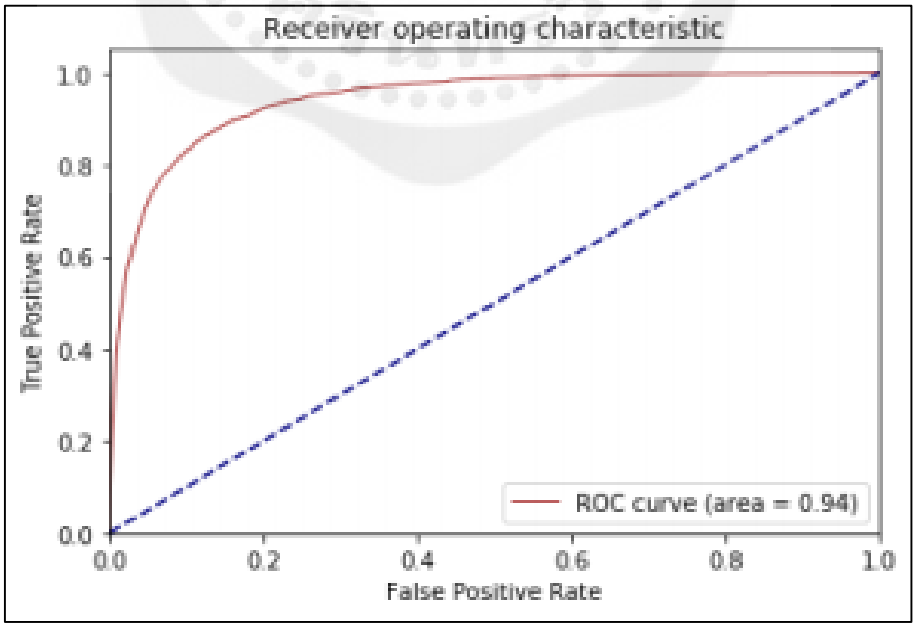
คือ จำนวนข้อมูลที่เป็น backorders แต่ทำนายว่าไม่เป็น backorders มีค่าเท่ากับ 3,360 หมายความว่า ในความเป็นจริงแล้วค่าจริงเป็นสินค้า backorders ควรจะทำนายว่าเป็นสินค้า backorders แต่กลับทำนายว่าไม่เป็นสินค้า backorders จำนวน 3,360 รายการจากสินค้าทั้งหมด ซึ่งให้ค่าการทำนายที่ไม่ถูกต้องสูง

ตาราง 9 ผลจากการเปรียบเทียบระหว่างค่าจริงกับค่าที่ทำนายของ XGBoost

ค่าที่แท้จริง	ค่าที่ทำนาย	
	เป็น backorders	ไม่เป็น backorders
เป็น backorders	28 (TP)	3,360 (FN)
ไม่เป็น backorders	43 (FP)	502,927 (TN)

