การพัฒนารูปแบบการทำนายความหยาบผิวก่อนเสร็จกระบวนการพิมพ์ในเครื่องพิมพ์สามมิติโดยใช้ข้อมูลอนุกรมเวลา

นายอภิวัฒน์ อินทร์ชู

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมเครื่องกลและระบบกระบวนการ
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2567

DEVELOPMENT OF A SURFACE ROUGHNESS PREDICTION MODEL DURING PRINTING PROCESS OF A 3D PRINTER USING TIME-SERIES DATA

APIWAT INCHOO

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Mechanical and Process System Engineering
Suranaree University of Technology
Academic year 2024

การพัฒนารูปแบบการทำนายความหยาบผิวก่อนเสร็จกระบวนการพิมพ์ในเครื่องพิมพ์สามมิติโดยใช้ข้อมูลอนุกรมเวลา

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

.....

(ศาสตราจารย์ ดร.สุจินต์ บุรีรัตน์)
ประธานกรรมการ

.....

(อาจารย์ ดร.จิตติมา วระกุล)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

.....

(รองศาสตราจารย์ ดร.ประเสริฐ เอ่งฉ้วน)
กรรมการ

.....

(รองศาสตราจารย์ ดร.ยุพาพร รักสกุลพิวัฒน์)
รองอธิการบดีฝ่ายวิชาการและประกันคุณภาพ

.....

(รองศาสตราจารย์ ดร.พรศิริ จงกล)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

อภิวัฒน์ อินทร์ชู : การพัฒนารูปแบบการทำนายความหยาบผิวก่อนเสร็จกระบวนการพิมพ์ในเครื่องพิมพ์สามมิติโดยใช้ข้อมูลอนุกรมเวลา (DEVELOPMENT OF A SURFACE ROUGHNESS PREDICTION MODEL DURING PRINTING PROCESS OF A 3D PRINTER USING TIME-SERIES DATA) อาจารย์ที่ปรึกษา : อาจารย์ ดร.จิตติมา วระกุล, 122 หน้า.

คำสำคัญ: Additive manufacturing/fused deposition modeling/Predictive modeling/ Surface roughness/Time-series

เทคโนโลยีการพิมพ์ 3 มิติแบบสะสมวัสดุหลอมรวม (Fused Deposition Modeling: FDM) เป็นเทคนิคที่ได้รับความนิยมเนื่องจากมีต้นทุนต่ำและเหมาะสำหรับการสร้างต้นแบบ อย่างไรก็ตาม ข้อจำกัดด้านคุณภาพพื้นผิวของชิ้นงาน เช่น ความคลาดเคลื่อนของหัวฉีด แรงสั่นสะเทือนของเครื่องพิมพ์ และการเปลี่ยนแปลงของอุณหภูมิ ยังคงเป็นปัจจัยสำคัญที่ต้องควบคุม งานวิจัยนี้นำเสนอแนวทางการทำนายความหยาบของพื้นผิวชิ้นงานโดยใช้ข้อมูลจากเซ็นเซอร์สามประเภท ได้แก่ เทอร์โมคัปเปิล (Thermocouple) สำหรับวัดอุณหภูมิ, เซ็นเซอร์เร่งความเร็ว (Accelerometer) สำหรับตรวจจับแรงสั่นสะเทือน และเซ็นเซอร์การปล่อยเสียงอะคูสติก (Acoustic Emission Sensor) สำหรับวิเคราะห์เสียงที่เกิดขึ้นระหว่างกระบวนการพิมพ์ โดยใช้ 6 อัลกอริทึม ได้แก่ AR, ARIMA, XGBoost, SVR, LSTM และ RNN ในการพัฒนาระบบทำนายผล

ผลการทดลองพบว่า อัลกอริทึม ARIMA ที่ใช้ข้อมูลจากเซ็นเซอร์วัดอุณหภูมิสามารถให้ค่าความผิดพลาดต่ำสุดที่ RMSE 0.16 ไมโครเมตร และ MAPE 8.49% โดยใช้เวลาในการประมวลผลเพียง 2.30 วินาที ขณะที่อัลกอริทึม XGBoost แสดงประสิทธิภาพสูงสุดสำหรับข้อมูลจากเซ็นเซอร์เร่งความเร็วและเสียงอะคูสติก โดยให้ค่า RMSE เท่ากับ 6.35 และ 6.58 ไมโครเมตร ตามลำดับ และใช้เวลาในการประมวลผลต่ำสุดเพียง 0.07-0.12 วินาที

งานวิจัยนี้แสดงให้เห็นถึงศักยภาพของการใช้โมเดลพยากรณ์เพื่อช่วยลดของเสียในการผลิตและปรับปรุงคุณภาพพื้นผิวของชิ้นงานพิมพ์ FDM ได้แบบเรียลไทม์ ซึ่งเป็นแนวทางที่มีประโยชน์ต่อการพัฒนาเทคโนโลยีการพิมพ์ 3 มิติให้มีประสิทธิภาพสูงขึ้น

สาขาวิชา วิศวกรรมการผลิต ลายมือชื่อนักศึกษา อภิวัฒน์ อินทร์ชู

ปีการศึกษา 2567 ลายมือชื่ออาจารย์ที่ปรึกษา

APIWAT INCHOO : DEVELOPMENT OF A SURFACE ROUGHNESS PREDICTION MODEL DURING PRINTING PROCESS OF A 3D PRINTER USING TIME-SERIES DATA. THESIS ADVISOR : JITTIMA VARAGUL, Ph.D. 122 PP.

Keyword: Additive manufacturing/fused deposition modeling/Predictive modeling/ Surface roughness/Time-series

Fused Deposition Modeling (FDM) is a widely used 3D printing technology known for its cost-effectiveness and suitability for prototyping. However, limitations such as nozzle misalignment, mechanical vibrations, and temperature fluctuations can significantly affect the surface quality of printed parts. This study proposes a predictive modeling approach to estimate surface roughness during the printing process using data from three types of sensors: thermocouple (measuring temperature), accelerometer (detecting vibrations), and acoustic emission sensor (analyzing sound emissions). Six machine learning algorithms—AR, ARIMA, XGBoost, SVR, LSTM, and RNN—were employed to develop predictive models.

Experimental results indicate that the ARIMA model is the most suitable for temperature data, achieving the lowest RMSE of 0.16 μm and a MAPE of 8.49%, with a processing time of only 2.30 seconds. For accelerometer and acoustic emission data, the XGBoost model demonstrated the highest accuracy, yielding RMSE values of 6.35 μm and 6.58 μm, respectively, with minimal processing times of 0.07–0.12 seconds.

This research highlights the potential of predictive modeling in enhancing the FDM process by enabling real-time assessment of surface roughness. The proposed approach can reduce material waste and improve the efficiency and quality of 3D-printed parts, contributing to the advancement of additive manufacturing technology.

School of Manufacturing Engineering
Academic Year 2024

Student's Signature Apiwat Inchoo
Advisor's Signature .....

กิตติกรรมประกาศ

ข้าพเจ้าขอกราบขอบพระคุณ อาจารย์ ดร.จิตติมา วระกุล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่เมตตาและทุ่มเทอย่างยิ่งในการให้คำแนะนำและแนวทางการทำวิจัยตลอดระยะเวลาที่ผ่านมา ความใส่ใจในรายละเอียดและความเอาใจใส่ในทุกขั้นตอนของท่าน ช่วยให้ข้าพเจ้าได้พัฒนาตนเองทั้งในด้านวิชาการและการวิจัยอย่างลึกซึ้ง ความกรุณาของท่านเป็นแรงบันดาลใจสำคัญที่นำพาข้าพเจ้าไปสู่ความสำเร็จในครั้งนี้

ข้าพเจ้าขอขอบพระคุณคณะกรรมการคุมสอบป้องกันวิทยานิพนธ์ที่กรุณาให้คำแนะนำอันทรงคุณค่าและข้อเสนอแนะที่ช่วยพัฒนาวิทยานิพนธ์ของข้าพเจ้าให้มีความสมบูรณ์และมีคุณภาพยิ่งขึ้น คำแนะนำของท่านทำให้ข้าพเจ้ามั่นใจในผลงานของตนเองและได้รับประสบการณ์ที่ทรงคุณค่าในกระบวนการวิจัย

ขอแสดงความขอบพระคุณอย่างยิ่งต่อ ผู้ช่วยศาสตราจารย์ ดร.วรรณวนัช บุ่งสุด หัวหน้าสาขาวิชา ที่มีบทบาทสำคัญในการสนับสนุนด้านวิชาการและการบริหารจัดการ ทำให้การดำเนินการวิจัยของข้าพเจ้าเป็นไปอย่างราบรื่น ท่านเป็นตัวอย่างของความมุ่งมั่นและความเป็นมืออาชีพในสายงานวิชาการที่ข้าพเจ้ายึดถือเป็นแรงบันดาลใจ

ข้าพเจ้าขอขอบพระคุณเจ้าหน้าที่ประจำสำนักวิศวกรรมศาสตร์ที่ช่วยจัดการเอกสาร อุปกรณ์และอำนวยความสะดวกในการดำเนินการวิจัย

ขอขอบคุณจากใจจริงต่อครอบครัวซึ่งเป็นแรงสนับสนุนสำคัญในทุกช่วงเวลาของชีวิต ความรัก ความห่วงใย และกำลังใจจากพ่อแม่ ญาติพี่น้อง และทุกคนในครอบครัว เป็นสิ่งที่ข้าพเจ้าตระหนักดีว่าเป็นปัจจัยสำคัญที่นำไปสู่ความสำเร็จในครั้งนี้

สุดท้ายนี้ ข้าพเจ้าขอขอบพระคุณ Dr. Htun Htun Win ผู้ซึ่งเป็นกำลังใจที่สำคัญยิ่งในช่วงเวลาที่ยากลำบาก ท่านเป็นในแรงผลักดันที่ทำให้ข้าพเจ้ามีความมุ่งมั่นและเข้มแข็งในการเผชิญกับทุกอุปสรรค

อภิวัฒน์ อินทร์ชู

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เทคโนโลยีการเติมแต่งสาร (Additive Manufacturing :AM) คือกระบวนการผลิตชั้นสูงที่รวมเนื้อของวัสดุเข้ารวมกันในลักษณะชั้นต่อชั้นเพื่อสร้างชิ้นงานสามมิติซึ่งสามารถสร้างได้โดยตรงจากการออกแบบโดยใช้คอมพิวเตอร์ช่วย(Computer Aided Design :CAD) (Bourell, Leu, & Rosen, 2009) เทคโนโลยีการเติมแต่งสารได้รับความนิยมในงานที่ต้องการความแม่นยำและมีความซับซ้อนสูง ในปัจจุบันเทคโนโลยีการเติมแต่งสารสามารถจำแนกได้เป็นเจ็ดประเภท material extrusion, powder bed fusion, vat photo polymerization, material jetting, binder jetting, sheet lamination และ directed energy deposition (Gibson, Rosen, & Stucker, 2010) FDM เป็นการสร้างแบบจำลองแบบหลอมรวมเส้นใยฟิลาเมนต์ต่อเนื่องและอัดขึ้นรูปลงบนแผ่นเพลท (Asadi-Eydivand et al., 2016) FDM มีการใช้กันอย่างแพร่หลายเนื่องจากความสามารถในการขึ้นรูปงานที่ซับซ้อนและมีต้นทุนที่ต่ำ

แม้ว่ากระบวนการสร้างแบบจำลองแบบหลอมรวมเป็นกระบวนการสร้างต้นแบบที่รวดเร็วหนึ่งในข้อจำกัดกระบวนการสร้างแบบจำลองแบบหลอมรวมคือการประกันคุณภาพของชิ้นงาน หนึ่งในความแปรปรวนที่พบได้บ่อยที่สุดในกระบวนการสร้างแบบจำลองแบบหลอมรวมเนื่องจากวัสดุหดตัวก่อให้เกิดการยกตัวของชิ้นงานและหลุดออกจากฐานในที่สุด (Zhang et al., 2002)

นอกจากนี้การผลิตด้วยกระบวนการสร้างแบบจำลองแบบหลอมรวมมักมีผิวสำเร็จไม่ดีเมื่อเทียบกับชิ้นงานที่มีการลบผิวออก ความหยาบของพื้นผิวส่งผลต่อพฤติกรรมไตรโบโลยี ของพื้นผิวเพราะผิวขรุขระสึกหรอมากกว่า จึงเป็นสิ่งสำคัญมากในการทำนายและควบคุมความหยาบของผิวชิ้นงานที่ผลิตด้วยกระบวนการสร้างแบบจำลองแบบหลอมรวม ปัจจัยมากมายที่มีผลกับความหยาบผิว เช่น อุณหภูมิ ความสูงของชั้น ความเร็วในการอัดรีด ระยะห่างระหว่างหัวฉีดและเพลท ซึ่งปัจจัยข้างต้นมีแนวโน้มที่เซ็นเซอร์จะสามารถเก็บข้อมูลในรูปแบบของอนุกรมความเวลา ซึ่งสอดคล้องกับผลการศึกษาที่ผ่านมา (Byun & Lee, 2006; Turner & Gold, 2015; Strano et al., 2013; Lia et al., 2019)

วัตถุประสงค์ของงานวิจัย พัฒนารูปแบบการทำนายออนไลน์ในการทำนายคุณภาพของผิวในกระบวนการสร้างแบบจำลองแบบหลอมรวมโดยอาศัยชุดข้อมูลอนุกรมเวลาสร้างอัลกอริทึมเพื่อใช้ในการทำนายคุณภาพ และ สามารถทำงานได้เมื่อติดตั้งบนกระบวนการสร้างแบบจำลองแบบหลอมรวม (FDM) ต่างรุ่น และเปรียบเทียบความแม่นยำของอัลกอริทึมต่างชนิด

จากวัตถุประสงค์ข้างต้นบทความนี้จะมีการจำลองพารามิเตอร์สำหรับการสร้างโมเดลทำนายความหยาบผิวบนกระบวนการ FDM โดยทำการเก็บข้อมูลลักษณะอนุกรมเวลาจาก ชิ้นงานจำนวน 100 ชิ้นงานที่แตกต่างกันรวบรวมข้อมูลด้วยเทอร์โมคัปเปิล, เซ็นเซอร์วัดความเร่งและอะคูส-ติกอิมิชชั่นเซนเซอร์ (เซ็นเซอร์ปล่อยเสียง) หลังจากรวบรวมข้อมูลจากที่กล่าวมาจะนำชุดข้อมูลร้อยละ 20, 40, 60, 80 และ 100 ของเวลาทั้งหมดแปลงอนุกรมเวลาของอุณหภูมิเป็นอนุกรมเวลาที่ไม่คงที่เพื่อให้ชุดข้อมูลไม่ขึ้นอยู่กับชิ้นงาน และแปลงอนุกรมเวลาของการสั่นสะเทือนและระดับเสียงเป็น อนุกรมความถี่ของการสั่นสะเทือน และ อนุกรมความถี่ของระดับเสียงเพื่อไม่ให้ชุดข้อมูลขึ้นอยู่กับเวลา ทำการสร้างโมเดลโดยอัลกอริทึมทั้งหมด6อัลกอริทึมเพื่อใช้ในการเปรียบเทียบประสิทธิภาพด้านความแม่นยำ และเวลาในการประมวลผลของอัลกอริทึมได้แก่ Autoregressive (AR), Autoregressive Integrated Moving Average (ARIMA), XGBoost, Support Vector Regression (SVR), Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN)

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 พัฒนาอัลกอริทึมสำหรับการตรวจสอบออนไลน์และทำนายความหยาบผิวของชิ้นงาน

1.2.2 เพื่อเปรียบเทียบประสิทธิภาพผลการพยากรณ์ลักษณะของความหยาบผิวชิ้นงานพิมพ์ก่อนเสร็จกระบวนการพิมพ์ได้

1.3 ขอบเขตของการวิจัย

1.3.1 ทดสอบบนเครื่องพิมพ์สามมิติแบบเปิด

1.3.2 ควบคุมอุณหภูมิแวดล้อมโดยเครื่องปรับอากาศ

1.4 สถานที่ทำงานวิจัย

อาคารเครื่องมือวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีสุรนารี

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 สามารถพัฒนาอัลกอริทึมเพื่อทำนายความหยาบผิวของชิ้นงานก่อนเสร็จกระบวนการได้

1.5.2 มีส่วนช่วยในการลดต้นทุนการผลิตในกระบวนการพิมพ์ 3 มิติได้

1.6 แผนการดำเนินงาน

กิจกรรม/ขั้นตอนการดำเนินการ	เดือนที่											
	1	2	3	4	5	6	7	8	9	10	11	12
1. ทบทวนการศึกษา รวบรวมข้อมูลรวมทั้งสำรวจปริทัศน์วรรณกรรม และงานวิจัยที่เกี่ยวข้อง	x	x	x									
2. ศึกษาปัจจัยที่มีผลต่อชิ้นงาน 30 Print		x	x	x								
3. ศึกษาอัลกอริทึมที่เกี่ยวข้องกับงานวิจัย			x	x	x							
4. ออกแบบการทดลอง ทำการทดลอง และบันทึกผลการทดลอง						x	x	x				
5. ทำการทดลองขริง และบันทึกผลการทดลอง							x	x	x			
6. วิเคราะห์ผลการทดลอง								x	x			
7. สรุปผลการศึกษาและจัดทำข้อเสนอแนะ									x	x	x	
8. จัดทำวิทยานิพนธ์											x	x
9. สอบวิทยานิพนธ์												x
ปริมาณงานที่วางแผนไว้ (%)	5	5	5	5	10	10	10	10	10	10	10	10
ปริมาณงานที่ทำได้จริง (%)	5	5	5	5	10	10	10	10	10	10	10	10
งานสะสมที่วางแผนไว้ (%)	5	10	15	20	30	40	50	60	70	80	90	100
งานสะสมที่ทำได้จริง (%)	5	10	15	20	30	40	50	60	70	80	90	100

1.7 การจัดทำรูปเล่มวิทยานิพนธ์

บทที่ 1 เป็นบทนำ ซึ่งจะกล่าวถึงที่มาและความสำคัญของปัญหา วัตถุประสงค์ หลักการสมมุติฐาน และกรอบแนวคิด ตลอดจนขอบเขตและประโยชน์ที่คาดว่าจะได้รับจากงานวิจัยนี้

บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้องกับการทำนายผิวชิ้นงานพิมพ์สามมิติที่ถูกสร้างโดยเครื่องพิมชนิด Fused Deposition Modeling เพื่อลดปัญหาของเสียในเวลาผลิตและสร้างโมเดลที่มีความยืดหยุ่นในการทำนายผิวชิ้นงาน

บทที่ 3 การดำเนินงาน โดยเริ่มจากการสุ่มเทียมปัจจัยที่มีผลกับความหยาบผิวของชิ้นงาน วัดความหยาบผิวของชิ้นงานและใช้โปรแกรม python เพื่อช่วยสร้างโมเดลที่ใช้อัลกอริทึมที่แตกต่างกัน 6 อัลกอริทึม

บทที่ 4 แสดงผลการทำงานของโมเดลที่แตกต่างกัน 90 โมเดลและเปรียบเทียบผลของโมเดลโดยงานวิจัยนี้เลือกใช้ Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) และเวลาในการประมวลผลในการเปรียบเทียบประสิทธิภาพ

บทที่ 5 เป็นการสรุปและข้อเสนอแนะ

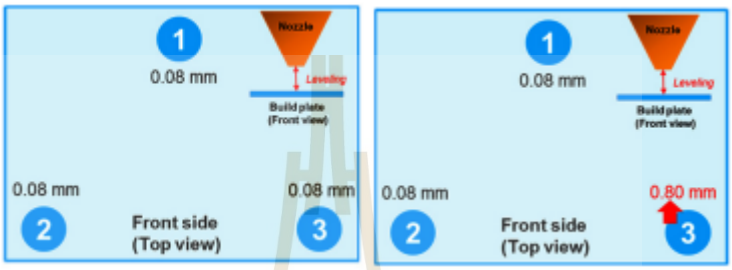
บทที่ 2
ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

2.1 เทคโนโลยีการเติมแต่งสาร(Additive Manufacturing : AM)