6.6 ค่ากราฟ ROC Curve ของอัลกอริทึม XGBoost

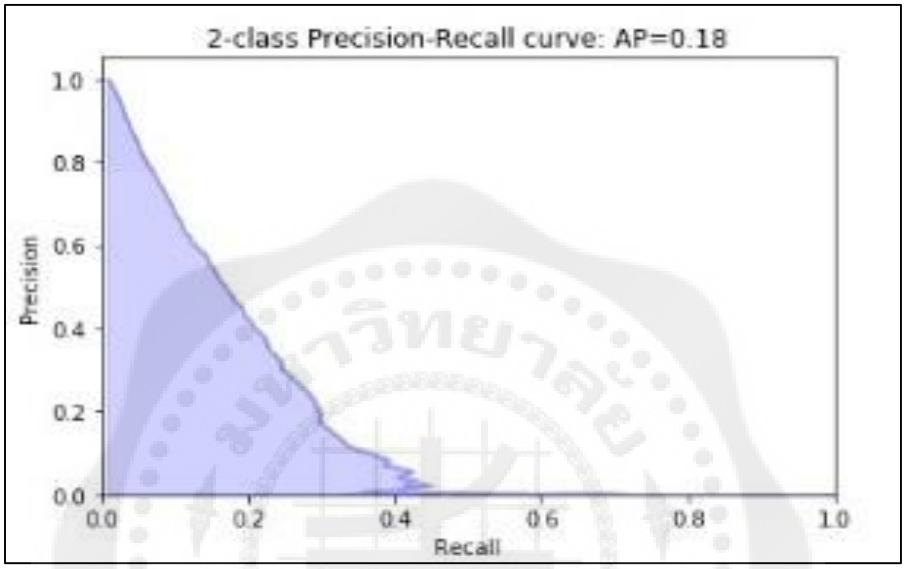
จากการประมวลผลคำสั่งในการเปรียบเทียบค่าของพื้นที่ใต้กราฟ ROC Curve (AUC) ของอัลกอริทึม XGBoost เท่ากับ 0.94 ดังภาพประกอบ 39 โดยค่าที่จะบ่งบอกถึงประสิทธิภาพของแบบการทำนายต้องมีค่าระหว่างร้อยละ 80 ถึงร้อยละ 95 ซึ่งค่าที่ได้จากค่าพื้นที่ใต้กราฟของอัลกอริทึม XGBoost มีค่าอยู่ในเกณฑ์ที่แสดงถึงประสิทธิภาพประกอบดีในการทำนาย ดังนั้นเมื่อทำการเปรียบเทียบค่าพื้นที่ใต้กราฟ XGBoost มีค่าสูงที่สุดเมื่อเทียบกับอัลกอริทึมอื่น



ภาพประกอบ 39 กราฟ ROC Curve ของ XGBoost

6.7 ค่า Average Precision และกราฟ Precision-Recall ของอัลกอริทึม XGBoost

จากการแสดงผลของค่า Average Precision ของ อัลกอริทึม XGBoost พบว่ามีค่าเท่ากับ 0.18 ซึ่งแสดงถึงค่าที่มีความแม่นยำน้อยกว่า เมื่อเปรียบเทียบกับ ของ Random Forest เพราะค่า Average Precision ของ XGBoost มีค่าน้อยกว่าสามารถแสดงเป็นกราฟ Precision-Recall ได้ดังภาพประกอบ 40



ภาพประกอบ 40 กราฟ Precision-Recall ของ อัลกอริทึม XGBoost

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

1. สรุปผลการวิจัย

จากการทดสอบประสิทธิภาพของแบบจำลองพบว่าประสิทธิภาพในการจำแนกของอัลกอริทึมที่ทดสอบค่าความถูกต้อง(Accuracy) ที่มากที่สุด คือ Random Forest มีค่าร้อยละ 99.43 แต่ใช้เวลาในการคำนวณ 96.16 วินาที ส่วนของตัวชี้วัดอื่นๆได้ แก่ Precision Recall และ F1-Score โดยคำนวณจากผลรวมทั้งสินค้าที่เป็น backorders และ ไม่เป็น backorders พบว่าทุกอัลกอริทึมแสดงค่าความถูกต้องออกมาได้สูงมากซึ่งไม่สะท้อนถึงความเป็นจริงอาจทำให้ส่งผลต่อการนำไปใช้ในการตัดสินใจที่ผิด จึงต้องทำการพิจารณาจำแนกเฉพาะคลาสอย่างเป็นอิสระต่อกัน พบว่าอัลกอริทึมที่มีประสิทธิภาพในการทำนายเฉพาะคลาสที่เป็นสินค้า backorders โดยผลลัพธ์ที่แสดงค่าสูงที่สุด เมื่อพิจารณาจากค่า Precision คือ Random Forest คิดเป็นร้อยละ 79.77 รองลงมาเป็น K-nearest neighbor คิดเป็นร้อยละ 51.40 ส่วน ค่า Recall ของอัลกอริทึมที่มีค่าสูงที่สุดคือ Naïve Bays คิดเป็นร้อยละ 98.60 รองลงมาเป็น Decision Tree คิดเป็นร้อยละ 37.50 และเมื่อพิจารณาจาก ค่า F1-Score พบว่าอัลกอริทึมที่มีค่าสูงที่สุดคือ Decision tree คิดเป็นร้อยละ 37.10 รองลงมาคือ Random forest คิดเป็นร้อยละ 32.75