เทคโนโลยีการเติมแต่งสาร คือ เทคโนโลยีการผลิตเป็นกระบวนการผลิตชิ้นงานแบบเพิ่มเนื้อโดยอาศัยการออกแบบโดยใช้คอมพิวเตอร์ช่วย(CAD) หรือที่รู้จักกันในชื่อ 3D Print การออกแบบโดยใช้คอมพิวเตอร์ช่วย (CAD) เทคโนโลยีในการนำคอมพิวเตอร์มาช่วยออกแบบหรือช่วยในการสร้างชิ้นส่วนหรือชิ้นงานสามมิติโดยแบบจำลองทางเรขาคณิตแบบจำลองการสะสมแบบหลอมรวม (Fused Deposition Modeling :FDM) เป็นเทคโนโลยีในการสร้างแบบจำลองหลอมเส้นใยฟิลาเมนต์ให้กลายเป็นของเหลวแล้วพ่นฉีดออกมาเป็นเส้นผ่านหัวฉีด (Nozzle) ต่อเนื่องและอัดขึ้นรูปลงบนแผ่นเพลทโดยเส้นใยฟิลาเมนต์จะถูกฉีดออกมาเป็นรูปหลักในแนวแกนระนาบทีละชั้นและซ้อนทับขึ้นไปจากการทบทวนวรรณกรรมบทความเกี่ยวกับการตรวจสอบกระบวนการ AM มีการนำเอาไรโรมิเตอร์, เทอร์โมคัปเปิล, ดิสเพลสเมนต์เซนเซอร์ และกล้องถ่ายภาพอินฟราเรด (IR) มาใช้ในการตรวจสอบกระบวนการAM ตามงานวิจัยของ Tapia และ Elwany (2014), Everton et al. (2016), และ Reutzel & Nassar (2015) มีงานวิจัยจำลองสถานะที่ผิดพลาดโดยการกำหนดระดับที่ผิดพลาดโดยการตั้งระยะห่างระหว่างหัวฉีดและเพลทไม่เท่ากัน(กำหนดให้จุดที่ 1 และ 2 มีระยะห่างเท่ากับ 0.08 มิลลิเมตร และจุดที่ 3 มีระยะห่างเท่ากับ 0.80 มิลลิเมตร) (รูปที่ 2.1) จากการกำหนดระดับที่ผิดพลาดส่งผลให้ชิ้นงานมีการยึดติดที่ไม่ดี (รูปที่ 2.2) และมีความพยายามในการทำนายความบกพร่องโดยการวัดวัดอุณหภูมิ การสั่นสะเทือน การปล่อยเสียง มีการนำเสนอว่าวิธีการตรวจสอบออนไลน์สำหรับความผิดพลาดในกระบวนการFDM โดยใช้เทอร์โมคัปเปิล, เซ็นเซอร์วัดความเร่ง, อะคูสติกอิมิชชั่นเซนเซอร์และเซ็นเซอร์อุณหภูมิอินฟราเรดเก็บข้อมูลแบบอนุกรมเวลาโดยการเก็บจะแปลงสัญญานโดยใช้ RMS ในแต่ละชั้นของการพิมพ์จะได้ชุดข้อมูลทั้งสิ้น 15 ชุดข้อมูล ก่อนจะทำการสร้างอัลกอริทึมเพื่อทำนายสถานะของเครื่องพิมพ์สามมิติโดยผลการทดลองแสดงให้เห็นว่าวิธีการนี้สามารถตรวจจับได้อย่างแม่นยำ

จากข้อสรุปข้างต้นการใช้เทอร์โมคัปเปิล, เซ็นเซอร์วัดความเร่ง และอะคูสติกอิมิชชั่นเซนเซอร์มีความเป็นไปได้ในการทำนายสถานะที่ผิดพลาดของชิ้นในระหว่างการทำของเครื่องพิมพ์สามมิติ แต่ในงานวิจัยนี้มีข้อจำกัดใน การเลือกใช้รูปแบบของชิ้นงานหนึ่งรูปแบบและการจำกัดพารามิเตอร์ส่งผลให้อัลกอริทึมที่ถูกสร้าง Overfitting เมื่อนำไปใช้บนชิ้นงานอื่น, พารามิเตอร์ที่แตกต่างออกไป ความแม่นยำของโมเดลจะถูกลดทอนลง และ เนื่องจากการพิมพ์ชิ้นงานในแต่ละชั้นใช้เวลาไม่เท่ากันการแปลงชุดข้อมูลด้วย RMS 15 ชุดข้อมูลต่อหนึ่งชั้นส่งผลให้เมื่อนำโมเดลนี้ไปทำ

การทดลองกับชิ้นงานที่มีระยะเวลาใน 1 ชั้นมากหรือน้อยกว่าชิ้นงานทดสอบ ความแม่นยำของอัลกอริทึมจะถูกลดลงหรือไม่มีประสิทธิภาพมากพอในการทำนายเช่นกันร่วมถึงการจำลองสถานะที่ผิดพลาดโดยการตั้งระยะห่างระหว่างของหัวฉีดและเพลท(กำหนดให้จุดที่ 1 และ 2 มีระยะห่างเท่ากับ 0.08 มิลลิเมตร และจุด ที่ 32 มีระยะห่างเท่ากับ 0.80 มิลลิเมตร) มีค่าของระยะห่างแตกต่างกันค่อนข้างมากโมเดลอาจจะไม่สามารถทำงานได้ในสถานะที่มีค่าของระยะห่างแตกต่างกันน้อยกว่างานวิจัยข้างต้น (Nam, Jo, Kim, & Lee, 2020)



(ก) (ข)

รูปที่ 2.1 การกำหนดสถานะที่ปกติและผิดปกติของงานวิจัย

รูปที่ 2.1(ก) คือสถานะปกติจะตั้งระยะห่างระหว่างหัวฉีดและเพลทในระยะที่เท่ากันทั้งสามตำแหน่งที่ระยะเท่ากับ 0.08 มิลลิเมตร

รูปที่ 2.1(ข) คือสถานะที่ผิดปกติจะตั้งระยะห่างระหว่างหัวฉีดและเพลทในระยะที่เท่ากันทั้งสองตำแหน่ง(ตำแหน่งที่ 1 และ 2) ที่ระยะเท่ากับ 0.08 มิลลิเมตร และตั้งระยะห่างระหว่างหัวฉีดและเพลทที่ตำแหน่งที่สามที่ระยะเท่ากับ 0.80 มิลลิเมตร

(ก) (ข)

รูปที่ 2.2 ภาพถ่ายของชิ้นงานที่พิมพ์โดยเครื่องพิมพ์ 3 มิติ: (ก) สถานะปกติ และ (ข) สถานะผิดพลาด

2.2 ความหยาบผิว (Roughness)

ความหยาบผิว คือ คุณสมบัติหนึ่งของพื้นผิววัสดุที่ไม่ขึ้นอยู่กับชนิดของวัสดุแต่ขึ้นอยู่กับกระบวนการผลิต ความหยาบของผิวมีความสำคัญต่อการใช้งานเช่น ความแม่นยำ และ ความหยาบ

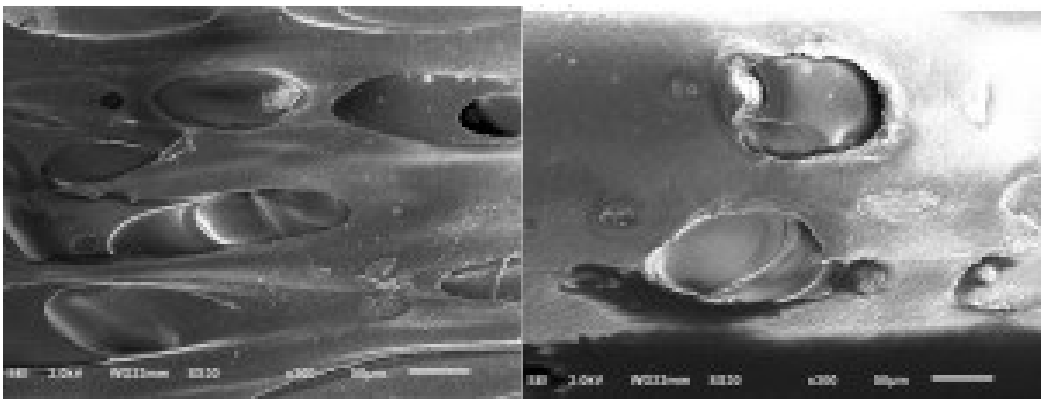
ผิวของพื้นผิวส่งผลสึกหรอ (ยิ่งความหยาบผิวสูงอัตราการสึกหรอจะสูงขึ้น) มีงานวิจัยที่ศึกษาเกี่ยวกับความเร็วในการฉีดของกระบวนการFDM โดยศึกษาโดยการพิมพ์ชิ้นงานการอัดรีดเส้นใย 1 ชั้นพบว่าการความเร็วในการฉีดมีผลกับผิวของชิ้นงานแรงดันหลอมจะส่งผลโดยตรงต่อสัณฐานวิทยาของพื้นผิวและเส้นผ่านศูนย์กลางการอัดรีดของเส้นใยและสูงกว่าความดันหลอมเหลวเป็นประโยชน์ต่อการลดข้อบกพร่องของพื้นผิวของเครื่องอัดรีดกระบวนการอัดรีดซึ่งพบว่ามีความสัมพันธ์กับความไม่เสถียรแรงอัดรีดที่ผันผวนเป็นข้อจำกัดหลักของความเสถียรของกระบวนการอัดรีดสังเกตการบวมของความผันผวนของแม่พิมพ์ละลายในระหว่างการฉีด (รูปที่ 2.3)

จากข้อสรุปข้างต้นการเลือกความเร็วในการอัดรีดจึงเป็นหนึ่งในพารามิเตอร์ที่ส่งผลกระทบต่อพื้นผิวของชิ้นงานแต่ข้อจำกัดของงานวิจัยนี้คือมีการทำการทดสอบในการพิมพ์เพียง 1 ชั้น มีความเป็นไปได้ว่าเมื่อพิมพ์ชิ้นงานที่มีจำนวนชั้นมากขึ้นความเร็วในการอัดรีดจะมีผลแตกต่างกันออกไป (Geng et al., 2019)

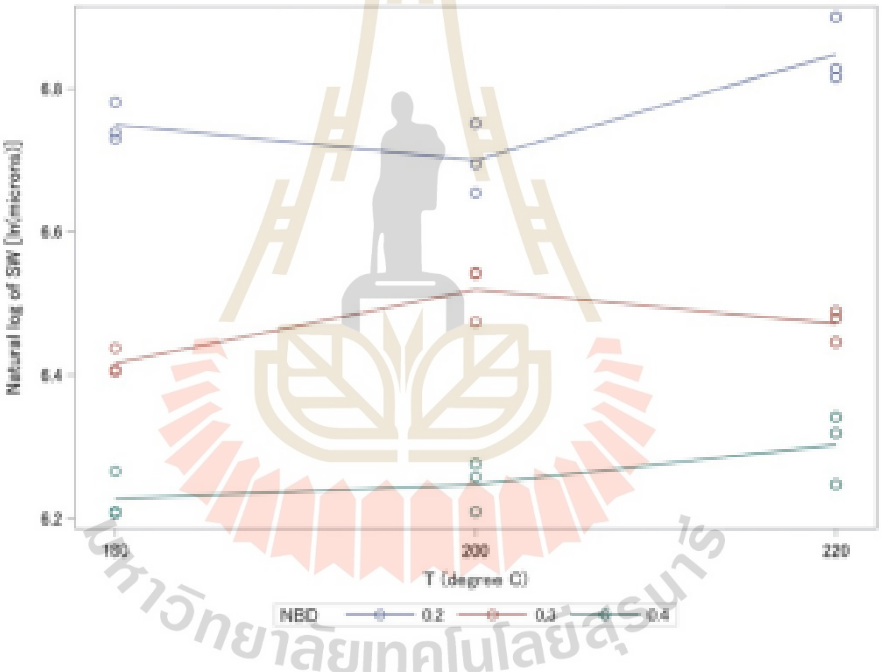
ระยะห่างของชิ้นงานและเพลทส่งผลกระทบต่อผิวของชิ้นงานหลังพิมพ์โดยมีงานวิจัยศึกษาเกี่ยวกับระยะห่างที่เหมาะสมกับกระบวนการพิมพ์สามมิติ โดยการศึกษานี้จะทำการทดลองโดยใช้ ANOVA เพื่อศึกษาความสัมพันธ์ของพารามิเตอร์ที่ส่งผลกับชิ้นงานพบว่าการเพิ่มระยะห่างระหว่าง 0.083 เป็น 0.20 มม. ความหยาบของพื้นผิวจะเพิ่มขึ้นก่อนแล้วจึงลดลง การก่อตัวของช่องว่างในชั้นแรกและด้วยเหตุนี้กลไกการกระจายความร้อนซึ่งต่อไปส่งผลต่อการยึดเกาะของชั้นผิวต่อไป และการที่ระยะห่างของชั้นมากขึ้นส่งผลให้การกลไกในการกระจายความร้อนทำให้คุณสมบัติทางกลของชิ้นงานลดลง จากข้อสรุปข้างต้นอุณหภูมิและระยะห่างของหัวฉีดและเพลทเป็นหนึ่งในพารามิเตอร์ที่ส่งผลกับลักษณะทางกลและผิวของชิ้นงาน สามารถแสดงความสัมพันธ์ระหว่างอุณหภูมิและระยะห่างของหัวฉีดและเพลทได้ดังกราฟ (รูปที่ 2.4)

ข้อจำกัดของงานคือศึกษาบนพารามิเตอร์ที่คงที่ทำให้การเลือกวิธีการทดสอบนี้อาจไม่สอดคล้องกับการพิมพ์บนเครื่องพิมพ์ที่ไม่สามารถคุมพารามิเตอร์ได้เช่น เครื่องพิมพ์สามมิติในลักษณะเปิดที่ไม่สามารถควบคุมอุณหภูมิให้คงที่ได้ (Wang et al., 2019) งานวิจัยนี้จะการศึกษาผลกระทบของการเย็นตัวของเส้นใยโดยจะพิจารณาพารามิเตอร์กระบวนการหลัก เช่น ความเร็วในการอัดขึ้นรูป ขนาดเส้นใยและวัสดุ ลำดับของการสะสมและอุณหภูมิของสิ่งแวดล้อมเพื่อทำนายวิวัฒนาการของอุณหภูมิและการยึดเกาะระหว่างกระบวนการทับถมและจนกว่าความเย็นจะเสร็จสิ้นอุณหภูมิของเส้นใยฟิลาเมนต์หลังจากทำการฉีดส่งผลต่อการยึดเกาะและผิวของชิ้นงาน

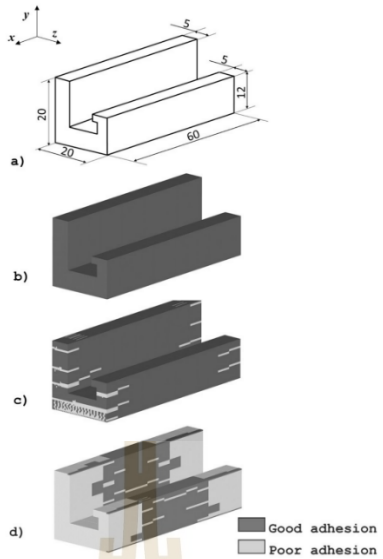
จากการศึกษางานวิจัยพบว่าอุณหภูมิที่หัวฉีดเป็นตัวกำหนดในอุณหภูมิของเส้นใยฟิลาเมนต์และ อุณหภูมิภายนอกจะส่งผลกับการเย็นตัวของชิ้นงาน โดยอุณหภูมิทั้งจะส่งผลกับการยึดเกาะของชิ้นงาน (รูปที่ 2.5) ในงานนี้มีข้อจำกัดโดยทดสอบบนชิ้นงานในลักษณะเดียวอาจไม่สอดคล้องเมื่อทำงานบนชิ้นงานที่แตกต่าง (Costa, Duarte, & Covas, 2017)



รูปที่ 2.3 ลักษณะของผิวชิ้นงานที่พิมโดยความเร็วในการพิมต่างกัน



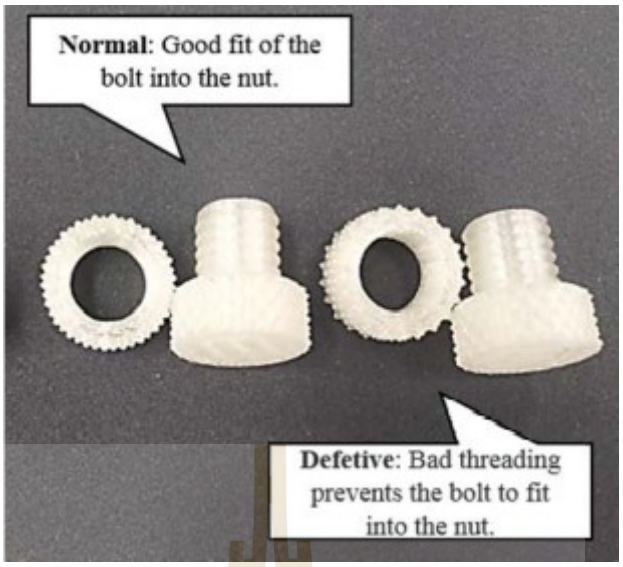
รูปที่ 2.4 กราฟแสดงความสัมพันธ์ระหว่างอุณหภูมิละระยะห่างของหัวฉีดและเพลท



รูปที่ 2.5 ลักษณะชิ้นงานที่อุณหภูมิส่งผลกับการยึดเกาะของชิ้นงาน

2.3 สถานะที่ผิดพลาดของสายพาน

สถานะที่ผิดพลาดของสายพานมีผลกระทบต่อความแม่นยำของเครื่องพิมพ์สามมิติ มีงานวิจัยใช้อะคูสติกอิมิชชั่นเซนเซอร์ (AE sensor) ในการทำนายสถานะที่ผิดพลาดของสายพานโดยการจำลองความผิดพลาดด้วยการนำลวดติดตั้งบนสายพาน มีการพิมพ์ชิ้นงานในลักษณะน็อตและโบลต์ในสถานะที่ปกติชิ้นงานจะต้องสามารถสวมเข้าหากันได้อย่างพอดี และในสถานะที่ผิดพลาดชิ้นงานจะไม่สามารถสวมเข้าหากันได้ (รูปที่ 2.6) ผลของงานวิจัยนี้ชี้ให้เห็นถึงประสิทธิภาพของการทำนายความผิดพลาดของสายพาน จากข้อสรุปข้างต้นการทำงานที่ผิดปกติของสายพานเป็นหนึ่งในพารามิเตอร์ที่ส่งผลกับความแม่นยำในการพิมพ์ชิ้นงานสามมิติ โมเดลที่ถูกสร้างโดยการเก็บข้อมูลโดยใช้ AE sensor มีความสามารถในการทำนายสถาณะที่ผิดพลาดของสายพานได้ งานวิจัยทดสอบความผิดพลาดโดยการสวมชิ้นงานเข้าด้วยกัน มีความเป็นไปได้ว่าเมื่อทำงานในสถานะที่ผิดพลาดของสายพานอาจจะส่งผลกับคุณลักษณะอื่นของชิ้นงานด้วย เช่น ความหยาบผิว (Yoon, He, & Van Hecke, 2014)



รูปที่ 2.6 ชิ้นงานที่ถูกพิมพ์ในสถานะที่ปกติและผิดพลาดของสายพาน

ชิ้นงานด้านซ้ายถูกพิมพ์บนสถานะของสายพานที่ปกติ (ไม่ได้มีการติดตั้งลวดบนสายพาน) พบว่าชิ้นงานน็อตและโบลต์ ที่ผลิตขึ้นจากสถานะนี้สามารถสวมชิ้นงานเข้าด้วยกันได้

ชิ้นงานด้านขวาถูกพิมพ์บนสถานะของสายพานที่ผิดพลาด (มีการจำลองสถานะที่ผิดพลาดของสายพานโดยการติดตั้งลวดบนสายพาน) พบว่าการทำงานที่ผิดพลาดส่งผลต่อความแม่นยำของเครื่องทำให้ชิ้นงานน็อตและโบลต์ที่ถูกพิมพ์โดยสภานะนี้มีรูปแบบคาดเคลื่อนและไม่สามารถสวมชิ้นงานเข้าด้วยกันได้

2.4 การทำนายความหยาบผิว

มีการศึกษาพบว่าความหนาของชั้นและความกว้างของแรสเตอร์มีผลกระทบที่ก็ให้เกิดความหยาบผิวของบนชิ้นงานพิมสามมิติใน FDM (Galantucci, Lavecchia, & Percoco, 2009) มีงานวิจัยนำเสนอแนวทางการเรียนรู้แบบกลุ่มเพื่อการทำนายความหยาบผิวในกระบวนการ FDM โดยการกำหนดพารามิเตอร์ อุณหภูมิของหัวฉีด, ความหนาของชั้น และความเร็วในการพิมพ์ เป็นสามระดับ และมีการเก็บข้อมูลโดยมีการติดตั้งเซ็นเซอร์หลายมีการใช้เซ็นเซอร์อุณหภูมิและมาตรความเร่งถูกใช้เพื่อรวบรวมข้อมูลการตรวจสอบสภาพตามเวลาจริง ชุดคุณสมบัติอนุกรมเวลาและความถี่ถูกดึงออกมาจากสัญญาณที่ใช้เซ็นเซอร์เพื่อปรับปรุงประสิทธิภาพการคำนวณรวมทั้งเพื่อหลีกเลี่ยง overfitting การฝึกฝนโดยใช้อัลกอริทึมการเรียนรู้อัตโนมัติ Machine Learning โดยอัลกอริทึมของเครื่องที่แตกต่างกัน 6 แบบ ได้แก่ RF, AdaBoost, เครือข่าย CART, SVR, RR และ RVFL ผลการวิจัยแสดงให้เห็นว่าโมเดลพยากรณ์สามารถทำนายพื้นผิวได้ จากข้อสรุปข้างต้นแนวทางในการปฏิบัติงานนี้มีความเป็นไปได้ที่จะใช้ในการทำนายผิวของชิ้นงาน แต่ข้อจำกัดของงานนี้คือทำการทดลองบนพื้น

ผิวชิ้นงานในรูปแบบเดียว และเนื่องจาการกำหนดพารามิเตอร์และข้อจำกัดมากจนเกินไปส่งผลให้การนำโมเดลนี้ไปใช้ในการทำนายยผลมีประสิทธิภาพลดลง (Lia, Zhang, Shi, & Wu, 2019)

โดยที่กล่าวมาข้างต้นมีงานวิจัยจำนวนมากที่ศึกษาพารามิเตอร์ที่ส่งผลกระทบกับการพิมพ์ชิ้นงานสามมิติในกระบวนการ FDM เช่นระยะห่างของหัวฉีดและเพลท, ความเร็วในการฉีดของหัวฉีด, อุณหภูมิของหัวฉีด, ความหนาของชั้นการพิมพ์ และการทำงานที่ผิดพลาดของสายพาน มีงานวิจัยใช้เทอร์โมคัปเปิล, เซ็นเซอร์วัดความเร่ง, อะคูสติกอิมิชชั่นเซนเซอร์ ในการเก็บข้อมูลแบบอนุกรมเวลาและสร้างอัลกอริทึมที่ใช้ในการทำนายสถานะของเครื่องและการทำนายผลของผิวชิ้นงานก่อนเสร็จกระบวนการพิมพ์ แต่การศึกษาข้างต้นเป็นการศึกษาโดยใช้การเก็บชุดข้อมูลบนข้อจำกัด, พารามิเตอร์ และชิ้นงานเพียงไม่กี่รูปแบบก่อให้เกิดสถานะ Overfitting ในอัลกอริทึมส่งผลให้โมเดลที่ถูกสร้างขึ้นมีประสิทธิภาพในการทำนายสูงบนข้อจำกัดและพารามิเตอร์รูปแบบเดิม เมื่อนำอัลกอริทึมไปใช้ในงานที่สร้างขึ้นบนข้อจำกัดและพารามิเตอร์หรือบนชิ้นงานอื่นจะส่งผลให้ประสิทธิภาพและความแม่นยำของอัลกอริทึมลดลงอย่างมาก

ในงานวิจัยนี้เสนอการสร้างโมเดลในการทำนายความหยาบผิวของชิ้นงานที่มีรูปแบบแตกต่างกันและ มีข้อจำกัดรวมถึงพารามิเตอร์ในกระบวนการผลิตต่างกันโดยใช้ชุดข้อมูลอนุกรมเวลาและอนุกรมความถี่ในการสร้างโมเดลในการทำนายความหยาบของผิวชิ้นงานในกระบวนการ FDM

2.5 อัลกอริทึม (Algorithm)

อัลกอริทึม คือ ชุดคำสั่งหรือเงื่อนไขที่ถูกออกแบบมาเพื่อให้คอมพิวเตอร์ อุปกรณ์อิเล็กทรอนิกส์ หรือหุ่นยนต์เพื่อให้สามารถปฏิบัติงานได้อย่างอัตโนมัติตามลำดับขั้นตอนที่กำหนดไว้ อัลกอริทึมมีลักษณะยืดหยุ่น โดยสามารถปรับเปลี่ยนหรือพัฒนาให้เหมาะสมกับชุดข้อมูลหรือสถานการณ์ที่แตกต่างกันได้ เพื่อให้สามารถทำงานร่วมกับเงื่อนไขต่าง ๆ ได้อย่างมีประสิทธิภาพ นอกจากนี้ การพัฒนาอัลกอริทึมให้มีความแม่นยำและประสิทธิภาพสูงมักต้องอาศัยข้อมูลที่มีคุณภาพหรือมีความเกี่ยวข้องโดยตรงกับปัญหาที่ต้องการแก้ไข ข้อมูลเหล่านี้ถูกนำมาใช้ในการสอน (Training) หรือปรับปรุงอัลกอริทึมให้สามารถเรียนรู้และทำงานได้อย่างเหมาะสมกับเงื่อนไขที่ซับซ้อนมากยิ่งขึ้น กระบวนการนี้มีบทบาทสำคัญต่อความสำเร็จในการสร้างระบบอัตโนมัติที่ตอบสนองความต้องการใช้งานในด้านต่าง ๆ อย่างมีประสิทธิผล

2.6 การวิเคราะห์อนุกรมเวลา (Time series analysis)

การวิเคราะห์อนุกรมเวลา เป็นการพยากรณ์ค่าของตัวแปลตามเพื่อทำนายผลที่อาจจะเกิดขึ้นในอนาคต โดยใช้ข้อมูลอนุกรมเวลาของตัวแปรนั้นมาศึกษาใช้วิธีการแยกส่วนประกอบของอนุกรมออกมาวิเคราะห์แบ่งได้ 4 ส่วนได้แก่

2.6.1 ค่าแนวโน้ม เป็นการเคลื่อนไหวของอนุกรมเวลาว่าจะโค้งเอียงไปทางใด ค่าแนวโน้มส่วนใหญ่จะบอกทิศทางของอนุกรมเวลานั้น ๆ

2.6.2 การเคลื่อนไหวตามฤดูกาล เป็นความเคลื่อนไหวของข้อมูลที่เกิดขึ้นเนื่องจากอิทธิพลของฤดูกาล ซึ่งจะเคลื่อนไหวในรูปแบบซ้ำ ๆ กันในรอบของช่วงเวลา เช่น ฤดูกาลในหนึ่งปี

2.6.3 การเคลื่อนไหวตามวัฏจักร เป็นการเคลื่อนไหวซ้ำๆคล้ายกับฤดูกาลแต่จะมีระยะที่ยาวนานกว่าฤดูกาลและเป็นวัฏจักรที่สามารถทำนายได้ยากกว่า

2.6.4 การเคลื่อนไหวผิดปกติ เป็นความเคลื่อนไหวที่ไม่แน่นอนไม่สามารถพยากรณ์ได้จากข้อมูลในอดีต

2.7 Machine Learning

Machine Learning หรือการเรียนรู้ของเครื่องจักร เป็นการเรียนรู้จากตัวอย่างโดยปราศจากคำสั่ง อาศัยข้อมูลและเครื่องมือทางสถิติเพื่อทำนายผลลัพธ์ หลักการทำงานคือรับข้อมูลขาเข้า (Input data) และข้อมูลขาออก (Output data) มีความสอดคล้องกันอย่างไร ด้วยเกณฑ์บางอย่าง หรืออัลกอริทึมบางชนิด ซึ่งไม่จำเป็นจะต้องสร้างอัลกอริทึมใหม่ทุกครั้งที่มีข้อมูลใหม่ อัลกอริทึมสามารถปรับตัวเข้ากับการปรับข้อมูลหรือการเพิ่มข้อมูลได้ เนื่องจากการเรียนของเครื่องจักรคล้ายกับพฤติกรรมการเรียนรู้ของมนุษย์จึงไม่สามารถทำงานนอกเหนือกฎเกณฑ์ได้

2.7.1 Autoregressive (AR) เป็นแบบจำลองการถดถอยอัตโนมัติที่ใช้ข้อมูลในอดีตของตัวแปรเดียวกันเพื่อทำนายค่าที่จะเกิดขึ้นในอนาคต โดยอาศัยความสัมพันธ์ระหว่างค่าปัจจุบันและค่าก่อนหน้าในอนุกรมเวลา โมเดลนี้เหมาะสำหรับการคาดการณ์ข้อมูลที่มีรูปแบบต่อเนื่องและแสดงให้เห็นถึงความแม่นยำเมื่อมีความสัมพันธ์เชิงเส้นระหว่างค่าต่าง ๆ ในอดีต

2.7.2 Autoregressive integrated moving average (ARIMA) เป็นหนึ่งในโมเดลที่ใช้กันอย่างแพร่หลายในการคาดการณ์อนุกรมเวลา โดยมีจุดเด่นในการผสมผสานองค์ประกอบหลักสามส่วน ได้แก่ Autoregressive (AR) ซึ่งใช้ค่าก่อนหน้าในอดีตของตัวแปรเดียวกันในการพยากรณ์ค่าในอนาคต, Integrated (I) ซึ่งเป็นกระบวนการปรับข้อมูลโดยการหาความแตกต่างของค่าเพื่อทำให้ข้อมูลมีความเสถียรและลดแนวโน้ม (Trend), และ Moving Average (MA) ซึ่งใช้ข้อผิดพลาดของค่าพยากรณ์ในอดีตมาปรับปรุงการคาดการณ์ในปัจจุบัน

2.7.3 Extreme Gradient Boosting (XGBoost) เป็นเทคนิคการเรียนรู้ของเครื่องที่พัฒนาต่อยอดมาจาก Gradient Boosting โดยมีประสิทธิภาพสูงและได้รับความนิยมอย่างแพร่หลายในการแก้ปัญหาด้านการจำแนกประเภท (Classification) และการถดถอย (Regression) XGBoost เป็นแบบจำลองที่ใช้แนวคิดของการเรียนรู้แบบเป็นลำดับ (Sequential learning) โดยนำการตัดสินใจของต้นไม้หลายต้นมารวมกัน (Ensemble learning) เพื่อให้ได้แบบจำลองที่มีความแม่นยำมากขึ้นเนื่องจากจุดเด่นในเรื่องของความแม่นยำและความสามารถในการจัดการข้อมูลที่มีมิติสูง

XGBoost จึงถูกนำไปใช้ในหลากหลายสาขา เช่น การคาดการณ์ทางการเงิน, การวิเคราะห์ข้อมูลทางการแพทย์, การตรวจจับการฉ้อโกง, และการแข่งขันด้าน Data Science อย่าง Kaggle ซึ่งมักใช้โมเดลนี้เพื่อพัฒนาประสิทธิภาพของการคาดการณ์

2.7.4 Support Vector Regression (SVR) เป็นอัลกอริทึมประเภท Supervised Learning ที่ใช้ในการวิเคราะห์ความสัมพันธ์ระหว่างอินพุตเวกเตอร์ (Input Vector) และตัวแปรเอาต์พุต (Output Variables) โดยมีหลักการทำงานคล้ายกับ Support Vector Machine (SVM) แต่ถูกปรับให้เหมาะกับปัญหาการถดถอย (Regression) แทนการจำแนกประเภท (Classification) SVR ทำงานโดยการค้นหา เส้น Hyperplane ที่สามารถทำนายค่าผลลัพธ์ได้อย่างแม่นยำที่สุด ภายใต้เงื่อนไขที่ค่าความคลาดเคลื่อน (Error) อยู่ในช่วงที่ยอมรับได้ ซึ่งถูกควบคุมโดยพารามิเตอร์ Epsilon (ε) SVR สามารถรับมือกับข้อมูลที่มีมิติสูงและสามารถใช้ Kernel Trick เพื่อเปลี่ยนข้อมูลให้อยู่ในมิติที่สูงขึ้น ทำให้สามารถจับความสัมพันธ์เชิงซับซ้อนได้ดี จึงเหมาะสำหรับการพยากรณ์อนุกรมเวลา (Time Series Forecasting)

2.8 Deep Learning

Deep Learning หรือการเรียนรู้เชิงลึกคือการเรียนรู้โดยอัตโนมัติ อาศัยการเรียนแบบพฤติกรรมของมนุษย์โดยการนำเอาโครงค่ายประสาทเทียมมาซ้อนทับกันเป็นชั้น และทำการเรียนรู้รวมถึงจับกลุ่มข้อมูลแบบอัตโนมัติสิ่งที่สำคัญของการเรียนรู้เชิงลึกคือการเตรียมข้อมูลปัจจัยส่งผลต่อสมรรถนะในการเรียนรู้ซึ่งเป็นรูปแบบการเรียนรู้ของเครื่องจักร (Machine Learning) แต่อาศัยความเฉพาะทางมากกว่า

2.8.1 Long Short-Term Memory (LSTM) เป็นโครงข่ายประสาทเทียมชนิดหนึ่งที่สามารถเรียนรู้การพึ่งพาลำดับระหว่างรายการในลำดับ มักใช้ในการแก้ปัญหาการคาดการณ์อนุกรมเวลา โครงสร้างของ LSTM มีลักษณะเฉพาะที่ช่วยให้สามารถเก็บรักษาข้อมูลสำคัญจากลำดับก่อนหน้าไว้ในหน่วยความจำภายใน (Internal Memory) และสามารถลบหรือปรับปรุงข้อมูลเหล่านั้นตามความจำเป็น ผ่านกลไกสำคัญ เช่น Gate Mechanisms (Input Gate, Forget Gate, Output Gate) ซึ่งทำหน้าที่ควบคุมการไหลของข้อมูลเข้าและออกจากหน่วยความจำ

2.8.2 Recurrent Neural Network (RNN) เป็น Network ชนิดหนึ่งที่นำตัวแปรเอาต์พุต (Output variables) จาก State ที่แล้วมาร่วมเป็นอินพุต (Input) ด้วยซึ่งการทำงานของ RNN มีลักษณะเป็น ลูป (Loop) ซึ่งช่วยให้โครงข่ายสามารถจดจำและใช้ข้อมูลที่เคยประมวลผลไปแล้วในขั้นตอนก่อนหน้า โดยผลลัพธ์ (Output) ของสถานะก่อนหน้าจะถูกนำมาใช้เป็นส่วนหนึ่งของข้อมูลนำเข้า (Input) ในสถานะถัดไป กระบวนการนี้ช่วยให้ RNN มีความสามารถในการเรียนรู้ความเชื่อมโยงระหว่างรายการในลำดับ เช่น การจดจำลำดับคำในประโยค การคาดการณ์ค่าที่เกิดขึ้นในอนาคตจากข้อมูลในอดีต หรือการแยกวิเคราะห์ลักษณะเฉพาะของข้อมูลที่เปลี่ยนแปลงไปตามเวลา

อย่างไรก็ตาม RNN ในรูปแบบพื้นฐานมีข้อจำกัดในเรื่องการจดจำความสัมพันธ์ระยะยาว เนื่องจากปัญหาการสูญเสียข้อมูล

จากการศึกษาข้างต้น ความหยาบของผิวชิ้นงานยังเป็นหนึ่งในข้อจำกัดในการประกันคุณภาพของชิ้นงาน ในการพิมพ์สามมิติ FDM ซึ่งพารามิเตอร์ที่ส่งผลกระทบกับความหยาบผิวในกระบวนการพิมพ์สามมิติได้แก่ ระยะห่างของหัวฉีดและเพลท,ความเร็วในการฉีดของหัวฉีด,อุณหภูมิของหัวฉีด ,ความหนาของชั้นการพิมพ์ และการทำงานที่ผิดพลาดของสายพาน การเลือกใช้เทอร์โมคัปเปิล, เซ็นเซอร์วัดความเร่งและอะคูสติกอิมิชชั่นเซนเซอร์ โดยทำการเก็บชุดข้อมูลแบบอนุกรมเวลา ผ่านการแปลงสัญญานเป็นอนุกรมความถี่ก่อนจะสร้าง Machine Learning หรือ Deep Learning เป็นแนวทางที่มีความเป็นไปได้ที่จะสามารถทำนายผลความเรียบบนผิวของชิ้นงานพิมพ์สามมิติก่อนจะเสร็จกระบวนการพิมพ์ได้ แต่มีข้อจำกัดที่สำคัญคือการศึกษาที่ผ่านมามุ่งเน้นการเก็บข้อมูลบนชิ้นงาน, ข้อจำกัด และ พารามิเตอร์เฉพาะเจาะจงส่งผลให้โมเดลที่ถูกสร้างขึ้นมีประสิทธิภาพลดลงหรือไม่สามารถทำนายได้อย่างมีประสิทธิภาพบนชิ้นงาน,ข้อจำกัด และพารามิเตอร์ที่แตกกต่างกันออกไป

ในงานวิจัยนี้ มีเป้าหมายในการสร้างโมเดลที่สามารถทำนายผลได้อย่างแม่นยำ โดยเริ่มจากการเก็บรวบรวมข้อมูลผ่านอุปกรณ์ตรวจวัดหลายชนิด ได้แก่ เทอร์โมคัปเปิล (Thermocouple) สำหรับตรวจจับอุณหภูมิ, เซ็นเซอร์วัดความเร่ง (Accelerometer) สำหรับตรวจวัดแรงสั่นสะเทือน และอะคูสติกอิมิชชั่นเซนเซอร์ (Acoustic Emission Sensor) สำหรับตรวจจับสัญญาณเสียง ซึ่งอุปกรณ์เหล่านี้จะติดตั้งบนชิ้นงานที่ถูกสร้างขึ้นในลักษณะที่แตกต่างกันทั้งหมด 100 ชิ้นงานระหว่างการทดลอง ได้มีการจำลองสถานการณ์ที่ส่งผลกระทบต่อค่าข้อจำกัดและพารามิเตอร์ต่าง ๆ ในกระบวนการพิมพ์ 3 มิติ เช่น ระยะห่างระหว่างหัวฉีดกับเพลท, ความเร็วในการฉีดของหัวฉีด, อุณหภูมิของหัวฉีด, ความหนาของชั้นการพิมพ์ และข้อผิดพลาดที่เกิดจากสายพาน การออกแบบการทดลองในลักษณะนี้ช่วยให้เกิดการกระจายตัวของข้อมูลที่หลากหลายและครอบคลุมสถานการณ์จริง

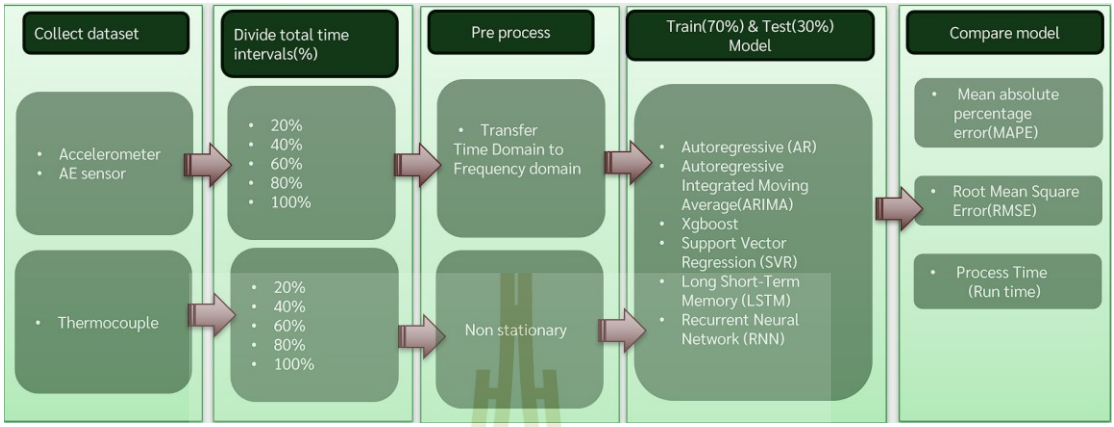
สำหรับการจัดการชุดข้อมูลเบื้องต้น ได้เลือกใช้วิธีแบ่งช่วงข้อมูลเป็นร้อยละ 20, 40, 60, 80 และ 100 ของเวลาทั้งหมดในการพิมพ์ เพื่อแก้ไขปัญหาความไม่สมดุลของเวลาการผลิตในชิ้นงานที่มีลักษณะแตกต่างกัน นอกจากนี้ ยังมีการนำสัญญาณแบบไม่คงที่ของอนุกรมเวลา (Time-Series) และอนุกรมความถี่ (Frequency-Series) มาใช้ในกระบวนการฝึกโมเดล เพื่อให้ข้อมูลที่ใช้มีความหลากหลาย และลดโอกาสเกิดปัญหา Overfitting ที่อาจทำให้โมเดลไม่สามารถทำงานได้ดีเมื่อเจอกับข้อมูลใหม่งานวิจัยนี้จะดำเนินการฝึกโมเดลด้วยอัลกอริทึมทั้งหมด 6 แบบ ได้แก่ AR, ARIMA, XGBoost, SVR, LSTM และ RNN อัลกอริทึมเหล่านี้ได้รับการคัดเลือกเนื่องจากมีความสามารถในการจัดการข้อมูลอนุกรมเวลา และเหมาะสมกับการคาดการณ์ข้อมูลที่ซับซ้อนหลังจากการฝึกโมเดลจะมีการประเมินและเปรียบเทียบประสิทธิภาพของแต่ละอัลกอริทึม โดยวิเคราะห์ทั้งความแม่นยำ (Accuracy) และความไว (Sensitivity) ในการทำนายผล เพื่อหาวิธีที่เหมาะสมที่สุดสำหรับการใช้งานในอนาคต