ตาราง 10 ผลการทดสอบประสิทธิภาพการจำแนกประเภทของอัลกอริทึมทั้งหมด

อัลกอริทึม	เวลาที่ใช้ (วินาที)	Accuracy	Precision	Recall	F1-Score	ROC (AUC)	AP
Naive Bayes	2.26*	60.10%	99.20%	6.00%	10.20%	0.60	0.01
Decision Tree	39.43	99.15%	99.20%	99.10%	99.20%	0.71	0.16
KNN	3,048.89	99.15%	99.10%	99.30%	99.20%	0.84	0.24
Random Forest	96.16	99.43%*	99.34%*	99.43%*	99.27%*	0.89	0.43*
AdaBoost	308.48	99.31%	98.90%	99.30%	99.00%	0.93	0.15
XGBoost	348.44	99.33%	98.90%	99.30%	99.00%	0.94*	0.18

ตาราง 11 ผลการทดสอบประสิทธิภาพการจำแนกของสินค้าที่เป็น backorders

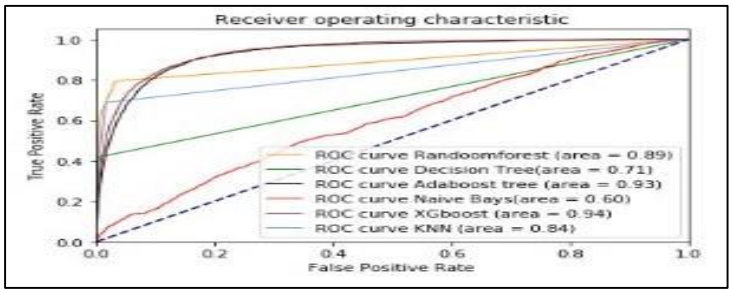
	Naive Bayes	Decision Tree	KNN	Random Forest	Adaboost	XGBoost
Precision	0.70%	36.60%	51.40%	79.77%*	33.40%	39.40%
Recall	98.60%*	37.50%	15.30%	20.60%	3.20%	0.80%
F1-score	1.40%	37.10%*	23.60%	32.75%	5.90%	1.60%

ตาราง 12 ผลการทดสอบประสิทธิภาพการจำแนกของสินค้าที่ไม่เป็น backorders

	Naive Bayes	Decision Tree	KNN	Random Forest	Adaboost	XGBoost
Precision	99.80%*	99.60%	99.40%	99.47%	99.40%	99.30%
Recall	5.40%	99.60%	99.90%	99.96%	100%*	100%*
F1-score	10.20%	99.60%	99.70%	99.72%*	99.70%	99.70%

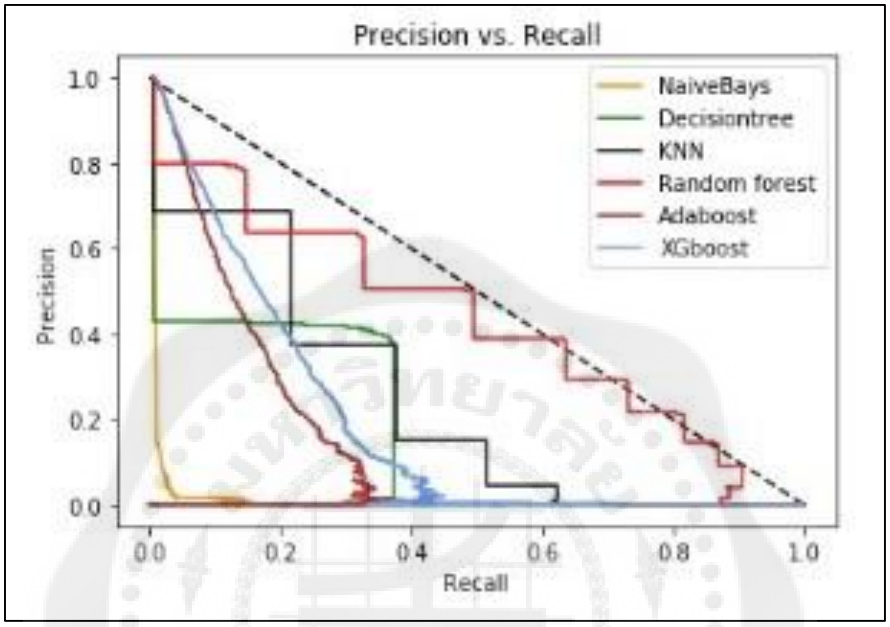
หมายเหตุ: *คือค่าของผลลัพธ์ที่ดีที่สุด

จากผลการทดสอบประสิทธิภาพโดยการใช้พื้นที่ใต้กราฟ ROC Curve (AUC) พบว่า Extreme Gradient Boosting มีค่าพื้นที่ใต้กราฟมากที่สุดคิดเป็นร้อยละ 94 รองลงมาเป็น AdaBoost คิดเป็นร้อยละ 93 และ Random Forest คิดเป็นร้อยละ 89 ทำให้แสดงให้เห็นว่าอัลกอริทึม Random Forest ให้ค่าการทดสอบประสิทธิภาพจากการวัดโดยใช้ค่าพื้นที่ใต้กราฟ ROC Curve อยู่ในเกณฑ์ที่สูงแม้จะไม่สูงที่สุดยังเป็นค่าที่อยู่ในเกณฑ์ที่ดีและเข้าใกล้ 1 (ภาพประกอบ 41)



ภาพประกอบ 41 ผลการเปรียบเทียบประสิทธิภาพโดยการใช้พื้นที่ใต้กราฟ ROC Curve (AUC)

จากผลการทดสอบประสิทธิภาพในการทำนายโดยใช้ ค่า Average Precision และกราฟ Precision-Recall ของแต่ละอัลกอริทึมพบว่าอัลกอริทึมที่แสดงผลลัพธ์ที่มีค่าสูงที่สุดคืออัลกอริทึม Random Forest มีค่าเท่ากับ 0.42 แสดงว่า ค่าความแม่นยำเฉลี่ยของอัลกอริทึม Random Forest มีค่าความแม่นยำมากที่สุด