บทที่ 3
วิธีดำเนินการวิจัย

3.1 ภาพรวมของการดำเนินงานวิจัย

ในงานวิจัยนี้ ได้ดำเนินการเก็บรวบรวมชุดข้อมูลอนุกรมเวลาที่เกี่ยวข้องกับอุณหภูมิ การสั่นสะเทือน และระดับเสียง ซึ่งเกิดขึ้นในกระบวนการทำงานของเครื่องพิมพ์สามมิติ โดยมีการกำหนดข้อจำกัดและพารามิเตอร์ที่แตกต่างกัน (อ้างอิงตารางที่ 3.1) รวมทั้งสิ้น 100 รูปแบบ ชิ้นงานที่ใช้ในการทดลองถูกสร้างขึ้นในรูปแบบลักษณะเรขาคณิตและชิ้นส่วนมาตรฐาน รวมทั้งหมด 20 รูปแบบที่มีขนาดแตกต่างกัน การสร้างรูปแบบเหล่านี้อาศัยการสุ่มค่าพารามิเตอร์ผ่าน Random stage ซึ่งเป็นวิธีการควบคุมการสุ่มค่าให้เหมาะสมสำหรับการเรียนรู้ของเครื่อง (Machine Learning) โดยในงานวิจัยนี้ได้กำหนดค่า Random stage = 42 ซึ่งข้อมูลพารามิเตอร์ที่เกี่ยวข้องสรุปไว้ในตารางที่ 3.2 หลังจากสร้างชิ้นงานและรวบรวมข้อมูลอนุกรมเวลา กระบวนการผลิตถูกแบ่งออกเป็น 5 ช่วงเวลา ได้แก่ ร้อยละ 20, 40, 60, 80 และ 100 ของเวลาผลิตทั้งหมด การเก็บข้อมูลอนุกรมเวลาของความเร่ง (Acceleration), เสียง (Acoustic Emission) และอุณหภูมิ (Temperature) จากชิ้นงานทั้งหมด 100 ชิ้น ส่งผลให้ได้ชุดข้อมูลอนุกรมเวลาเริ่มต้นทั้งหมด 300 ชุดเพื่อเพิ่มความหลากหลายของข้อมูล ชุดข้อมูลทั้งหมดถูกแบ่งย่อยตามช่วงเวลาการผลิตดังกล่าว ซึ่งแต่ละช่วงเวลาจะสร้างชุดข้อมูลเพิ่มขึ้น รวมเป็น 1500 ชุดข้อมูลย่อย โดยแบ่งเป็นชุดข้อมูลอนุกรมเวลาของอุณหภูมิ 500 ชุด การสั่นสะเทือน 500 ชุด และระดับเสียงอีก 500 ชุดในขั้นตอนถัดมา มีการแปลงข้อมูลอนุกรมเวลาบางส่วนเพื่อเพิ่มความเหมาะสมสำหรับการวิเคราะห์และการสร้างโมเดล โดยใช้การแปลงฟูเรียร์ (Fourier Transform) ในการแปลงอนุกรมเวลาของความเร่งและเสียงให้เป็นอนุกรมความถี่ (Frequency-Series) ขณะเดียวกัน ได้ทำการทดสอบ Unit Root กับอนุกรมเวลาอุณหภูมิเพื่อตรวจสอบว่าชุดข้อมูลเป็น Stationary (ชุดข้อมูลที่มีความเสถียรตามเวลา) หรือ Nonstationary (ชุดข้อมูลที่ไม่มีความเสถียร) หากพบว่าเป็นชุดข้อมูลที่ Stationary จะมีการแปลงให้เป็น Nonstationary เพื่อขจัดผลกระทบของความแตกต่างระหว่างชิ้นงานและทำให้ข้อมูลมีลักษณะอิสระมากขึ้นหลังจากการปรับแต่งข้อมูลเบื้องต้น (Preprocessing) ชุดข้อมูลที่เตรียมไว้นี้ถูกนำไปใช้ในการฝึกโมเดล (Model Training) ด้วยอัลกอริทึมทั้งหมด 6 แบบ AR, ARIMA, XGBoost, SVR, LSTM และ RNN โดยการฝึกโมเดลในแต่ละอัลกอริทึมใช้ชุดข้อมูลเดียวกันซึ่งครอบคลุมอนุกรมเวลาของอุณหภูมิ อนุกรมความถี่ของความเร่ง และอนุกรมความถี่ของระดับเสียงในแต่ละช่วงเวลาการผลิต (ร้อยละ 20, 40, 60, 80 และ 100)ดังนั้น ในแต่ละอัลกอริทึมจะสร้างโมเดลทั้งหมด 15 โมเดล และเมื่อรวมทุกอัลกอริทึม งานวิจัยนี้จะสร้างโมเดลทั้งหมด 90 โมเดล ซึ่งจะนำมาทำการเปรียบเทียบ

ประสิทธิภาพในเชิงความแม่นยำ ความไว และความเหมาะสมในสถานการณ์ต่าง ๆ (สรุปผลในตารางที่ 3.3) พร้อมทั้งแสดงแผนผังภาพรวมของกระบวนการวิจัยทั้งหมด (รูปที่ 3.1)



รูปที่ 3.1 แผนภาพกระบวนการปฏิบัติงาน

แผนภาพกระบวนการดำเนินงานในงานวิจัยนี้เริ่มต้นจากการเก็บรวบรวมข้อมูลที่สำคัญ โดยใช้อุปกรณ์ตรวจวัด ได้แก่ เซ็นเซอร์และเทอร์โมคัปเปิล เพื่อตรวจจับสัญญาณต่าง ๆ ที่เกี่ยวข้องในกระบวนการทำงานของเครื่องพิมพ์สามมิติ ข้อมูลที่ได้จะอยู่ในรูปแบบของอนุกรมเวลา (Time-Series) ซึ่งครอบคลุมค่าความเร่ง (Acceleration), ระดับเสียง (Acoustic Emission) และอุณหภูมิ (Temperature)จากนั้น ชุดข้อมูลอนุกรมเวลาถูกจัดแบ่งออกเป็น 5 ช่วงเวลา ได้แก่ ร้อยละ 20, 40, 60, 80 และ 100 ของระยะเวลาการผลิตทั้งหมด เพื่อให้ข้อมูลมีความหลากหลายและสะท้อนกระบวนการผลิตในช่วงเวลาต่าง ๆ ได้อย่างครบถ้วนเมื่อแบ่งข้อมูลเรียบร้อยแล้ว มีการดำเนินการแปลงข้อมูลในขั้นตอนถัดไป โดยอนุกรมเวลาของค่าความเร่งและระดับเสียงจะถูกแปลงให้อยู่ในรูปแบบอนุกรมความถี่ (Frequency-Series) โดยใช้เทคนิคการแปลงฟูเรียร์ (Fourier Transform) ซึ่งช่วยเน้นคุณสมบัติสำคัญในสัญญาณที่เกี่ยวข้องกับความถี่ ขณะเดียวกัน อนุกรมเวลาอุณหภูมิจะถูกตรวจสอบว่ามีคุณสมบัติเป็นชุดข้อมูลคงที่ (Stationary) หรือไม่ หากพบว่าเป็นข้อมูลคงที่ จะมีการแปลงให้เป็นชุดข้อมูลที่ไม่คงที่ (Nonstationary) เพื่อขจัดผลกระทบจากลักษณะเฉพาะของชิ้นงาน และเพื่อให้ข้อมูลที่ได้เหมาะสมสำหรับการวิเคราะห์และการสร้างโมเดลเมื่อข้อมูลทั้งหมดถูกจัดเตรียมในรูปแบบที่พร้อมใช้งานแล้ว ขั้นตอนต่อไปคือการสร้างและทดสอบโมเดลการคาดการณ์ผลลัพธ์ โดยเลือกใช้อัลกอริทึมจำนวน 6 แบบ ได้แก่ แบบ AR, ARIMA, XGBoost, SVR, LSTM และ RNN ซึ่งถูกออกแบบมาให้รองรับการทำงานกับข้อมูลอนุกรมเวลาในขั้นตอนการทดสอบโมเดล จะมีการวิเคราะห์และเปรียบเทียบประสิทธิภาพของโมเดลแต่ละตัวในด้านความแม่นยำ (Accuracy) และความไว (Sensitivity) ในการทำนายผล เพื่อค้นหาอัลกอริทึมที่เหมาะสมที่สุดสำหรับการใช้งานจริงสรุปได้ว่า

กระบวนการดำเนินงานในงานวิจัยนี้ครอบคลุมตั้งแต่การเก็บรวบรวมและแปลงข้อมูล ไปจนถึงการสร้างและเปรียบเทียบประสิทธิภาพของโมเดล เพื่อให้ได้ผลลัพธ์ที่แม่นยำและสอดคล้องกับเป้าหมายของงานวิจัย

3.2 เครื่องมือและอุปกรณ์

3.2.1 เครื่องพิมพ์ 3 มิติชนิดสะสมแบบหลอมรวม (FDM) รูปแบบเปิด

3.2.2 เซ็นเซอร์วัดความเร่งชนิด 1 แกน

3.2.3 เทอร์โมคัปเปิล

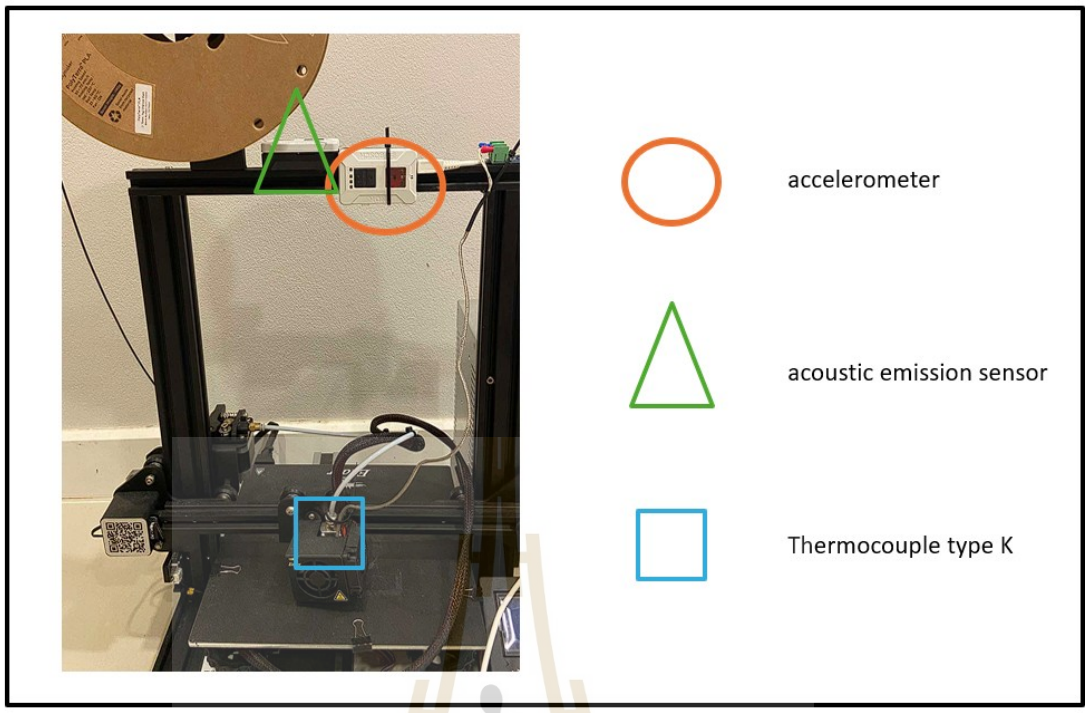
3.2.4 อะคูสติกอิมิชชั่นเซนเซอร์

3.2.5 อุปกรณ์แปลงสัญญาน

3.2.6 คอมพิวเตอร์

3.3 การติดตั้งเซ็นเซอร์ การกำหนดเงื่อนไข และพารามิเตอร์

3.3.1 การติดตั้งเซ็นเซอร์จะทำการติดตั้งเซ็นเซอร์ 3 ชนิด เซ็นเซอร์วัดการสั่นสะเทือนหรือความเร่ง (Accelerometer) เซ็นเซอร์ชนิดนี้ถูกออกแบบมาเพื่อวัดการสั่นสะเทือนหรือการเปลี่ยนแปลงของความเร่งในโครงสร้างของเครื่องพิมพ์สามมิติ โดยมีการติดตั้งเซ็นเซอร์ในตำแหน่งที่เหมาะสมบริเวณโครงของเครื่องพิมพ์สามมิติ การเลือกตำแหน่งนี้ช่วยให้สามารถเก็บข้อมูลการสั่นสะเทือนที่เกิดขึ้นระหว่างการพิมพ์ได้อย่างแม่นยำและสอดคล้องกับการทำงานของเครื่อง เซ็นเซอร์จับสัญญาณเสียง (Acoustic Emission Sensor: AE sensor) สำหรับการตรวจจับเสียงที่เกิดขึ้นระหว่างกระบวนการพิมพ์สามมิติ เซ็นเซอร์ Acoustic Emission ถูกติดตั้งไว้ในตำแหน่งที่อยู่ด้านบนของเครื่องพิมพ์สามมิติ ตำแหน่งนี้ได้รับการคัดเลือกอย่างรอบคอบเพื่อให้สามารถเก็บข้อมูลเสียงที่เกิดขึ้นจากทั้งกระบวนการพิมพ์และการเคลื่อนไหวของชิ้นส่วนต่าง ๆ ได้อย่างมีประสิทธิภาพเทอร์โมคัปเปิล (Thermocouple) เพื่อการวัดอุณหภูมิที่แม่นยำ เทอร์โมคัปเปิลถูกติดตั้งไว้บริเวณหัวฉีด (Nozzle) ซึ่งเป็นจุดสำคัญที่มีการเปลี่ยนแปลงอุณหภูมิอย่างชัดเจนระหว่างการพิมพ์ การติดตั้งในตำแหน่งนี้ช่วยให้สามารถเก็บข้อมูลอุณหภูมิที่สะท้อนถึงสภาวะของกระบวนการพิมพ์ได้โดยตรงดัง (รูปที่ 3.2)



รูปที่ 3.2 การติดตั้งเซ็นเซอร์บนเครื่องพิม3มิติ

การติดตั้งเซ็นเซอร์เพื่อการเก็บข้อมูลจะทำการติดตั้งเซ็นเซอร์ที่ใช้วัดความเร่งตำแหน่งโครงของเครื่องพิมพ์สามมิติ เซ็นเซอร์วัดระดับเสียงด้านบนของเครื่องพิมพ์สามมิติ และติดตั้งเทอร์โมคัปเปิลตำแหน่งหัวฉีด

3.3.2 การกำหนดเงื่อนไข และพารามิเตอร์ จะแบ่งออกเป็น 2 ประเภท 1) การกำหนดเงื่อนไข และพารามิเตอร์ที่คงที่ ได้แก่ กระบวนการจะใช้วัสดุ PLA เครื่องพิมพ์สามมิติชนิด FDM ชนิด Direct extruder ขนาดเส้นใยฟิลาเมนต์ 2.85 มิลลิเมตร ขนาดของหัวฉีด 0.40 มิลลิเมตรและอุณหภูมิของเพลท 60 องศาเซลเซียส และการกำหนดเงื่อนไข และ 2) พารามิเตอร์แบบแปรผันโดยจะทำการสุ่มค่าโดยใช้ Random state เป็นไฮเปอร์พารามิเตอร์ของโมเดลใช้ในการควบคุมการสุ่มค่าที่เกี่ยวข้องกับโมเดลของการเรียนรู้ของเครื่องจักร สำหรับ Random state สามารถใช้จำนวนเต็มใด ๆ จำนวนเต็มที่นิยมได้แก่ 0 และ 42 เมื่อเราใช้ฟังก์ชั้น Random state จะให้ผลลัพธ์เดียวกันในการดำเนินการต่าง ๆ ผลลัพธ์จะเปลี่ยนเมื่อมีการเปลี่ยนค่าจำนวนเต็ม ซึ่งในงานวิจัยนี้จะใช้ Random state = 42 ในการสุ่มตัวแปรที่มีผลโดยตรงกับความเรียบของผิวชิ้นงาน ได้แก่อุณหภูมิที่หัวฉีด 170, 180, 190, 200, 210 องศาเซลเซียส (Tlegenov, Hong, & Lu, 2018) ซึ่งจะจำลองความแปรผันของอุณหภูมิโดยการใช้ปัจจัยภายนอกเพิ่มเติมใช้แอร์คอนดิชั่นโดยการเพิ่ม-ลด,ปิด-เปิดในระหว่างกระบวนการ, ระยะหางระหว่างหัวฉีดกับเพลท 0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.18, 0.20 มิลลิเมตร โดยใช้ Standard clearance Gauge (รูปที่3.3) และในงานวิจัยมีการจำลองสถานะ

ที่ผิดพลาดของสายพานโดยการนำลวดไปติดตั้งบนสายพาน [14] (รูปที่ 3.4) ความเร็วในการฉีด 20, 30, 40, 50, 60 มิลลิเมตร/วินาที และ ความหนาของชั้น 0.20, 0.25, 0.30 มิลลิเมตร (Wu, Wei, & Terpenny, 2019) สรุปได้ดังนี้ (ตารางที่ 3.1)

(ก) (ข)

รูปที่ 3.3 การใช้ Standard Clearance Gauge ในการวัดระยะห่างหัวฉีดกับเพลท

รูปที่ 3.3(ก) Standard Clearance Gauge ที่ใช้ในการกำหนดระยะห่างระหว่างหัวฉีดและเพลทโดยกำหนดระยะเป็น 0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.18, 0.20 มิลลิเมตร

รูปที่ 3.3(ข) คือวิธีการใช้ Standard clearance Gauge ในการกำหนดระยะห่างระหว่างหัวฉีดและเพลทโดยจะทำการตั้งระยะห่างก่อนเริ่มพิมพ์ทุกครั้งโดยจะทำการตั้งระยะห่างทั้งหมด 4 ตำแหน่ง ณ มุมของเพลทเพราะเป็นตำแหน่งที่สามารถปรับระดับได้

รูปที่ 3.4 การกำหนดสถานะผิดปกติด้วยการนำลวดไปติดตั้งบนสายพาน

การจำลองสถานะที่ผิดพลาดในสายพานของเครื่องพิมพ์สามมิติโดยอาศัยการติดตั้งลวดลงบนสายพาน

ตารางที่ 3.1 ตารางการกำหนดเงื่อนไข และพารามิเตอร์

3DPrint Type	FDM
Material	PLA
Nozzle diameter	0.4 mm
Filament diameter	1.75±0.50 mm
Bed temperature	60℃
FDM type	Direct extruder
Nozzle temperature	170, 180, 190, 200, 210 ℃
Nozzle distance (Standard Clearance Gauge)	0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.18, 0.20 mm
Status of Toothed belt	Wire prices install, uninstall
Extruder speed	20, 30, 40, 50, 60 mm/sec
Layer thickness	0.20, 0.25, 0.30 mm

เป็นการกำหนดเงื่อนไขและพารามิเตอร์ในงานวิจัย จากการทบทวนงานวิจัยพบว่ามีค่าพารามิเตอร์ที่ส่งผลโดยตรงกับชิ้นงานซึ่งงานวิจัยนี้จะใช้ในการศึกษาโดย ได้แก่ อุณหภูมิที่หัวฉีด, ระยะหางระหว่างหัวฉีดกับเพลท, สถานะที่ผิดพลาดของสายพาน,ความเร็วในการฉีด และ ความหนาของชั้น และมีเงื่อนไขคือศึกษาบนเครื่องพิมพ์สามมิติแบบหลอมรวม (FDM) โดยเป็นเครื่องพิมพ์ชนิด Direct extruder แบบเครื่องเปิด ใช้วัสดุเป็น PLA ขนาดเส้นใย 1.75±50 มิลิเมตร และ ขนาดหัวฉีดเท่ากับ 0.40 มิลิเมตร ประเภทของซัพพอร์ตชิ้นงานจะเป็น Skirt

3.3.3 การออกแบบและสร้างชิ้นงาน งานวิจัยนี้จะสร้างชิ้นงานที่มีลักษณะเป็นรูปทรงเรขาคณิตทั้งหมดลักษณะทางเรขาคณิตสามมิติ คือรูปลักษณะที่เป็นสามมิติ โดยนอกจากจะแสดงความกว้าง ความยาว จะแสดงความลึกหรือความนูนของชิ้นงาน ปริซึม มีหน้าตัดเป็นรูปหลายเหลี่ยมและอยู่ในระนาบเดียวกันมีหน้าข้างเป็นรูปสี่เหลี่ยม การเรียกชื่อปริซึมจะเรียกตามลักษณะหน้าตัดของฐาน เช่น ปริซึมสี่มิติ ปริซึมสามมิติทรงกระบอก มีลักษณะฐานทั้งสองข้างที่เท่ากันทุกประการและอยู่ในระนาบเดียวกันจำนวน 100 ชิ้นงานจากการสุ่มโดยใช้ Random state จาก 20 รูปแบบ ได้แก่

1) ปริซึมสามเหลี่ยม
2) ปริซึมสามเหลี่ยมด้านเท่า
3) ปริซึมสามเหลี่ยมหน้าจั่ว
4) ปริซึมสามเหลี่ยมมุมฉาก