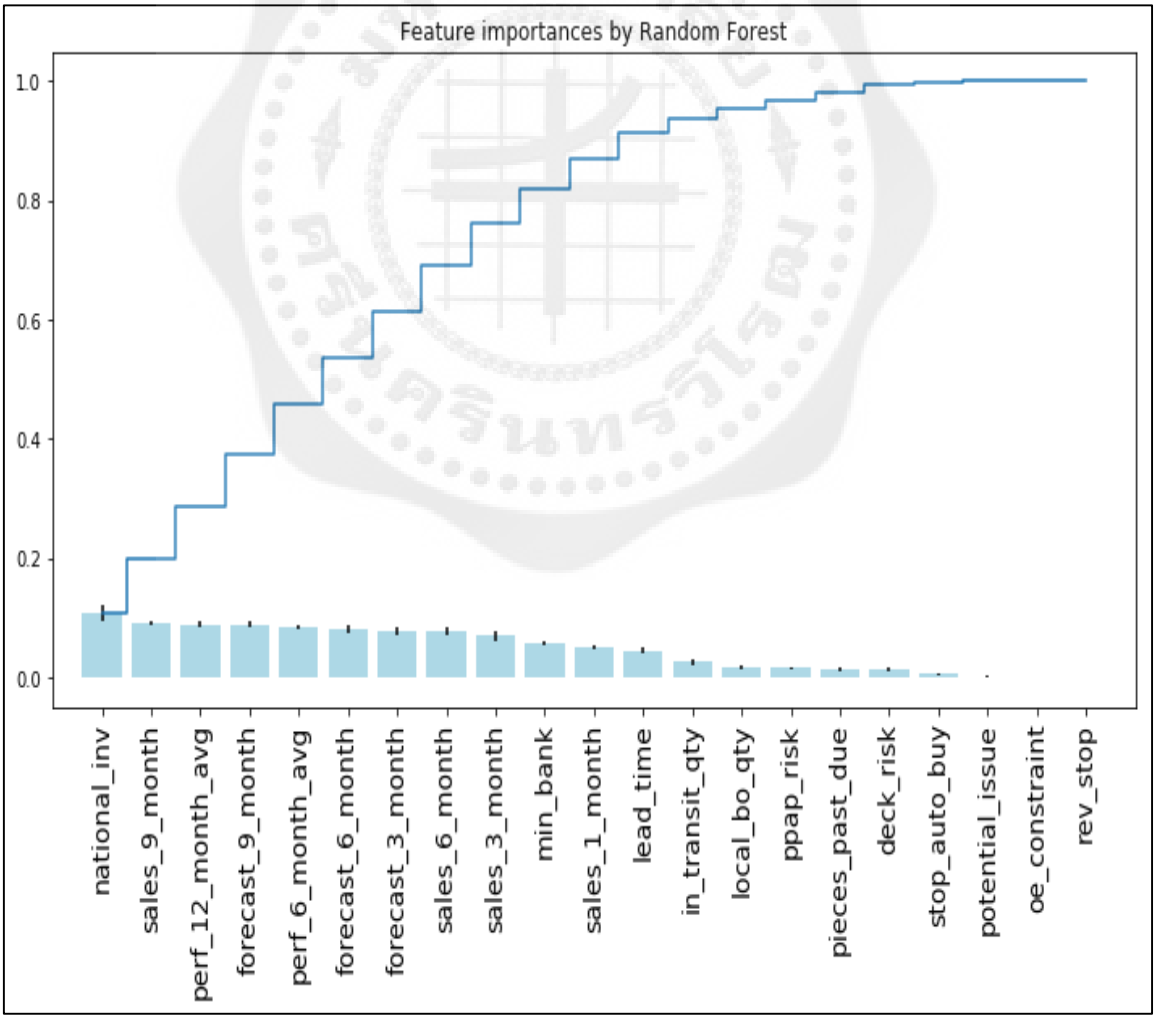


ภาพประกอบ 42 ผลการเปรียบเทียบประสิทธิภาพโดยการใช้พื้นที่ใต้กราฟ Precision-Recall

จากภาพประกอบ 41 และภาพประกอบ 42 แสดงให้เห็นว่าการใช้พื้นที่ใต้กราฟ ROC Curve อาจทำให้ได้ผลลัพธ์สำหรับวัดประสิทธิภาพสำหรับแบบจำลองของอัลกอริทึมแสดงผลลัพธ์ที่ยังไม่ชัดเจน เนื่องจากมีบางเส้นที่แสดงในกราฟ ROC Curve ของบางอัลกอริทึม เช่น AdaBoost และ XGBoost มีความใกล้เคียงกันมากจนไม่สามารถที่จะแยกออกได้อย่างชัดเจนว่าค่าใดแสดงผลลัพธ์ที่ดีที่สุด จึงใช้ค่าจากกราฟ Precision-Recall curve เข้ามาร่วมใช้ในการพิจารณาเพราะจะทำให้มีความชัดเจนมากขึ้นและอธิบายถึงความแตกต่างของอัลกอริทึมได้ดี [18] เนื่องจากกราฟ ROC Curve ไม่สามารถใช้ประกอบการพิจารณาสำหรับกรณีที่ข้อมูลมีความไม่สมดุลสูง เพราะว่าอัตราส่วนของ False Positive Rate (False Positives หารด้วยผลรวมของจำนวนข้อมูลที่ค่าจริงไม่เป็นสินค้า backorders) ทำให้ไม่เห็นได้อย่างชัดเจน ในกรณีที่จำนวนรวมของค่าจริงที่ไม่เป็นสินค้า backorders มีจำนวนมาก แต่ว่า ค่า Precision มีความไวต่อ False Positives (การทำนายผิดว่าเป็นสินค้า backorders) ไม่ได้มีผลมาจากการมีสินค้าที่ไม่เป็น backorders ที่มีจำนวนมาก จึงเหมาะสมที่จะใช้ Precision ในกราฟ Precision-Recall มากกว่า False Positive Rate ในกราฟ ROC Curve