5) ทรงลูกบาศก์
6) ปริซึมสี่เหลี่ยมพื้นผ้า
7) ปริซึมสี่เหลี่ยมขนมเปียกปูน
8) ปริซึมสี่เหลี่ยมด้านขนาน
9) ปริซึมสี่เหลี่ยมรูปว่าว
10) ปริซึมสี่เหลี่ยมคางหมู
11) ปริซึมสี่เหลี่ยมด้านไม่เท่า
12) ปริซึมห้าเหลี่ยม
13) ปริซึมหกเหลี่ยม
14) ปริซึมเจ็ดเหลี่ยม
15) ปริซึมแปดเหลี่ยม
16) ปริซึมเก้าเหลี่ยม
17) ปริซึมสิบเหลี่ยม
18) ทรงกระบอก
19) standard (ASTM D1708)
20) กรวย

หลังจากการสุ่มรูปทรงที่จะสร้างได้ครบทั้ง 100 ชิ้นจะทำการสุ่มค่าที่มีผลกับรูปทรงของชิ้นงาน เช่น ความกว้าง ความยาว ความสูง และรัศมี เป็นต้น โดยการสุ่มจะกำหนดขอบเขตอยู่ในช่วง 1-15 เซนติเมตร เพื่อไม่ให้ขนาดของชิ้นงานใหญ่กว่าขนาดของพื้นที่พิมพ์ หลังจากการสุ่มลักษณะของชิ้นงานจะทำการสร้างชิ้นงานโดยใช้คอมพิวเตอร์ช่วย

3.4 การวัดความหยาบผิว

ในงานวิจัยนี้ ใช้เครื่องวัดความหยาบผิวแบบสามมิติ (3D Profilometer) ซึ่งเป็นเครื่องมือที่มีความแม่นยำสูง โดยหลักการทำงานของเครื่องจะใช้หัววัดแบบปลายแหลม (Stylus) ลากผ่านพื้นผิวของชิ้นงาน เพื่อเก็บข้อมูลลักษณะทางกายภาพของพื้นผิวในลักษณะพิกัดตำแหน่งและความหยาบผิว จากนั้น เครื่องจะประมวลผลข้อมูลที่ได้ด้วยการแปลงสัญญาณไฟฟ้าเป็นคุณสมบัติของพื้นผิว โดยผลลัพธ์ที่ได้จะถูกวิเคราะห์เพื่อหาค่าความหยาบเฉลี่ยของพื้นผิว (Ra) ซึ่งแสดงถึงระดับความเรียบของพื้นผิว

ขั้นตอนการวัดและการเก็บข้อมูล

1) ตำแหน่งการวัดการวัดจะดำเนินการบนพื้นผิวด้านบนของชิ้นงาน ซึ่งเป็นบริเวณที่ไม่ได้สัมผัสกับเพลท (Build Plate) เนื่องจากผิวที่สัมผัสกับเพลทได้รับผลกระทบจากอุณหภูมิสูงของเพลท ทำให้พื้นผิวมีลักษณะเรียบกว่าพื้นผิวในบริเวณอื่น ๆ (ดังแสดงในรูปที่ 3.6)

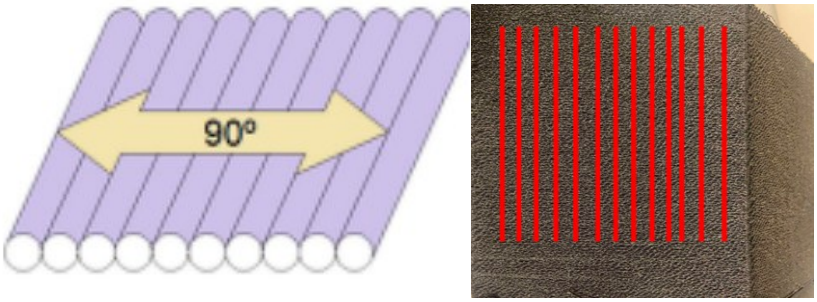
2) การกรองข้อมูลใช้ฟิลเตอร์ Gaussian เพื่อกรองข้อมูลที่ได้จากการวัด เพื่อลดสัญญาณรบกวนและเพิ่มความแม่นยำในการวิเคราะห์ค่าความหยาบ

3) การวิเคราะห์ค่าความหยาบผิว (Ra) ค่าความหยาบเฉลี่ย (Arithmetic Mean Roughness, Ra) จะถูกคำนวณจากส่วนเบี่ยงเบนของพื้นผิวชิ้นงานจากเส้นค่าเฉลี่ย โดยใช้ค่าทางสถิติจากพื้นที่ตัวอย่างในการวัด

มีงานวิจัยที่เลือกใช้ 3D Profilometer ในการศึกษาความหยาบผิวของชิ้นงานสามมิติที่พิมพ์ขึ้นรูปด้วยวัสดุ PLA โดยเน้นศึกษาผลกระทบของความหนาของชั้นพิมพ์ (Layer Thickness) ที่มีต่อค่าความหยาบผิว พบว่าชั้นพิมพ์ที่หนามักส่งผลให้ค่าความหยาบผิวเพิ่มขึ้น นอกจากนี้ งานวิจัยยังเน้นการเก็บข้อมูลลักษณะพื้นผิวในทิศทางที่ตั้งฉากกับทิศทางการอัดรีด (Perpendicular to Extrusion Direction) และในทิศทางที่ขนานกับการอัดรีด (Parallel to Extrusion Direction)

ผลการวิเคราะห์พบว่า พื้นผิวที่วัดในทิศทางตั้งฉากกับการอัดรีดมีค่าความหยาบผิว (Ra) สูงกว่าทิศทางขนาน และการจำแนกพื้นผิวในทิศทางตั้งฉากสามารถทำได้อย่างแม่นยำด้วยการใช้อัลกอริทึมการเรียนรู้ของเครื่องจักร การใช้งานเครื่องมือในการวัดเครื่อง 3D Profilometer ไม่เพียงแค่วัดค่าความหยาบผิวที่แม่นยำ แต่ยังช่วยให้สามารถสร้างโมเดลการคาดการณ์และการจำแนกประเภทพื้นผิวชิ้นงานได้ โดยอาศัยข้อมูลเชิงลึกจากค่าความหยาบที่วัดได้ทั้งสองทิศทาง เพื่อสร้างความเข้าใจในกระบวนการผลิตและผลกระทบต่อคุณภาพพื้นผิวของชิ้นงาน (ดังแสดงในรูปที่ 3.5 และ 3.6) งานวิจัยที่เกี่ยวข้องยังได้แสดงให้เห็นถึงความสำคัญของการวิเคราะห์ค่าความหยาบผิวในแต่ละทิศทาง เพื่อปรับปรุงกระบวนการพิมพ์สามมิติและเพิ่มประสิทธิภาพของชิ้นงานในแง่คุณภาพของพื้นผิวและความแม่นยำในการผลิต (Ayrilmis, 2018; Barrios & Romero, 2019)

รูปที่ 3.5 เครื่องวัดความหยาบผิว (Profilometer) ที่ใช้ในการวัดความหยาบผิวของชิ้นงาน



(ก) (ข)

รูปที่ 3.6 ลักษณะพื้นที่ในการเก็บข้อมูลของเครื่อง Profilometer

รูปที่ 3.6(ก) ทิศทางในการเก็บข้อมูลความหยาบผิวของชิ้นงานโดยจะมีทิศทางตั้งฉากกับการอัดรีด

รูปที่ 3.6(ข) ตัวอย่างพื้นที่ในการเก็บข้อมูลบนผิวชิ้นงาน การเก็บข้อมูลจะทำการเก็บทิศทางตั้งฉากกับการอัดรีดและการเก็บข้อมูลของชิ้นงานจะทำการเก็บข้อมูลผิวบนของชิ้นงาน (ผิวด้านที่ไม่ได้มีการสัมผัสกับเพลท)

3.5 การเก็บข้อมูล

มีงานวิจัยที่ใช้ Accelerometer และ AE sensor ร่วมกับ Machine Learningเพื่อทำนายความหยาบผิวของชิ้นงานระหว่างการกลึงแห้งเหล็กกล้า AISI 4140 ผลการวิจัยนี้พบว่าสามารถทำนายค่าความหยาบผิวได้ด้วยความแม่นยำสูง (Asiltürk, Kuntoğlu, Binali, Akkuş, & Salur, 2023)

โดยในงานวิจัยนี้จะมีการเก็บข้อมูลในงานวิจัยนี้จะทำการเก็บข้อมูลโดยใช้ Sensor ทั้งหมด 3 ชนิดได้แก่ thermocouple, Accelerometer และ AE sensor ชิ้นงานครั้งละ 1 ชิ้นโดยจะเก็บเป็นชุดข้อมูลอนุกรมเวลาตลอดระยะเวลาในการพิมพ์โดยภายใน 1 วินาที่จะทำการเก็บข้อมูล 2 ครั้ง ด้วยอุปกรณ์เก็บข้อมูล 3 ชนิด ได้แก่เซ็นเซอร์ที่ใช้สำหรับวัดการสั่นสะเทือนความเร่ง (Accelerometer), เซ็นเซอร์ที ใช้ในการจับสัญญาณเสียง (Acoustic Emission: AE) และการติดตั้งเทอร์โมคัปเปิล ทำการเก็บข้อมูลบนชิ้นงาน 100 ชิ้นที่มีลักษณะแตกต่างกัน โดยทั้งหมดจะถูกสุ่มเทียมโดย Random state สามารถสรุปได้ดัง (ตารางที่ 3.2) โดยชุดข้อมูลที่มีการเก็บจะรวบรวมตามระยะเวลาที่ติดต่อกันอย่างเป็นระบบ ชุดข้อมูล ประกอบด้วย อุณหภูมิที่เปลี่ยนแปลงไปตามช่วงเวลา, ระดับของเสียงที่เปลี่ยนไปตามช่วงเวลา และ การสั่นของ เครื่องพิมพ์สามมิติที่เปลี่ยนแปลงไปตามเวลา โดยการเก็บข้อมูลจะทำการเก็บตั้งแต่เริ่มพิมพ์ชิ้นงานจนเสร็จกระบวนการพิมพ์ โดยมีลักษณะการเก็บเวลาทุกวินาที่วินาทีละ 2 ครั้ง

การเก็บข้อมูลดิบแบบอนุกรมเวลา การวิเคราะห์ข้อมูลอนุกรมเวลาเป็นวิธีที่ใช้ในการวิเคราะห์ข้อมูลที่มี การเปลี่ยนแปลงไปตามเวลาที่เกิดขึ้น หรือมีการเปลี่ยนแปลงของตัวแปรในช่วงเวลาที่แตกต่างไป ซึ่งลักษณะของการเปลี่ยนแปลงอาจจะมีรูปแบบตายตัวหรือไม่มีก็ได้หรือช่วงของการเก็บข้อมูลเวลาจะเท่ากันหรือไม่ก็ได้ แต่เมื่ออนุกรมเวลาแสดงให้เห็นถึงความเปลี่ยนแปลงในช่วงเวลาที่ผ่านมามีความเป็นไปได้ที่จะสามารถคาดการณ์หรือพยากรณ์สิ่งที่จะเกิดขึ้นในอนาคตได้ (Forecasting Model) โมเดลการคาดการณ์ด้วยการเรียนรู้แบบอัตโนมัติ Machine Learning และการเรียนรู้แบบอัตโนมัติด้วยการเลียนแบบการทำงานของโครงข่ายประสาทเทียม Deep Learning สำหรับ Regression คือ การหาความสัมพันธ์ของตัวแปรต่าง ๆ ไม่ว่าจะเป็นอดีตของตัวมันเอง หรือตัวแปรอื่น ๆ ซึ่งไม่ได้จำกัดว่าจะต้องมีกี่ตัวแปร หรือจะต้องมีอนุกรมเวลาในลักษณะใด เพื่อมาคำนวณและทำนายผลที่จะเกิดขึ้นในอนาคต

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน

NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample1	Cubic	0.08	20	0.3	0	170	5.295	87420
sample2	Equilateral triangular prism	0.06	20	0.2	0	170	18.923	11100
sample3	Kite-shaped prism	0.14	40	0.2	1	190	25.539	9540
sample4	Parallelogram prism	0.12	30	0.3	0	180	4.535	4740
sample5	Parallelogram prism	0.12	30	0.25	0	180	20.678	10140
sample6	Square prism	0.1	30	0.2	0	180	17.676	8880
sample7	Cubic	0.08	20	0.2	0	170	27.563	18720
sample8	Cylindrical	0.08	60	0.2	0	210	26.354	660
sample9	Right triangular prism	0.18	20	0.3	1	170	28.972	14400
sample10	Standard (ASTM D1708)	0.06	60	0.2	0	210	23.515	300
sample11	Heptagonal prism	0.06	50	0.3	0	200	25.102	4440

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน (ต่อ)

NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample12	Isosceles triangular prism	0.08	20	0.3	0	170	26.147	1560
sample13	Equilateral triangular prism	0.12	20	0.3	0	170	28.535	24480
sample14	Right triangular prism	0.12	20	0.2	0	170	19.356	3240
sample15	Rhombus prism	0.06	30	0.3	0	180	18.154	17460
sample16	Parallelogram prism	0.12	30	0.25	0	180	18.277	8160
sample17	Decagonal prism	0.18	60	0.2	1	210	37.329	300
sample18	Conical	0.12	60	0.2	0	210	34.642	1320
sample19	Equilateral triangular prism	0.2	20	0.2	1	170	40.903	3420
sample20	Cylindrical	0.14	60	0.2	1	210	26.39	7800
sample21	Rhombus prism	0.06	30	0.2	0	180	22.68	2040
sample22	Cylindrical	0.1	60	0.3	0	210	38.366	12300
sample23	Heptagonal prism	0.18	50	0.3	1	200	25.099	2400
sample24	Parallelogram prism	0.16	30	0.2	1	180	26.925	15600
sample25	Octagonal prism	0.14	50	0.3	1	200	32.603	1920
sample26	Standard (ASTM D1708)	0.1	60	0.2	0	210	43.842	300
sample27	Kite-shaped prism	0.12	40	0.3	0	190	10.635	840
sample28	Equilateral triangular prism	0.16	20	0.3	1	170	25.345	5520

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน (ต่อ)

NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample29	rectangular prism	0.08	30	0.3	0	180	37.534	16920
sample30	heptagonal prism	0.08	50	0.3	0	200	23.384	2820
sample31	Rectangular prism	0.18	40	0.25	1	190	20.29	7200
sample32	Kite-shaped prism	0.08	40	0.2	0	190	44.593	12360
sample33	Square prism	0.16	30	0.25	1	180	28.251	1500
sample34	Rhombus prism	0.16	30	0.3	1	180	25.775	1200
sample35	Rectangular prism	0.14	40	0.25	1	190	29.921	2220
sample36	Cubic	0.06	20	0.2	0	170	31.003	2100
sample37	Right Triangular prism	0.2	20	0.2	1	170	20.463	12540
sample38	Hexagonal prism	0.08	50	0.3	0	200	36.737	8460
sample39	Cubic	0.18	20	0.25	1	170	34.746	18720
sample40	Pentagonal prism	0.08	40	0.25	0	190	23.462	1860
sample41	pentagonal prism	0.14	40	0.25	1	190	22.55	11760
sample42	conical	0.16	60	0.2	1	210	49.781	1860
sample43	Kite-shaped prism	0.12	40	0.2	0	190	24.636	3240
sample44	Isosceles triangular prism	0.08	20	0.25	0	170	37.435	10740
sample45	Octagonal prism	0.06	50	0.2	0	200	24.586	2280
sample46	Cylindrical	0.12	60	0.2	0	210	30.081	5640
sample47	Cubic	0.14	20	0.25	1	170	19.845	10080
sample48	Hexagonal prism	0.08	50	0.2	0	200	22.657	14220

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน (ต่อ)

NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample49	Right triangular prism	0.12	20	0.25	0	170	18.881	8760
sample50	Cylindrical	0.08	60	0.25	0	210	41.75	1200
sample51	Trapezoidal prism	0.18	40	0.3	1	190	30.893	11220
sample52	Conical	0.14	60	0.25	1	210	36.648	1260
sample53	Pentagonal prism	0.2	40	0.2	1	190	22.631	960
sample54	Standard (ASTM D1708)	0.16	60	0.3	1	210	32.479	300
sample55	Rhombus prism	0.1	30	0.25	0	180	24.718	3600
sample56	Right triangular prism	0.16	20	0.3	1	170	27.547	6480
sample57	Isosceles triangular prism	0.16	20	0.2	1	170	24.736	1680
sample58	Parallelogram prism	0.12	30	0.25	0	180	18.029	5100
sample59	Trapezoidal prism	0.14	40	0.2	1	190	19.448	2160
sample60	Right triangular prism	0.08	20	0.3	0	170	23.079	13860
sample61	Parallelogram prism	0.1	30	0.25	0	180	30518	13260
sample62	Cubic	0.12	20	0.3	0	170	26.304	11580
sample63	Hexagonal prism	0.1	50	0.3	0	200	45.257	1920
sample64	Kite-shaped prism	0.2	40	0.25	1	190	41.149	3780
sample65	Octagonal prism	0.18	50	0.3	1	200	24.315	960
sample66	Pentagonal prism	0.14	40	0.2	1	190	23433	2040

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน (ต่อ)

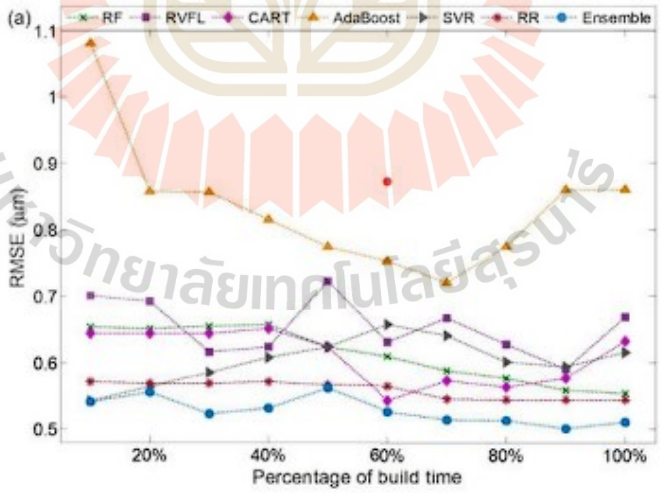
NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample67	Rectangular prism	0.12	30	0.3	0	180	9.639	17160
sample68	Pentagonal prism	0.16	40	0.2	1	190	23.258	18300
sample69	Pentagonal prism	0.06	40	0.2	0	190	34.699	2160
sample70	Rhombus prism	0.12	30	0.3	0	180	36.436	4320
sample71	Kite-shaped prism	0.06	40	0.2	0	190	12.747	9240
sample72	Right triangular prism	0.16	20	0.25	1	170	28.245	8700
sample73	Conical	0.18	60	0.2	1	210	22.664	8040
sample74	Rectangular prism	0.14	30	0.2	1	180	17.672	10620
sample75	Cylindrical	0.08	60	0.2	0	210	15.331	1860
sample76	Parallelogram prism	0.12	30	0.25	0	180	29.194	1140
sample77	Rectangular prism	0.16	30	0.25	1	180	23.503	6960
sample78	Octagonal prism	0.12	50	0.25	0	200	24.431	3600
sample79	Hexagonal prism	0.2	50	0.3	1	200	24.979	6900
sample80	Kite-shaped prism	0.18	40	0.25	1	190	27.838	151200
sample81	Cylindrical	0.2	60	0.2	1	210	44.062	960
sample82	Parallelogram prism	0.1	30	0.25	0	180	17.958	8100
sample83	Rectangular prism	0.14	40	0.25	1	190	28.4	4380
sample84	Isosceles triangular prism	0.1	20	0.2	0	170	22.492	9480

ตารางที่ 3.2 ตารางข้อมูลชิ้นงาน (ต่อ)

NO	Type of shape	Nozzle distance	Extruder speed	Layer thickness	status Toothed belt	Nozzle temperature (Celsius)	Ra (μm)	Total time (sec)
sample85	Parallelogram prism	0.12	30	0.3	0	180	33.004	27300
sample86	Isosceles triangular prism	0.14	20	0.25	1	170	39.299	9480
sample87	Rectangular prism	0.18	40	0.3	1	190	26.984	600
sample88	Hexagonal prism	0.18	50	0.3	1	200	33.871	20400
sample89	Kite-shaped prism	0.16	40	0.3	1	190	18.975	4440
sample90	Right triangular prism	0.12	20	0.2	0	170	29.946	25140
sample91	Rhombus prism	0.1	30	0.3	0	180	37.685	7260
sample92	Standard (ASTM D1708)	0.2	60	0.3	1	210	32.234	300
sample93	Rectangular prism	0.08	40	0.2	0	190	22.271	18780
sample94	Rhombus prism	0.06	30	0.3	0	180	18.769	2040
sample95	Hexagonal prism	0.08	50	0.3	0	200	28.36	1260
sample96	Hexagonal prism	0.1	50	0.2	0	200	20.351	6120
sample97	Octagonal prism	0.1	50	0.2	0	200	20.251	13800
sample98	Square prism	0.18	30	0.25	1	180	20.039	65160
sample99	Kite-shaped prism	0.08	40	0.25	0	190	23.379	5820
Sample 100	Square prism	0.18	30	0.25	1	180	16.814	1620

3.6 การแปลงข้อมูล

เนื่องจากงานวิจัยนี้มีความประสงค์ที่จะสร้างอัลกอริทึมที่สามารถทำนายผลของผิวก่อนเสร็จกระบวนการบนชิ้นงานที่มีรูปร่าง เวลาในการพิมพ์ที่แตกต่างกัน การเก็บข้อมูลช่วงเวลาในการทำงานของเครื่องจะ ได้มาซึ่งชุดข้อมูลที่มีปริมาณข้อมูลที่แตกต่างกัน และเนื่องจากไม่ทราบว่า ชุดข้อมูลในช่วงเวลาใดที่มีผลกับความ หยาบผิวมากที่สุด จึงได้ออกแบบการทดลองโดยอาศัยการแบ่งช่วงเวลาการผลิตชิ้นงานทั้งหมดและจำลองปริมาณ ชุดข้อมูล ณ ช่วงเวลาที่แตกต่างกัน โดยในงานวิจัยนี้เลือกนำชุดข้อมูลที่ได้จากการแบ่งช่วงเวลาของการเก็บข้อมูล แบบอนุกรมเวลาเป็นร้อยละของช่วงเวลาทั้งหมด 5 ช่วงเวลาได้แก่ ร้อยละ 20, 40, 60, 80 และ 100 ของเวลาใน การพิมพ์ทั้งหมด โดยการแยกข้อมูลเป็นร้อยละเพื่อจำลองการประมวลผลของอัลกอริทึมที่เวลาที่แตกต่างกันและมีปริมาณชุดข้อมูลในการทำนายผลลัพธ์ไม่เท่ากัน จะส่งผลต่อมีประสิทธิภาพของโมเดล (ความแม่นยำ และความไวในการทำนายผล) แตกต่างกันมากน้อยเพียงใดและสามารถตัดสินใจได้ว่า ณ ช่วงเวลาใดมีประสิทธิภาพในการทำนายผลของโมเดลที่สร้างมากที่สุด (ความแม่นยำ และความไวในการทำนายผล) มีงานวิจัยใช้วิธีการแบ่งช่วงเวลาเพื่อเช็คประสิทธิภาพการทำนายผิวของชิ้นงานที่ถูกพิมพ์ด้วยกระบวนการ FDM พบว่า ณ ช่วงเวลาเดียวกันโมเดลต่างชนิดมีประสิทธิภาพการทำนายไม่เท่ากันและ ณ ช่วงเวลาที่แตกต่างกันโมเดลเดียวกันมีประสิทธิภาพไม่เท่ากันและสามารถเพิ่มลดได้ (Guin, 2006) รูปที่ 3.7



รูปที่ 3.7 กราฟแสดงความสัมพันธ์ระหว่างร้อยละของเวลาทำงานและความแม่นยำของโมเดล

พบว่า ณ ช่วงเวลาที่แตกต่างกันประสิทธิภาพของโมเดลสามารถเพิ่มหรือลดได้ขึ้นอยู่กับจำนวนข้อมูลในการ ทำนายผลและอัลกอริทึมที่ใช้ในการทำนายผล

การเก็บข้อมูลดิบแบบอนุกรมเวลา การวิเคราะห์ข้อมูลอนุกรมเวลาเป็นวิธีที่ใช้ในการวิเคราะห์ข้อมูลที่มี การเปลี่ยนแปลงไปตามเวลาที่เกิดขึ้น หรือมีการเปลี่ยนแปลงของตัวแปรในช่วงเวลาที่แตกต่างไป ซึ่งลักษณะของการเปลี่ยนแปลงอาจจะมีรูปแบบตายตัวหรือไม่มีก็ได้หรือช่วงของการเก็บข้อมูลเวลาจะเท่ากันหรือไม่ก็ได้ แต่เมื่ออนุกรมเวลาแสดงให้เห็นถึงความเปลี่ยนแปลงในช่วงเวลาที่ผ่านมามีความเป็นไปได้ที่จะสามารถคาดการณ์ หรือ พยากรสิ่งที่จะเกิดขึ้นในอนาคตได้ (Forecasting Model) ปัจจุบันมีการสร้างโมเดลการคาดการณ์สิ่งที่จะเกิดขึ้น ในอนาคต (Forecasting Model) โดยอาศัยการเรียนรู้ของเครื่อง (Machine Learning) และการเลียนแบบการทำงานของโครงข่ายประสาทเทียม (Deep Learning) สำหรับการทำนายในลักษณะ Regression คือ การหาความสัมพันธ์ของตัวแปรต่าง ๆ ไม่ว่าจะเป็นผลของอดีตที่เคยเกิดขึ้นหรือตัวแปรอื่นๆที่ส่งผลกระทบกับปัจจุบัน ซึ่งไม่ได้จำกัดว่าจะต้องมีกี่ตัวแปรหรือจะต้องมีอนุกรมเวลาในลักษณะใด เพื่อมาคำนวณและทำนายผลที่จะเกิดขึ้นในอนาคต

3.6.1 อนุกรมเวลาของอุณหภูมิจะทำการตรวจสอบว่าเป็นอนุกรมเวลาที่คงที่หรือไม่ หากเป็นอนุกรมเวลาที่คงที่ จะแปลงอนุกรมเวลาที่คงที่ของอุณหภูมิเป็นอนุกรมเวลาที่ไม่คงที่เพื่อให้ชุดข้อมูลไม่ขึ้นอยู่กับชิ้นงาน การทดสอบ Unit Root เป็นการทดสอบเพื่อให้ทราบว่าชุดข้อมูลอนุกรมเวลาคงที่(Stationary) แสดงในสมการที่ 3.1-3.3 หรือเป็นชุดอนุกรมเวลาที่ไม่คงที่ (Nonstationary) แสดงในสมการที่ 3.4-3.6

$$\text{ค่าเฉลี่ย} = E(X_t) = u \tag{3.1}$$

$$\text{ความแปรปรวน} = Var(X_t) = E(X_t - u^2) = \sigma^2 \tag{3.2}$$

$$\text{ความแปรปรวนร่วม} = E[(X_{t-u})(X_{t-k} - u)] = \gamma_k \tag{3.3}$$

เมื่อสมมุติว่า X1 เป็นอนุกรมเวลาที่ไม่คงที่ (Nonstationary) มีคุณสมบัติดังนี้

$$\text{ค่าเฉลี่ย} = E(X_t) = t_u \tag{3.4}$$

$$\text{ความแปรปรวน} = Var(X_t) = E(X_t - u^2) = t\sigma^2 \tag{3.5}$$

$$\text{ความแปรปรวนร่วม} = E[(X_{t-u})(X_{t-k} - u)] = t\gamma_k \tag{3.6}$$

3.6.2 แปลงอนุกรมเวลาของการสั่นสะเทือนและระดับเสียงเป็นอนุกรมความถี่ของการสั่นสะเทือนและ ระดับเสียงเพื่อไม่ให้ชุดข้อมูลไม่ขึ้นอยู่กับเวลา การแปลงชุดข้อมูลอนุกรมฟูเรียร์ (Fourier series) เป็นเครื่องมือทางคณิตศาสตร์ที่ใช้ในการแปลงสัญญาณ คาบให้อยู่ในรูปแบบของผลรวมเชิงเส้นของเลขชี้กำลังเชิงซ้อนซึ่งมีประโยชน์มากสำหรับการวิเคราะห์สัญญาณการแปลงฟูเรียร์ทำหน้าที่ในการแปลงสัญญาณในอนุกรมเวลาหรือสัญญาณที่เป็นฟังก์ชันของเวลาให้อยู่ในรูปของสัญญาณในอนุกรมความถี่หรือสัญญาณที่เป็นฟังก์ชันของความถี่ การวิเคราะห์สัญญานความถี่ง่ายกว่าการ วิเคราะห์การวิเคราะห์สัญญานในอนุกรมเวลา และสัญญาณในอนุกรมความถี่ยังบอกให้ทราบถึงลักษณะของ สัญญานซึ่งช่วยให้เข้าใจสมบัติต่าง ๆ และความผิดพลาดของสัญญานได้มากขึ้น แสดงในสมการ 3.7-3.9

$$a_0 = \frac{1}{T}\int_{t_0}^{t_0+T_0} g\,(t)dt \tag{3.7}$$

สามารถเขียนได้ในรูปอนุกรมฟูเรียร์นรูปเอ็กซ์โพเนนเชียล

$$g(t) = \sum_{n=-\infty}^{\infty} G_n\, e^{jnw_0}; w_0 = 2\pi f_0 \tag{3.8}$$

โดยที่ G_n คือค่าสัมประสิทธิเชิงซ้อนและสามารถหาได้จาก

$$gG_n = \frac{1}{T}\int_{t_0}^{t_0+T_0} g\,(t)e^{-jnw_0 t}dt \tag{3.9}$$

เมื่อ f = ความถี่พื้นฐาน (Fundamental Frequency) ของสัญญาณ $g(t)$ และ
f_0 = f/T
a_0 = ค่าเฉลี่ยของ $g(t)$

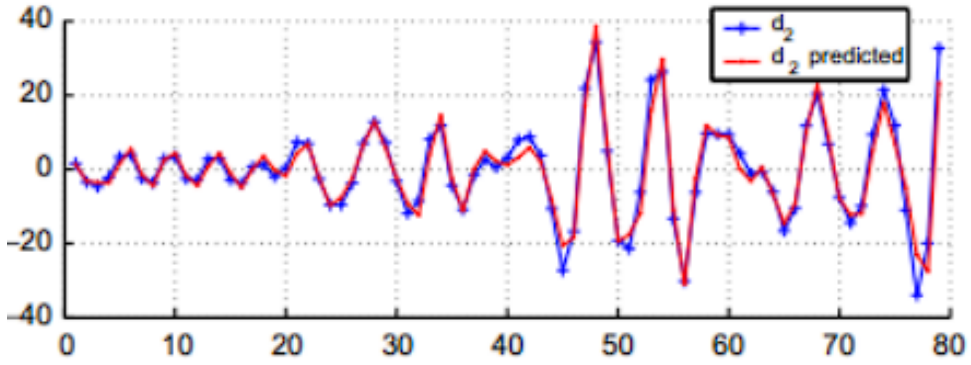
3.7 การฝึกโมเดล Machine Learning และ Deep Learning

การฝึกโมเดลโดยใช้โปรแกรม ภาษา Python ด้วย อัลกอริทึมกรอบการเรียนรู้ของเครื่องออกแบบมาเพื่อเลือกพฤติกรรมที่สำคัญคุณลักษณะและคาดการณ์ระดับ ความเสี่ยงซึ่งรวมคุณลักษณะตามการเรียนรู้ในการสร้างโมเดลเราจะใช้ชุดข้อมูลของตัวแปรที่กำหนดขึ้น (อินพุต) ซึ่งส่งผลกับผลลัพธ์ (เอาต์พุต) ในการสอนอัลกอริทึมเพื่อให้อัลกอริทึมสามารถสร้างฟังก์ชันหรือความเชื่อมโยงระหว่างอินพุต และ เอาต์พุต เมื่อทราบถึงความเชื่องโยงของตัวแปร และ ผลลัพธ์ โมเดลจะ

พร้อมในการใช้ทำนายผลลัพธ์ถัดไป ในส่วนนี้จะใช้ชุดข้อมูลส่วนหนึ่งของข้อมูลทั้งหมดโดยงานวิจัยนี้จะใช้ร้อยละ 70 ของข้อมูลทั้งหมดในการสร้าง โดยเรียกชุดข้อมูลส่วนนี้ว่า “ชุดข้อมูลเรียนรู้ (Training Data)” หลังจากทำการสร้างโมเดลจะใช้ชุด ข้อมูลที่ถูกแบ่งไว้อีกส่วนทดสอบความแม่นยำของโมเดลโดยจะใช้ชุดข้อมูลที่เป็นตัวแปรให้โมเดลทำนายผลลัพธ์ และเปรียบเทียบกับผลลัพธ์ที่เกิดขึ้นจริงว่ามีความแตกต่างกันมากน้อยเพียงใดซึ่งจะสามารถบอกได้ถึง ประสิทธิภาพในการทำนายผลของโมเดล ในส่วนนี้จะใช้ชุดข้อมูลส่วนหนึ่งของข้อมูลทั้งหมดโดยงานวิจัยนี้จะใช้ ร้อยละ 30 ของข้อมูลทั้งหมด ซึ่งเป็นชุดข้อมูลที่ไม่เกี่ยวข้องกับ ชุดข้อมูลเรียนรู้ (Training Data) โดยเรียกชุด ข้อมูลส่วนนี้ว่า “ชุดข้อมูลทดสอบ (Test data)” มีงานวิจัยศึกษาผลของการเลือกปริมาณ ชุดข้อมูลเรียนรู้ และ ชุดข้อมูลทดสอบต่อประสิทธิภาพของโมเดลโดยทดสอบบนอัลกอริทึมที่แตกต่างกัน 4 อัลกอริทึม และ เปรียบเทียบระหว่าง ชุดข้อมูลเรียนรู้ต่อชุดข้อมูลทดสอบ ร้อยละ 50 ต่อ 50 เปรียบเทียบกับชุดข้อมูลเรียนรู้ต่อชุดข้อมูลทดสอบร้อยละ 70 ต่อ 30 พบว่าที่ชุดข้อมูลเรียนรู้ต่อชุดข้อมูลทดสอบร้อยละ 70 ต่อ 30 มีประสิทธิภาพการ ทำนายผลสูงกว่า (Li, Zhang, Shi, & Wu, 2019)

ชุดข้อมูลเรียนรู้ (Training Data) เป็นการสร้างฟังก์ชันที่สามารถอธิบายถึงลักษณะความสัมพันธ์ของข้อมูล ซึ่งฟังก์ชันนี้จะสามารถทำนายผลที่เกิดจากตัวแปรต่างๆที่เกิดขึ้นในอนุกรมเวลา และ สรุปออกมาเป็นรูปแบบความสัมพันธ์ได้ ชุดข้อมูลทดสอบ (Test data) เป็นการตรวจสอบฟังก์ชันที่สร้างขึ้นเพื่อทำนายผลว่ามีความแม่นยำมาก น้อยเพียงใด โดยใช้ชุดข้อมูลที่นอกเหนือจากชุดข้อมูลเรียนรู้ เป็นขั้นตอนในการตรวจสอบความถูกต้อง และอาจมีการปรับปรุงฟังก์ชันในการทำนายผลเพื่อให้ได้ประสิทธิภาพที่ดีขึ้น การเลือกแบบ AR, ARIMA, XGBoost, SVR, LSTM และ RNN โดยทั้งหมดจะเป็นอัลกอริทึมที่มี ความสามารถทำนายชุดข้อมูลในลักษณะอนุกรมเวลาและอนุกรทความถี่ได้ โดยจะมีการทำนายในลักษณะ Regression เพื่อให้ได้ค่าความหยาบผิวของชิ้นงาน

3.7.1 Autoregressive (AR) Autoregressive เป็นหนึ่งในเทคนิคที่ใช้ในการวิเคราะห์อนุกรมเวลา แบบจำลองการ ถดถอยอัตโนมัติคือแบบจำลองอนุกรมเวลาที่อธิบายว่าค่าในอดีตของตัวแปรหนึ่งๆ มีอิทธิพลต่อค่าปัจจุบันอย่างไร กล่าวอีกนัยหนึ่ง โมเดล AR พยายามทำนายค่าถัดไปในชุดข้อมูลโดยรวมค่าจากค่าที่ผ่านมาล่าสุด แบบจำลองการถดถอยอัตโนมัติกำลังฝึกตัวแบบการถดถอยกับค่าของตัวแปรตอบสนองเองจากการคาดการณ์อนุกรมเวลา การสร้าง แบบจำลองการถดถอยอัตโนมัติจะหมายถึงการสร้างแบบจำลองที่ตัวแปรตอบสนอง Y จะขึ้นอยู่กับค่าก่อนหน้า ของ Y ที่เวลาช่วงเวลาคงที่ที่กำหนดไว้ล่วงหน้าโมเดล AR สามารถใช้เพื่อสร้างแบบจำลองอะไรก็ได้ที่มีระดับของ ความสัมพันธ์อัตโนมัติ ซึ่งหมายความว่ามีความสัมพันธ์ระหว่างการตรวจสอบในขั้นตอนเวลาที่ติดกัน มีข้อจำกัดคือ จะต้องทำงานบนชุดข้อมูลมีลักษณะนิ่ง (Stationary) หน้าโดย Y_t กำหนดว่าที่ค่าจริง ที่เวลา t ใด ๆ ขึ้นอยู่กับค่าจริงที่เวลา รูปที่ 3.8



รูปที่ 3.8 การทำงานของ Autoregressive(AR)

การทำนายจะคำนวณจากตัวแปรที่ส่งผลกับผลลัพธ์ในอดีตและส่งผลกับอนาคต Autoregressive Integrated Moving Average (ARIMA)แต่สามารถทำงานไม่ได้บนชุดข้อมูลที่ไม่คงที่ (non-stationary) t-1,t-2,t-3,t-4,...t-p โดยที่ P เป็นคาบเวลา เขียนได้ในรูปแบบสมการที่ 3.10

$$Y_t = \theta_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t \qquad (3.10)$$

เมื่อ Y_t คือค่าจริงที่เวลาใด ๆ

θ_0 คือค่าคงที่

ϕ_1 คือค่าพารามิเตอร์ถ่วงน้ำหนัก

ε_t คือค่าคาดเคลื่อนที่เวลา t

3.7.2 Autoregressive Integrated Moving Average (ARIMA) แบบจำลองนี้คือการทำนายการเคลื่อนไหวของ อนุกรมเวลาในอนาคตโดยตรวจสอบความแตกต่างระหว่างค่าในชุดข้อมูลแทนที่จะพิจารณาจากค่าจริง แบบจำลอง ARIMA จะใช้ในกรณีที่ข้อมูลแสดงความไม่คงที่ในการวิเคราะห์อนุกรมเวลา ข้อมูลที่ไม่คงที่ จะถูกแปลงเป็นข้อมูลคงที่เสมอ เป็นการทำนายลักษณะเดียวกับ Autoregressive (AR) แต่สามารถทำงานบนชุด ข้อมูลมีลักษณะไม่นิ่งได้ (Non- Stationary) ซึ่งเทคนิค ARIMA เป็นเทคนิคที่ได้รับความนิยมสูงเนื่องจากให้ผลการ พยากรณ์ที่มีความแม่นยำเป็นการนำเอา Autoregressive (AR) พัฒนารวมกับ Moving average (MA) คือ ค่าเฉลี่ยลักษณะเชิงเส้นของความผิดพลาดเทอมของข้อผิดพลาด ที่ระบบควบคุมระหว่าง แบบจำลองค่าเฉลี่ย เคลื่อนที่กับ การสังเกตย้อนหลังในอดีตก่อนหน้ารูปที่ 3.9 แสดงในสมาการที่ 3.11

$$Y_t = \theta_0 + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_p \varepsilon_{t-p} + \varepsilon_t \qquad (3.11)$$

เมื่อ Y_t คือค่าจริงที่เวลาใด ๆ

θ_0 คือค่าคงที่

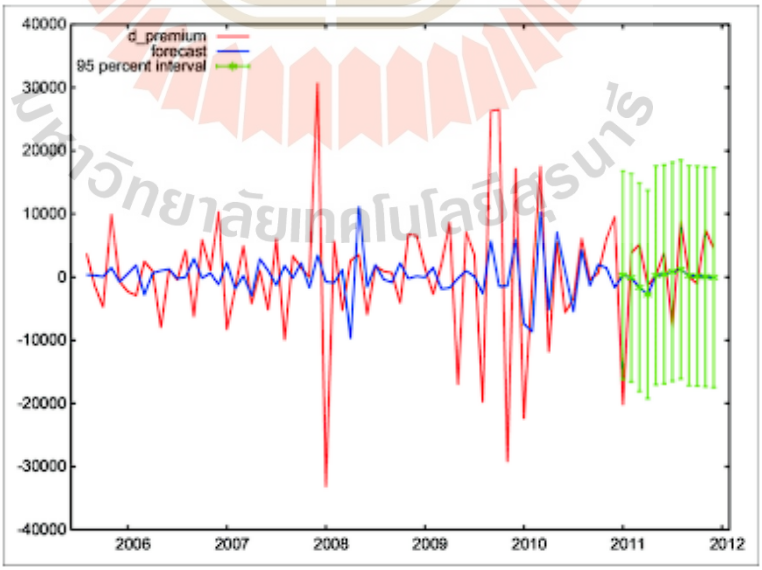
ϕ_1 คือค่าพารามิเตอร์ถ่วงน้ำหนัก

ε_t คือค่าคาดเคลื่อนที่เวลา t

เมื่อนำเอา Autoregressive (AR) และ กับ Moving average (MA) จะได้สมาการทำนายอนุกรมเวลา Autoregressive Integrated Moving Average (ARIMA) เขียนในรูปสมาการ 3.12