เนื่องจากสัดส่วนของข้อมูลมีความแตกต่างกันมากของสินค้าที่เป็น backorders และ

สินค้าที่ไม่เป็น backorders ทำให้ผลการทดสอบประสิทธิภาพของอัลกอริทึม ที่แยกเฉพาะคลาสสินค้าที่เป็น backorders จึงมีค่าน้อย ดังตาราง 11 แต่เมื่อนำผลของทั้งสองคลาสมาเฉลี่ยกันทำให้ได้ค่าความถูกต้องมีค่าที่สูง แต่อย่างไรก็ตามอัลกอริทึม Random Forest ก็ยังคงเป็นอัลกอริทึมที่มีความถูกต้องและประสิทธิภาพในการทำนายดีเมื่อเทียบกับอัลกอริทึมอื่นเมื่อพิจารณาจากค่าของตัวชี้วัดที่นำมาทดสอบและพบว่า Features ที่มีความสำคัญต่อการเกิด backsorders มากที่สุดของอัลกอริทึม Random Forest คือ Features national_inv ซึ่งหมายถึง ระดับของสินค้าคงคลังที่มี ณ ปัจจุบัน โดยหมายความถ้าระดับของสินค้าคงคลังที่มี ณ ปัจจุบัน มีค่าน้อยก็ทำให้ส่งผลต่อการเกิดกรณีสินค้าค้างส่งซึ่งสมเหตุสมผลกับหลักความเป็นจริง และ รองลงมาเป็น sales_9_month หมายถึง ยอดขายของสินค้าช่วง 9 เดือน ซึ่งเป็นยอดขายที่เกิดขึ้นในระยะยาว โดยการขายสินค้าในระยะยาวมักเกิดสถานการณ์การเกิดสินค้าค้างส่งได้ สาเหตุเกิดจากหลายกรณี เช่น ผลิตสินค้าไม่ทัน หรือ สินค้าที่สำรองไม่เพียงพอต่อความต้องการเพราะมีคำสั่งซื้อสินค้ามีสูงขึ้น เป็นต้น ดังภาพประกอบ 43



ภาพประกอบ 43 กราฟแสดงลำดับความสำคัญของ Feature (Feature importances)

โดยสรุปจากการศึกษาในงานวิจัยนี้สิ่งที่ได้รับจากการศึกษาในครั้งนี้คือ

1.1 ผู้วิจัยได้ศึกษาการสร้างแบบจำลอง (Modeling) ในกรณีของจำแนกประเภทสินค้าค้างส่ง (backorders) ด้วยเทคนิคการเรียนรู้ของเครื่อง (Machine learning) นี้ผู้วิจัยได้ทำการศึกษาการสร้างแบบจำลองด้วยอัลกอริทึมดังต่อไปนี้คือ Naive Bays Decision tree K-nearest neighbors Random Forest AdaBoost และ Extreme Gradient Boosting

1.2 ผู้วิจัยได้ทำการศึกษาการเปรียบเทียบประสิทธิภาพของอัลกอริทึมที่ใช้ในการทำนายสินค้าคงคลังรวมไปถึงเทคนิคที่ใช้ในกรณีการจำแนกประเภทสินค้าค้างส่ง (backorders) พบว่าอัลกอริทึมที่มีผลลัพธ์ที่มีประสิทธิภาพดีที่สุดคือ อัลกอริทึม Random Forest โดยตัวชี้วัดที่เหมาะสมที่ใช้ในการวัดผลของอัลกอริทึมที่ดีที่สุดคือค่า Precision Recall และ F1 score โดยพิจารณาจากค่าของคลาสของสินค้าที่เป็น backorders เป็นหลัก (ดังตาราง11) และนอกจากนี้ยังพิจารณาจากกราฟ Precision Recall และค่า Average Precision ด้วย

1.3 ผู้วิจัยได้ทำการศึกษาการใช้เทคนิคการแบ่งกลุ่มข้อมูลฝึกฝนและข้อมูลทดสอบที่เหมาะสมสำหรับข้อมูลที่มีความไม่สมดุลกัน (Imbalanced Data) โดยพบว่าวิธีการที่เหมาะสมที่สุดที่ใช้สำหรับการสุ่มข้อมูลในกรณีของ Imbalanced นี้คือ K-Fold Cross Validation แบบ Stratified Shuffle Split