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t + \varepsilon_t + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_p \varepsilon_{t-p} \quad (3.12)$$

มีงานวิจัยเก็บข้อมูลระยะเวลาในการเดินทางและพัฒนาโมเดลในการทำนายระยะเวลาในการเดินทางโดย เก็บจากชุดข้อมูลที่มีความแปรผันและไม่ต่อเนื่อง โดยการเดินทางไม่มีรูปแบบตายตัว และใช้อัลกอริทึม Autoregressive Integrated Moving Average (ARIMA) ในการสร้างโมเดลในการทำนายผล ผลสรุปของงานวิจัยชี้ให้เห็นว่า Autoregressive Integrated Moving Average (ARIMA) สามารถทำนายชุดข้อมูลที่ไม่คงที่ และใช้ชุดข้อมูลที่ไม่ต่อเนื่องได้ และมีประสิทธิภาพที่สูง (Sing et al., 2021)



รูปที่ 3.9 การทำนายผลของ Autoregressive Integrated Moving Average (ARIMA)

การทำนายผลของ Autoregressive Integrated Moving Average (ARIMA)จะคำนวณจากตัวแปรที่ส่งผลกับ ผลลัพธ์ในอดีตและส่งผลกับอนาคตเช่นเดียวกับ Autoregressive (AR) แต่สามารถทำงานได้ดีบนชุดข้อมูลที่ไม่คงที่ (non-stationary)

เนื่องจาก Autoregressive Integrated Moving Average (ARIMA) และ Autoregressive (AR) เป็นอัลกอริทึมที่มีความนิยมมากในการทำนายผลชุดข้อมูลอนุกรมเวลา และทั้ง 2 อัลกอริทึมมีลักษณะการทำนายผลที่ คล้ายกันคือการคำนวณจากตัวแปรที่มีผลกระทบต่อผลลัพธ์ในอดีตเพื่อทำนายผลลัพธ์ในปัจจุบัน โดยความแม่นยำจะขึ้นอยู่กับชุดข้อมูลด้วยส่วนหนึ่ง จึงเป็นหนึ่งปัจจัยที่เลือก 2 อัลกอริทึมนี้มาทำนายผลลัพธ์ของการพิมพ์สามมิติ

3.7.3 XGBoost เป็นอัลกอริทึมที่ใช้การตัดสินใจโดยต้นไม้ตัดสินใจ ซึ่งสามารถประเมินความสำคัญสัมพัทธ์ของคุณลักษณะต่างๆ โดยใช้เมตริกต่างๆ เช่น จำนวนครั้งที่ฟีเจอร์ถูกใช้ในการแบ่งข้อมูลในทุกต้นไม้ คุณลักษณะที่มีความสำคัญมากจะถูกนำมาใช้บ่อยขึ้นในการสร้างต้นไม้เพิ่มเติม ส่วนที่เหลือจะถูกใช้เพื่อปรับปรุงประสิทธิภาพของโมเดล XGBoost สามารถเลือกชุดย่อยของคุณลักษณะที่เหมาะสมที่สุดโดยอิงจากการแลกเปลี่ยนระหว่างประสิทธิภาพการเรียนรู้ของโมเดล เป็นอัลกอริทึมที่ไม่มีพารามิเตอร์ซึ่งทำงานโดยใช้โครงสร้างที่คล้ายต้นไม้และทำการตัดสินใจที่สอดคล้องกับคลาสของเป้าหมาย โดยที่แต่ละโหนดในแผนผังการตัดสินใจจะแสดงถึงแอตทริบิวต์

ในกรอบการทำงานแบบ Boosting ค่าเฉลี่ยของค่าเป้าหมายจะถูกคำนวณและข้อผิดพลาดที่เหลือจะเริ่มต้นด้วยความสอดคล้องกัน ข้อผิดพลาดที่เหลือนี้จะได้รับการฝึกด้วยตัวแปรอิสระและใช้เป็นข้อมูลเพื่อคาดการณ์ ผลการคาดการณ์จะคำนวณร่วมกับผลลัพธ์ก่อนหน้าจากการคำนวณแต่ละขั้นจนถึงขั้นที่ M การคำนวณในขั้นสุดท้ายจะเป็นผลรวมของการคำนวณทั้งหมดก่อนหน้านี้

การวิจัยเกี่ยวกับโรคแท้งติดต่อในประเทศจีนได้มีการเก็บข้อมูลเป็นชุดอนุกรมเวลาและสร้างโมเดลด้วยอัลกอริทึม XGBoost และ Autoregressive Integrated Moving Average (ARIMA) เพื่อทำนายผลและเปรียบเทียบประสิทธิภาพ พบว่าทั้งสองโมเดลสามารถทำนายผลได้ดี โดยที่ XGBoost มีประสิทธิภาพสูงกว่า ARIMA โดยเฉพาะในแง่ของความเร็วในการประมวลผลและความแม่นยำ (ตารางที่ 3.3) (Chang & Lin, 2011)

ตารางที่ 3.3 ตารางเปรียบเทียบประสิทธิภาพระหว่าง ARIMA และ XGBoost

	Training set			Test set		
Model	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**	**MAPE(%)**
ARIMA(0,1,1)×(0,1,1) $_{12}$	338.867	450.223	10.323	529.406	586.059	17.676
XGBoost	189.332	262.458	4.475	249.307	280.645	7.643

ปัจจัยที่เลือก XGBoost เป็นหนึ่งในอัลกอริทึมต้นไม้ตัดสินใจที่ได้รับความนิยมมากที่สุด เพราะมีความแม่นยำสูง XGBoost ได้รับการยอมรับว่าใช้งานง่ายกว่าสำหรับชุดข้อมูลขนาดเล็กที่

ทำงานบน CPU ใช้ฐานข้อมูลน้อยกว่าการเรียนรู้เชิงลึก มีความเป็นไปได้ที่ XGBoost อาจจะมีประสิทธิภาพลดลงได้หากมีชุดข้อมูลที่มากเมื่อเทียบกับ ARIMA

3.7.4 Support Vector Regression (SVR) หรือ ซัพพอร์ตเวกเตอร์รีเกรสชัน เป็นเทคนิคหนึ่งที่ใช้วิธีการซัพพอร์ตเวกเตอร์แมชชีน (SVM) ซึ่งใช้ในการวิเคราะห์ความสัมพันธ์ระหว่างอินพุตและเอาต์พุต โดยการประยุกต์ใช้ในการทำนายอนุกรมเวลา โดยมีการแบ่งคลาสโดยใช้ SVM และใช้ SVR ในการทำนายผลลัพธ์ของอนุกรมเวลา เทคนิคนี้สามารถทำงานได้ดีบนชุดข้อมูลที่มีลักษณะไม่เป็นเชิงเส้น ซึ่ง SVR จะช่วยคำนวณค่าที่อาจเกิดขึ้นในอนาคตโดยอาศัยข้อผิดพลาดในอดีต โดยใช้ฟังก์ชันเคอร์เนล (Kernel function) ที่สามารถเปลี่ยนข้อมูลจากมิติที่ต่ำกว่าไปสู่มิติที่สูงขึ้น เพื่อเพิ่มประสิทธิภาพในการทำนายผล

การศึกษาเกี่ยวกับประสิทธิภาพในการทำนายของอัลกอริทึม SVR พบว่า SVR มีประสิทธิภาพสูงในการทำนายผลลัพธ์ (รูปที่ 3.10) โดยมีงานวิจัยที่ได้นำ SVR มาใช้ในการทำนายการไหลและการระบายน้ำ โดยใช้ชุดข้อมูลที่เก็บจากอดีตในการฝึกอบรมโมเดล ซึ่งใช้ข้อมูล 70% สำหรับการฝึกอบรม และ 30% สำหรับการทดสอบ โดยมีการใช้วิธีการตรวจสอบไขว้ห้าเท่า (5-fold cross-validation) ผลลัพธ์ยืนยันว่า SVR สามารถทำนายผลลัพธ์ได้อย่างแม่นยำ (Sashoo et al., 2019) เขียนได้ในรูปสมการ 3.13-3.15

$$f(x) = w_t x + b \tag{3.13}$$

เมื่อ F และ b เป็นความชันทดถอย

สามารถเขียนสมการSVR เพื่อทำนายค่าเอาต์พุตจากอินพุตได้โดยสมาการ

$$f(x) = w_0 t_x + b = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) k(x_i, x) + b \tag{3.14}$$

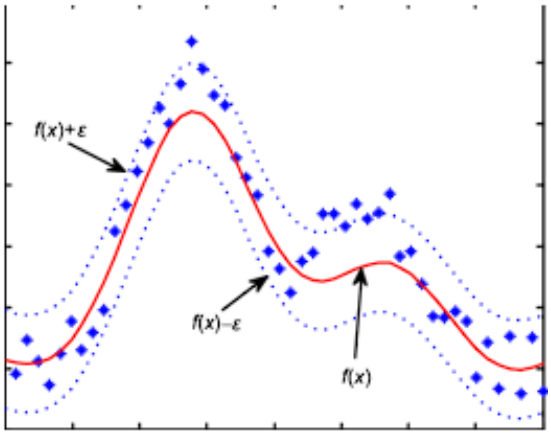
โดยที่เวกเตอร์ถ่วงน้ำหนักเป็นไปดังสมการ

$$w_0 = \sum\sum_{i=1}^{l} (\alpha i - \alpha i^*) xi \tag{3.15}$$

สามารถสร้างสมการการถดถอยแบบไม่เชิงเส้นโดยอาศัยฟังก์ชั่นเคอร์เนล (Kernel function) โดยที่นิยมใช้มีด้วยกันทั้งหมด2ลักษณะ

(1) ฟังก์ชั่นเคอร์เนลแบบเส้น (Linear Kernel) : $k(x_i, x) = x_i^t x$

(2) ฟังก์ชั่นเคอร์เนลแบบโพลีโนมียล (Polynomial Kernal) : $(1+x_i, x_j)^d$



รูปที่ 3.10 การทำงานของ Support Vector Regression (SVR) และ kernel function

การทำงานของSVRสามารถกำหนดหรือเปลี่ยนแปลงได้ตามชุดของข้อมูลมีความยืดหยุ่นในการใช้งานสูงและ สามารถปรับขนาดของเคอร์เนลฟังก์ชันในกระบวนการสร้างแบบจำลองเพื่อให้เหมาะสมกับชุดข้อมูลได้

ปัจจัยที่เลือก Support Vector Regression (SVR) เนื่องจากให้ความแม่นยำสูงและเป็นวิธีการที่มีความ รวดเร็วในการประมวลผล และ สามารถทำงานในการประมวลผลชุดข้อมูลแบบอนุกรมเวลาได้ดี

3.8 การนำโครงข่ายประสาทเทียม (Deep Learning)

การนำโครงข่ายประสาทเทียม คือ วิธีการเรียนรู้แบบอัตโนมัติด้วยการเลียนแบบการทำงานของโครงข่ายประสาทของมนุษย์ Deep Learning จะรับข้อมูลดิบเข้าทันที และทำการประมวลอัตโนมัติ เพื่อหาข้อมูลตัวอย่างที่จำเป็นในการทำนายผลลัพธ์ มีงานวิจัยได้สร้างแบบจำลองเพื่อการพยากรณ์ข้อมูลอนุกรมเวลาในรูปแบบฤดูกาลจำนวน 8 ชุดข้อมูลด้วยโครงข่ายประสาทเทียมโดยทดลองกับโมเดลโครงข่ายประสาทเทียมแบบต่าง ๆ แสดงให้เห็นว่าโครงข่ายประสาทเทียมมีประสิทธิภาพในการแก้ปัญหาการพยากรณ์อนุกรมเวลาได้เป็น อย่างดีในบทความนี้ได้เลือกอัลกอริทึมโครงข่ายประสาทเทียม 2 อัลกอริทึม ได้แก่ Recurrent Neural Network (RNN) และ Long Short-Term Memory (LSTM)

3.8.1 โครงข่ายประสาทเทียมแบบ Recurrent Neural Network (RNN) หรือที่เรียกว่าโครงข่ายประสาทที่เกิดซ้ำ และ Long Short-Term Memory (LSTM) สามารถอธิบายได้ตามบริบทและระดับของนามธรรมต่างๆ โดย RNN คือ โครงข่ายประสาทที่มีการเชื่อมต่อแบบวนซ้ำระหว่างหน่วยหรือเลเยอร์หนึ่ง ๆ หรือหลาย ๆ ชั้นในโครงข่าย ซึ่งชั้นเหล่านี้มีลักษณะเป็นหน่วยที่เชื่อมต่อแบบวนซ้ำ โดยโครงข่าย RNN อาจจะไม่ประกอบด้วยชั้นที่เกิดซ้ำเสมอไป ในส่วนของ LSTM เป็น

ประเภทหนึ่งของ RNN ที่มีการเชื่อมต่อแบบวนซ้ำ ซึ่งทำให้ LSTM เป็นโครงข่ายที่มีการจำระยะยาว (Long Short-Term Memory) โดยสามารถจำข้อมูลได้ดีในระยะยาว ทั้งนี้ โครงข่าย RNN ทุกประเภทสามารถแสดงเป็นกราฟที่มีการเชื่อมต่อแบบวนซ้ำระหว่างหน่วยต่าง ๆ ได้ตั้งแต่หนึ่งรายการขึ้นไป

Recurrent Neural Network (RNN) เหมาะสำหรับงานที่เกี่ยวข้องกับการเรียงลำดับข้อมูล เช่น การจัดลำดับคำหรือประโยค ในทางปฏิบัติ มักจะใช้ Long Short-Term Memory (LSTM) แทน RNN เพราะ LSTM มีประสิทธิภาพในการคำนวณสูงกว่า LSTM ถูกพัฒนาขึ้นมาเพื่อแก้ปัญหาที่ RNN ประสบอยู่ เช่น ปัญหาการไล่ระดับสีที่หายไป แต่ปัญหาของ LSTM คืออาจจะใช้เวลาในการประมวลผลมากขึ้นเมื่อมีชุดข้อมูลที่ใหญ่ขึ้น

มีงานวิจัยที่ใช้โมเดลที่สร้างจาก RNN, LSTM, และ New Gate Control Unit (NGCU) ซึ่งมีพื้นฐานมาจาก RNN ในการทำนายราคาทองคำในอนาคตจากชุดข้อมูลอนุกรมเวลา ผลการศึกษาเปรียบเทียบประสิทธิภาพพบว่า RNN, LSTM, และ NGCU มีค่า R^2 เท่ากับ 0.9736, 0.9872, และ 0.9231 ตามลำดับ และใช้เวลาในการประมวลผลเท่ากับ 6.1494 วินาที, 254.4748 วินาที และ 8.7322 วินาที ตามลำดับ (Alim et al., 2020; Wang et al., 2022)

3.8.2 Long Short-Term Memory (LSTM) เป็นเทคนิคที่ถูกพัฒนามาจาก Recurrent Neural Network (RNN) โดยมีหลักการทำงานเหมือนกัน RNN แต่สามารถบันทึกค่าที่ได้จากการคำนวณในแตะละโหนดก่อนหน้าไว้ และสามารถกลับมาดูเพื่อใช้ในการคำนวนหรือตรวจสอบได้โดย จุดเด่นของ5 Long Short-Term Memory (LSTM) คือมีฟังก์ชั่นประตู (GATE) ที่ควบคุมชุดข้อมูลที่จะเข้ามาในแต่ละโหนด Forget gate layer, Input gate layer และ Output gate layer, forget gate layer แสดงในสมาการ 3.16

$$ft = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \tag{3.16}$$

เมื่อ ft คือ Forget gate

σ คือ ฟังก์ชั่น sigmoid

W_f คือ ค่าน้ำหนักของ matrices

h_{t-1}คือ ค่า output ของ cell state ก่อนหน้า (ที่ timestamp t-1)

x_t คือ ค่า input ที่เข้ามาใน cell state ณ เวลk t และ b_f คือ ค่า bias

ผลลัพธ์ของ Forget gat อยู่ระหว่าง 0 และ 1 ถ้า 0 จะน า cell state ออกไป และถ้าผลลัพธ์เป็น 1 จะคงค่า cell state นี้ต่อไป

Recurrent Neural Network (RNN) มีหลักการทำงานโดยการนำผลลัพธ์จากการคำนวณของโหนดก่อนหน้ามาเป็นอินพุตสำหรับโหนดถัดไป ซึ่งแต่ละโหนดใน RNN จะมีข้อมูลเข้ามาสองส่วน คือ ข้อมูลอินพุตของโหนดนั้น ๆ และผลลัพธ์ (เอาต์พุต) จากโหนดก่อนหน้า ทั้งสองข้อมูลนี้จะถูกรวมกันและแยกออกเป็นผลลัพธ์ (เอาต์พุต) ของโหนดปัจจุบัน และจะถูกส่งไปเป็นอินพุตสำหรับโหนดถัดไป

เนื่องจาก Recurrent Neural Network (RNN) และ Long Short-Term Memory (LSTM) มีประสิทธิภาพในการทำนายผลโดยใช้ชุดข้อมูลอนุกรมเวลาและมีลักษณะการประมวลผลคล้ายกัน คือการคิดจากผลลัพธ์ของค่า ก่อนหน้าเพื่อทำนายผลลัพธ์ถัดไป ซึ่งความแม่นยำและระยะเวลาในการประมวลผลจะขึ้นอยู่กับปริมาณของชุดข้อมูล จึงเป็นหนึ่งปัจจัยที่เลือก 2 อัลกอริทึมนี้มาทำนายผลลัพธ์ของการพิมพ์สามมิติ

จากการศึกษาวิธีการทำงานของอัลกอริทึม พบว่าอัลกอริทึม Autoregressive (AR), Autoregressive Integrated Moving Average (ARIMA), XGBoost, Support Vector Regression (SVR), Long Short-Term Memory (LSTM) และ Recurrent Neural Network (RNN) มีความสามารถในการทำนายผลการวิเคราะห์การถดถอย (Regression Analysis) บนอนุกรมเวลา โดยประสิทธิภาพในการทำนาย (ทั้งในด้านความแม่นยำและความไวในการประมวลผล) สามารถเพิ่มหรือลดได้ ขึ้นอยู่กับชุดข้อมูลที่ใช้ในการสร้างโมเดล ทั้งนี้ อัลกอริทึมดังกล่าวมีศักยภาพในการทำงานได้ดีในชุดข้อมูลอนุกรมเวลาที่เก็บจากเทอร์โมคัปเปิล, เซ็นเซอร์การปล่อยคลื่นเสียง (Acoustic Emission Sensor) และเซ็นเซอร์วัดความเร่ง ซึ่งสามารถใช้ในการทำนายลักษณะผิวของชิ้นงานพิมพ์ก่อนเสร็จสิ้นกระบวนการพิมพ์

3.9 การวัดประสิทธิภาพของโมเดล

ในการศึกษานี้การทดสอบประสิทธิภาพของโมเดลถูกดำเนินการโดยการวัดความแม่นยำของการทำนาย โดยใช้ค่าความผิดพลาดเฉลี่ยกำลังสอง (Root Mean Square Error: RMSE) และ ค่าความคลาดเคลื่อนสัมบูรณ์เฉลี่ยเชิงเปอร์เซ็นต์ (Mean absolute percentage error: MAPE)

RMSE เป็นการคำนวณค่าความแตกต่างระหว่างค่าจริงและค่าที่ทำนายไว้ แล้วนำมาคำนวณค่าเฉลี่ยกำลังสองของความผิดพลาดเหล่านั้น การใช้ RMSE ทำให้สามารถประเมินได้ว่าโมเดลนั้นมีความแม่นยำเพียงใด โดยค่าของ RMSE จะบ่งชี้ว่าโมเดลสามารถทำนายได้ใกล้เคียงกับค่าจริงมากแค่ไหน หากค่า RMSE มีค่าน้อยหรือเข้าใกล้ศูนย์ จะบ่งชี้ว่าโมเดลมีความสามารถในการทำนายที่แม่นยำและใกล้เคียงกับค่าจริงสมการที่ใช้ในการคำนวณ RMSE สามารถแสดงได้ในสมการที่ 3.17

$$\boldsymbol{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad (3.17)$$

เมื่อ y_i คือ ค่าจริงของข้อมูล

$\hat{y}_i$ คือ ค่าที่โมเดลทำนาย

n คือ จำนวนข้อมูลทั้งหมด

ค่าของ RMSE ที่มีค่าน้อยจะหมายถึงโมเดลมีความแม่นยำสูงในการทำนาย และหากค่า RMSE มีค่าสูง แสดงว่าโมเดลนั้นอาจจะไม่แม่นยำหรือมีการทำนายที่ผิดพลาดมาก