2. ข้อเสนอแนะ

จากงานวิจัยนี้ในอนาคตจะทำการศึกษาวิธีการสุ่มตัวอย่างด้วยวิธีอื่นๆ สำหรับใช้แบ่งประเภทของข้อมูลเพื่อให้ได้ข้อมูลที่มีความแม่นยำมากกว่านี้และพยายามหาอัลกอริทึมอื่นๆ เพื่อนำมาทดสอบประสิทธิภาพการจำแนกร่วมด้วย เช่น Support vector machine (SVM) และ Logistic Regression เป็นต้น

บรรณานุกรม

บรรณานุกรม

Chompoonoot, Kasemset; et al. (2014). *The Application of Inventory Management Theory for Bakery* Factor. *Journal of Industrial Technology UBRU*. 4(1): 12-26.

Han, J.; Pei, J.; & Kamber, M. (2011). *Data mining: concepts and techniques*. *Elsevier*.

Kittipong, Chomboon; &,Kittisak, Kerdprasop; & Kerdprasop, Nittaya. (2013). *Rare class discovery techniques for highly imbalance data. In Proc. International multi conference of engineers and computer scientists (Vol. 1).*

Kueipo, Huang. (2017). *Backorders Ml: Imbalamced Sampling | Auc ~ 0.947*. Retrieved February 25, from https://www.kaggle.com/kueipo/backorders-ml-imbalamced-sampling-auc-0-947

Noppamas, Pukkhem. (2016). *Bayes' Theorem* Retrieved March 15, 2017, from http://mis.csit.sci.tsu.ac.th/noppamas/download/DataMining/DataMiningCh7V1.pdf

Sripaoraya, Surawat; Sinsomboonthong, Saichon. (2017). *Efficiency Comparison of Data MiningClassification Methods for Chronic Kidney Disease: A Case Study of a Hospital in India*. *Journal of Science and Technology*. 25(5): 839-853.

Ethem, Alpaydin. (2014). *Introduction to Machine Learning*. Retrieved March 15, 2017, from http://slideplayer.com/slide/5124893

NoorAbdulHamid, Mohd. (2015). *Classification Using Decision Tree*. Retrieved March 15, 2017, from https://www.slideshare.net/knottisme/classification-using-decision-tree-53984611

Yuqi, Li. (2017). *Backorder Prediction Using Machine Learning for Danish Craft Beer Breweries*. *Ph.D. Thesis.* AALBORG University of Aalborg at Denmark.

Morgun, Ivan. (2017). *Classification Using K-Nearest Neighbor in R*. Retrieved March 15, 2017 from http://en.proft.me/2017/01/22/classification-using-k-nearest-neighbors-r

Leo, Breiman. (2001). *Random Forests*. *Machine learning*. 45(1): 5-32.

Dong, Guo. (2012). *Additive Model and Boosting Tree*. 1 November Ed. Retrieved March 15, 2017 from https://www.slideshare.net/guo_dong/additive-model-and-boosting-tree

Rory, Mitchell; & Eibe, Frank. (2017). *Accelerating the Xgboost Algorithm Using Gpu Computing*. *PeerJ Computer Science*. 3: e127.

Pasapitch, Chujai. et al. (2015). *Ensemble Learning for Imbalanced Data Classification Problem*.

Dred, Law. (2017). *Predicting Backorders with 3 Models*. Retrieved February 25, 2017

from https://www.kaggle.com/dredlaw/predicting-backorders-with-3-models

Rodrigo, Santis; Aguiar, Eduardo; & Goliatt, Leonardo. (2017). *Predicting material backorders in inventory management using machine learning.* Computational Intelligence (LA-CCI). *IEEE Latin American Conference.* p. 1-6.

Wu Ting-Fan; Lin Chih-Jen; & Weng Ruby C. (2004). *Probability Estimates for Multi-Class Classification by Pairwise Coupling*. *Journal of Machine Learning Research*. 5(8): 975-1005.

Davis, Jesse; & Mark, Goadrich. (2006). *The Relationship between Precision-Recall and Roc Curves*. *Proceedings of the 23rd international conference on Machine learning*. ACM. p. 233-240

ภาคผนวก

การนำเข้าไลบารีที่สำคัญที่ใช้ในการประมวลผลเพื่อใช้ในการศึกษา

ไลบารีที่นำเข้ามาใช้ในการศึกษา	จุดประสงค์ในการใช้
from pandas import Series	ใช้สำหรับการเขียนภาษาโปรแกรม python
import numpy as np	ใช้สำหรับการเขียนฟังก์ชัน เกี่ยวกับคณิตศาสตร์และการคำนวณ
import pandas as pd	ใช้ในการนำเข้าไฟล์ CSV
import matplotlib .pyplot as plt	ใช้ในการสร้างกราฟในรูปแบบ scatter plot
import seaborn as sns	ใช้ในการสร้างกราฟแสดงความสัมพันธ์ของข้อมูล
from sklearn .neighbors import KNeighborsClassifier	ใช้ในการสร้างแบบจำลองของอัลกอริทึม K-Nearest neighbors
from sklearn .ensemble import RandomForestClassifier	ใช้ในการสร้างแบบจำลองของอัลกอริทึม Random forest
from sklearn .tree import DecisionTreeClassifier	ใช้ในการสร้างแบบจำลองของอัลกอริทึม Decision tree
from sklearn .ensemble import AdaBoostClassifier	ใช้ในการสร้างแบบจำลองของอัลกอริทึม Adaboost
from sklearn .naive_bayes import GaussianNB	ใช้ในการสร้างแบบจำลองของอัลกอริทึม Naive Bays
import xgboost as xgb	ใช้ในการสร้างแบบจำลองของอัลกอริทึม XGboost
from sklearn .model_selection import StratifiedShuffleSplit	ใช้ในการสุ่มข้อมูลออเป็น K กลุ่ม
from sklearn .metrics import accuracy_score	ใช้ในการวัดผลประสิทธิภาพของแบบจำลองจากค่าความถูกต้อง
import time	ใช้ในการวัดผลประสิทธิภาพด้านเวลาของแบบจำลอง
from sklearn .metrics import classification_report	ใช้ในการวัดผลประสิทธิภาพของแบบจำลองด้วยค่า Precision Recall และ F1 Score
from sklearn .metrics import confusion_matrix	ใช้ในการวัดผลประสิทธิภาพของแบบจำลองด้วย Confusion matrix

การนำเข้าไลบารีที่สำคัญที่ใช้ในการประมวลผลเพื่อใช้ในการศึกษา

ไลบารีที่นำเข้ามาใช้ในการศึกษา	จุดประสงค์ในการใช้
from sklearn import metrics	ใช้ในการนำเข้าตัวชี้วัด
from sklearn .metrics import average_precision_score	ใช้ในการวัดผลประสิทธิภาพของแบบจำลองด้วย ค่าเฉลี่ยระหว่าง Precision และ Recall
from sklearn .metrics import precision_recall_curve	ใช้ในการวัดผลประสิทธิภาพของแบบจำลองด้วย Precision-recall curve

ประวัติย่อผู้ทำสารนิพนธ์

ประวัติย่อผู้ทำสารนิพนธ์

ชื่อ ชื่อสกุล	นางสาวเจนจิรา มหาสุข
วันเดือนปีเกิด	25 มกราคม พ.ศ. 2533
สถานที่เกิด	โรงพยาบาล ราชวิถี จังหวัดกรุงเทพมหานคร
สถานที่อยู่ปัจจุบัน	บ้านเลขที่ 6004/130 ซอยประชาสงเคราห์ 5 ถนนประชาสงเคราห์ ดินแดง กรุงเทพมหานคร 10400
ตำแหน่งหน้าที่การงานปัจจุบัน	หัวหน้าฝ่ายเคมี 6
สถานที่ทำงานปัจจุบัน	บริษัท โตโยต้า ทูโช (ไทยแลนด์) จำกัด

ประวัติการศึกษา

พ.ศ. 2554	วิทยาศาสตรบัณฑิต สาขาวิชาบริหารธุรกิจเกษตร จาก สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2560	วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ จาก มหาวิทยาลัยศรีนครินทรวิโรฒ