MAPE เป็นตัวชี้วัดความแม่นยำของแบบจำลองพยากรณ์หรือการทำนายข้อมูล โดยใช้ค่าความคลาดเคลื่อนสัมบูรณ์ระหว่างค่าจริงกับค่าที่ทำนาย จากนั้นนำมาคำนวณเป็นเปอร์เซ็นต์สมการที่ใช้ในการคำนวณ MAPE สามารถแสดงได้ในสามาการที่ 3.18

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right| \times 100 \qquad (3.18)$$

เมื่อ A_i คือ ค่าจริงของข้อมูล

F_i คือ ค่าที่โมเดลทำนาย

n คือ จำนวนข้อมูลทั้งหมด

หลังจากการสร้างโมเดลทั้ง 90 โมเดลสำเร็จด้วยโปรแกรม Python ซึ่งเป็นเครื่องมือสำหรับสร้างและปรับแต่งโมเดลการเรียนรู้ของเครื่อง (Machine Learning Models) ขั้นตอนถัดไปคือการนำผลลัพธ์ที่ได้จากโมเดลทั้งหมดมาทำการประเมินและเปรียบเทียบประสิทธิภาพอย่างละเอียด โดยมีเป้าหมายเพื่อค้นหาโมเดลที่เหมาะสมที่สุดสำหรับการนำไปใช้ในการคาดการณ์ (Prediction) โดยมีเกณฑ์การวัดประสิทธิภาพหลักสองด้าน ได้แก่ ประสิทธิภาพด้านความแม่นยำการวัดความแม่นยำของโมเดล (Accuracy) เป็นการประเมินว่าโมเดลสามารถทำนายผลได้ถูกต้องตามข้อมูลจริง (Ground Truth) ได้ดีเพียงใด โดยค่าความแม่นยำนี้จะได้จากการเปรียบเทียบค่าที่โมเดลคาดการณ์กับค่าจริงในชุดข้อมูลทดสอบ (Test Data)วิธีการประเมินความแม่นยำค่าความคลาดเคลื่อน (Error) ระหว่างค่าที่โมเดลทำนายกับค่าจริงจะถูกนำมาคำนวณเป็นตัวชี้วัด RMSE และ MAPE โดยค่าที่ต่ำกว่าหมายถึงโมเดลสามารถทำนายได้ใกล้เคียงค่าจริงมากขึ้นเป้าหมายเพื่อระบุว่าโมเดลใดให้ผลลัพธ์ที่แม่นยำที่สุดสำหรับแต่ละประเภทของข้อมูล เช่น อุณหภูมิ (Thermocouple), การสั่นสะเทือน (Accelerometer) และสัญญาณเสียง (Acoustic Emission)

ประสิทธิภาพด้านความไวในการทำนายผล (Prediction Speed) เป็นอีกปัจจัยสำคัญ โดยวัดว่าโมเดลสามารถประมวลผลและให้ผลลัพธ์การคาดการณ์ได้รวดเร็วเพียงใดวิธีการประเมินความไวการวัดเวลาที่ใช้ในการประมวลผล (Inference Time) ต่อข้อมูลหนึ่งชุด เช่น เวลาที่ใช้ในการวิเคราะห์และทำนายผลจากข้อมูลที่ส่งเข้าโมเดล โดยมีหน่วยเป็นวินาที

การประเมินนี้ยังอาจรวมถึงการวัดความซับซ้อนของโมเดล (Model Complexity) ซึ่งโมเดลที่มีโครงสร้างซับซ้อนอาจใช้เวลาประมวลผลมากกว่าเป้าหมายเพื่อค้นหาโมเดลที่ให้ผลลัพธ์อย่างรวดเร็วที่สุดในขณะที่ยังคงความแม่นยำที่เหมาะสม

บทที่ 4
ผลดำเนินการวิจัยและการอภิปรายผล

โดยในงานวิจัยนี้ได้รวบรวมผลลัพธ์จากการฝึกโมเดลนำค่าความแม่นยำและความไวของโมเดลทั้ง 90 โมเดลมาจัดเรียงและวิเคราะห์ โดยในงานวิจัยนี้ ได้มีการทดสอบโมเดลการพยากรณ์ข้อมูลจากเซ็นเซอร์สามประเภท ได้แก่ Accelerometer ใน Frequency Domain, Acoustic Emission ใน Frequency Domain, และ Thermocouple ใน Time Domain โดยเปรียบเทียบโมเดลการพยากรณ์หกประเภท ได้แก่ ARIMA, Autoregression, XGBoost, SVR, LSTM และ RNN โดยใช้ข้อมูลในระดับ 20%, 40%, 60%, 80% และ 100% ของเวลาทั้งหมดเพื่อดูว่าอัลกอริทึมใดมีประสิทธิภาพสูงสุดวิเคราะห์ผลรวมทุกด้านโมเดลที่มีความแม่นยำสูงสุดอาจไม่ได้มีความไวในการประมวลผลสูงที่สุดเสมอไป ดังนั้นจึงต้องมีการประเมินว่ามีการประนีประนอม (Trade-off) ระหว่างความแม่นยำและความไวในระดับที่เหมาะสมหรือไม่การวิเคราะห์นี้จะช่วยให้สามารถเลือกโมเดลที่เหมาะสมที่สุดสำหรับแต่ละประเภทของข้อมูล และสำหรับงานวิจัยนี้ โมเดลที่ให้ทั้งความแม่นยำและความไวสูงสุดจะถูกพิจารณาเป็นตัวเลือกที่เหมาะสมที่สุดสำหรับการนำไปประยุกต์ใช้ในสถานการณ์จริง ผลลัพธ์ของอัลกอริทึมที่สร้างโดยชุดข้อมูลที่เก็บจาก Thermocouple แสดงในตารางที่ 4.1, ผลลัพธ์ของอัลกอริทึมที่สร้างโดยชุดข้อมูลที่เก็บจาก Accelerometer แสดงในตารางที่ 4.2 และผลลัพธ์ของอัลกอริทึมที่สร้างโดยชุดข้อมูลที่เก็บจาก Acoustic Emission แสดงในตารางที่ 4.3

4.1 ผลการทดลองโมเดลที่สร้างจาก Thermocouple

ตารางที่ 4.1 ตารางเปรียบเทียบประสิทธิภาพโมเดลที่สร้างโดยชุดข้อมูล Temperature

Temperature					
	20%	40%	60%	80%	100%
ARIMA RMSE (μm)	0.16	0.22	0.16	0.16	0.15
ARIMA MAPE (%)	8.49	8.48	8.71	8.96	8.71
ARIMA runtime (sec)	14.5	27.14	15.63	36.69	23.95
Autoregressive RMSE (μm)	0.16	0.16	0.16	0.15	0.15
Autoregression MAPE (%)	10.96	11.18	11.35	12.45	9.05

ตารางที่ 4.1 ตารางเปรียบเทียบประสิทธิภาพโมเดลที่สร้างโดยชุดข้อมูล Temperature (ต่อ)

Temperature					
	20%	40%	60%	80%	100%
Autoregressive runtime (sec)	0.49	0.9	1.31	1.75	2.11
XGBoost RMSE (μm)	6.58	6.63	6.71	6.65	6.69
XGBoost MAPE (%)	10.49	10.35	10.68	10.59	10.75
XGBoost runtime (sec)	0.22	0.36	0.52	0.67	0.80
SVR RMSE (μm)	7.13	7.05	7.13	7.29	7.90
SVR MAPE (%)	9.61	9.62	9.65	9.66	9.67
SVR runtime (sec)	110.14	2471.74	11478.4	74304.87	358023.4
LSTM RMSE (μm)	7.29	7.03	7.09	6.98	7.01
LSTM MAPE (%)	10.73	12.01	11.18	10.28	10.19
LSTM runtime (sec)	34.94	65.99	96.46	132.61	160.02
RNN RMSE (μm)	10.60	10.31	7.10	7.30	7.50
RNN MAPE (%)	9.96	10.07	11.91	10.51	9.95
RNN runtime (sec)	32.79	64.97	96.02	131.35	157.64

เซ็นเซอร์ Thermocouple ใช้สำหรับวัดอุณหภูมิ โดยให้ข้อมูลในรูปแบบของ Time Domain ซึ่งต่างจากเซ็นเซอร์ Accelerometer และ AE ที่อยู่ใน Frequency Domain ผลการทดลองพบว่าโมเดลที่ให้ค่าความผิดพลาดต่ำที่สุดคือ ARIMA โดยมี ค่า RMSE ต่ำมากที่ 0.16 μm ในทุกระดับของข้อมูล และค่า MAPE อยู่ที่ประมาณ 8.48% - 8.73% ซึ่งต่ำที่สุดเมื่อเทียบกับโมเดลอื่น ๆ นี่แสดงให้เห็นว่า ARIMA สามารถจัดการกับข้อมูลแบบ Time Series ได้ดี

โมเดล SVR และ XGBoost มีค่าความผิดพลาดสูงขึ้นมาก โดยค่า RMSE อยู่ที่ 7.05 μm - 7.13 μm และค่า MAPE สูงกว่าของ ARIMA อย่างไรก็ตาม Runtime ของ SVR ที่ระดับข้อมูล 100% ของเวลาทั้งหมด สูงถึง 35802 วินาที ซึ่งอาจเกิดจากข้อจำกัดของอัลกอริทึมที่ต้องใช้เวลาในการคำนวณสูง ในขณะที่ XGBoost มี runtime ที่ดีกว่ามาก และอาจเป็นตัวเลือกสำรองหากต้องการลดเวลาการประมวลผล

โมเดลเชิงลึกอย่าง LSTM และ RNN ให้ค่าความผิดพลาดใกล้เคียงกับ XGBoost และ SVR แต่ใช้เวลาในการประมวลผลสูงกว่า จึงไม่เป็นตัวเลือกที่เหมาะสมเมื่อเทียบกับ ARIMA

4.2 ผลการทดลองโมเดลที่สร้างจาก Accelerometer

ตารางที่ 4.2 ตารางเปรียบเทียบประสิทธิภาพโมเดลที่สร้างโดยชุดข้อมูล Accelerometer

Accelerometer					
	20%	40%	60%	80%	100%
ARIMA RMSE (μm)	9.18	9.26	9.34	10.48	9.30
ARIMA MAPE (%)	8.51	8.49	8.73	8.98	8.73
ARIMA runtime (sec)	2.30	9.63	12.42	18.44	11.09
Autoregressive RMSE (μm)	6.72	6.70	6.73	6.74	6.70
Autoregression MAPE (%)	11.01	11.27	11.40	12.56	9.10
Autoregressive runtime (sec)	0.28	1.69	2.24	3.25	1.20
XGBoost RMSE (μm)	6.35	6.36	6.40	6.46	6.42
XGBoost MAPE (%)	10.52	10.42	10.79	10.70	10.82
XGBoost runtime (sec)	0.12	0.66	0.95	1.59	0.47
SVR RMSE (μm)	6.69	6.66	6.70	6.71	6.68
SVR MAPE (%)	9.63	9.65	9.66	9.68	9.70
SVR runtime (sec)	0.15	0.85	1.61	2.81	1.49
LSTM RMSE (μm)	6.57	6.38	6.42	6.52	6.42
LSTM MAPE (%)	10.84	12.12	11.30	10.39	10.31
LSTM runtime (sec)	452.99	907.93	1474.54	1438.24	1380.43
RNN RMSE (μm)	6.34	6.37	6.41	6.47	6.43
RNN MAPE (%)	9.97	10.13	12.03	10.62	9.98
RNN runtime (sec)	385.71	806.74	1241.23	1012.70	972.65

เซ็นเซอร์ Accelerometer ทำหน้าที่วัดความเร่งในรูปของข้อมูลเชิงความถี่ ซึ่งมักใช้ในการวิเคราะห์แรงสั่นสะเทือนหรือการเปลี่ยนแปลงของการเคลื่อนที่ เมื่อวิเคราะห์ผลการทดลองพบว่าโมเดลที่ให้ค่าความผิดพลาดต่ำที่สุดคือ LSTM และ RNN ซึ่งให้ค่า RMSE ต่ำสุดอยู่ที่ 6.34 μm - 6.67 μm และค่า MAPE ต่ำกว่า 11% อย่างไรก็ตาม ข้อเสียหลักของโมเดลเหล่านี้คือใช้เวลาในการประมวลผลสูงมาก โดย LSTM ใช้เวลา 452 วินาทีที่ 20% ของเวลาทั้งหมด และเพิ่มขึ้นถึง 1380 วินาทีที่ 100% ของเวลาทั้งหมด ในขณะที่ RNN มี runtime อยู่ที่ 385.71 วินาทีที่ 20% และเพิ่มขึ้นเป็น 1241.23 วินาทีที่ 60% ก่อนที่จะลดลงเล็กน้อยที่ 100% ของเวลาทั้งหมด

ถึงแม้ว่าโมเดล LSTM และ RNN จะมีความแม่นยำสูง แต่ข้อเสียด้านเวลาการประมวลผลทำให้ XGBoost และ SVR เป็นตัวเลือกที่เหมาะสมกว่าในเชิงปฏิบัติ โดย XGBoost ให้ค่า RMSE อยู่ที่ 6.35 μm - 6.68 μm และใช้เวลาในการประมวลผลเพียง 0.12 วินาที ในขณะที่ SVR มีค่า RMSE ใกล้เคียงกันที่ 6.66 μm - 6.71 μm และ runtime อยู่ระหว่าง 0.15 - 1.49 วินาที ทำให้สามารถใช้งานได้จริงในสถานการณ์ที่ต้องการการพยากรณ์แบบเรียลไทม์

ในขณะที่ ARIMA และ Autoregression ซึ่งเป็นโมเดลที่ใช้พื้นฐานทางสถิติให้ผลลัพธ์ที่ด้อยกว่ามาก โดย ARIMA มีค่า RMSE อยู่ที่ 9.18 μm - 10.48 μm และค่า MAPE ที่ 8.49%- 8.73% แสดงให้เห็นว่าความแม่นยำไม่เพียงพอสำหรับการพยากรณ์จากเซ็นเซอร์ Accelerometer ขณะที่ Autoregression มีค่า RMSE อยู่ที่ประมาณ 6.7 μm ซึ่งแม้ว่าจะดีกว่า ARIMA แต่ยังด้อยกว่า XGBoost และ SVR อย่างชัดเจน

4.3 ผลการทดลองโมเดลที่สร้างจาก AE sensor

ตารางที่ 4.3 ตารางเปรียบเทียบประสิทธิภาพโมเดลที่สร้างโดยชุดข้อมูล Acoustic Emission Sensor

Acoustic Emission Sensor					
	20%	40%	60%	80%	100%
ARIMA RMSE (μm)	9.18	9.26	9.34	10.48	9.30
ARIMA MAPE (%)	8.457	8.481	8.691	8.923	8.682
ARIMA runtime (sec)	2.05	4.11	5.18	7.47	10.12
Autoregressive RMSE (μm)	6.72	6.70	6.73	6.74	6.70
Autoregression MAPE (%)	10.9827	11.1925	11.3206	12.4987	9.0725
Autoregressive runtime (sec)	0.22	0.45	0.64	0.90	1.12

ตารางที่ 4.3 ตารางเปรียบเทียบประสิทธิภาพโมเดลที่สร้างโดยชุดข้อมูล Acoustic Emission Sensor (ต่อ)

Acoustic Emission Sensor					
	20%	40%	60%	80%	100%
XGBoost RMSE (μm)	6.58	6.60	6.63	6.63	6.60
XGBoost MAPE (%)	10.4593	10.375	10.6432	10.5492	10.6975
XGBoost runtime (sec)	0.07	0.12	0.16	0.21	0.27
SVR RMSE (μm)	6.78	6.78	6.81	6.79	6.79
SVR MAPE (%)	9.598	9.612	9.635	9.65	9.668
SVR runtime (sec)	0.05	0.16	0.28	0.48	0.82
LSTM RMSE (μm)	6.58	6.60	6.63	6.63	6.60
LSTM MAPE (%)	10.7458	12.0321	11.1453	10.2458	10.1584
LSTM runtime (sec)	254.15	491.00	717.45	1011.69	1215.25
RNN RMSE (μm)	6.58	6.61	6.63	6.63	6.61
RNN MAPE (%)	9.9315	10.0956	11.8742	10.4871	9.9237
RNN runtime (sec)	207.55	411.35	599.88	836.77	1003.95

Acoustic Emission Sensor ใช้สำหรับตรวจจับเสียงที่เกิดจากการเสียดสีหรือรอยร้าวของวัสดุ ซึ่งเป็นข้อมูลที่อยู่ใน Frequency Domain เช่นเดียวกับ Accelerometer ผลการทดลองแสดงให้เห็นว่าแนวโน้มของโมเดลที่มีประสิทธิภาพดีที่สุดมีความคล้ายคลึงกับเซ็นเซอร์ Accelerometer

LSTM และ RNN ยังคงเป็นโมเดลที่ให้ค่าความผิดพลาดต่ำที่สุด โดยมี RMSE อยู่ที่ 6.58 - 6.61 μm และค่า MAPE ในช่วง 10.7% - 11.8% อย่างไรก็ตาม เช่นเดียวกับกรณีของ Accelerometer LSTM ใช้เวลาตั้งแต่ 254 วินาทีไปจนถึง 1584 วินาที ขณะที่ RNN ใช้เวลา 207 - 1003 วินาที ซึ่งเป็นเวลาที่สูงมาก ทำให้ไม่น่าเหมาะสมกับการใช้งานจริง

โมเดลที่ให้สมดุลที่ดีที่สุดระหว่างความแม่นยำและเวลาในการประมวลผลยังคงเป็น XGBoost และ SVR โดย XGBoost ให้ค่า RMSE อยู่ที่ 6.58 - 6.79 μm และใช้เวลาเพียง 0.07 - 0.27 วินาที ในขณะที่ SVR มีค่า RMSE ใกล้เคียงกัน และ runtime อยู่ระหว่าง 0.05 - 0.68 วินาที

ARIMA และ Autoregression ยังคงมีค่าความผิดพลาดสูง โดย RMSE ของ ARIMA อยู่ที่ 9.18 - 10.48 μm และของ Autoregression อยู่ที่ 6.7 μm ทำให้โมเดลเหล่านี้ไม่เหมาะสมสำหรับการพยากรณ์ข้อมูลจากเซ็นเซอร์ AE

จากการวิเคราะห์ประสิทธิภาพของเซ็นเซอร์ทั้งสามชนิด พบว่า เซ็นเซอร์ Thermocouple ที่ให้ข้อมูลใน Time Domain สามารถใช้โมเดล ARIMA ได้อย่างมีประสิทธิภาพมากที่สุด โดยให้ค่าความผิดพลาดต่ำสุดและใช้เวลาในการประมวลผลต่ำกว่าทางเลือกอื่น โมเดล Autoregression ก็ให้ผลลัพธ์ที่ใกล้เคียงกันในแง่ของความแม่นยำและเวลาในการประมวลผล แสดงให้เห็นว่าเซ็นเซอร์ประเภทนี้เหมาะกับโมเดลที่ออกแบบมาสำหรับข้อมูลที่มีแนวโน้มคาดเดาได้ง่าย เมื่อเปรียบเทียบเซ็นเซอร์เครื่องเร่งความเร็ว (Accelerometer) และเซ็นเซอร์การปล่อยเสียงอะคูสติก (AE Sensor) ซึ่งให้ข้อมูลใน Frequency Domain พบว่า Accelerometer มีประสิทธิภาพดีกว่า AE Sensor โดยเฉพาะเมื่อใช้โมเดล XGBoost และ SVR ที่ให้สมดุลระหว่างความแม่นยำและเวลาในการประมวลผลได้ดีที่สุด สำหรับ AE Sensor พบว่ามีแนวโน้มที่จะมีค่าความผิดพลาดสูงขึ้นเล็กน้อยเมื่อเทียบกับ Accelerometer ในโมเดลเดียวกัน

การเปรียบเทียบอัลกอริทึมทั้ง 6 ชนิด

- ARIMA และ Autoregressive เหมาะสำหรับข้อมูลที่มีแนวโน้มคาดเดาได้ง่าย เช่น ข้อมูลอุณหภูมิจาก Thermocouple แต่ไม่สามารถรับมือกับข้อมูลที่ซับซ้อนจาก Accelerometer และ AE Sensor ได้ดีนัก
- XGBoost เป็นโมเดลที่มีความสมดุลดีที่สุด เหมาะสำหรับข้อมูลที่ต้องการทั้งความแม่นยำและความเร็ว เช่น Accelerometer และ AE Sensor ที่อยู่ใน Frequency Domain
- LSTM และ RNN เหมาะสำหรับข้อมูลที่มีความซับซ้อนสูง และสามารถเรียนรู้รูปแบบข้อมูลที่ไม่เป็นเชิงเส้นได้ดี อย่างไรก็ตาม ข้อเสียหลักคือเวลาในการประมวลผลที่สูงมาก ทำให้ไม่เหมาะสำหรับการใช้งานที่ต้องการความรวดเร็ว
- SVR มีจุดเด่นที่เวลาในการประมวลผลต่ำ แต่มีข้อจำกัดในด้านความแม่นยำ เมื่อเทียบกับ XGBoost และ LSTM ในการพยากรณ์ข้อมูลจาก Accelerometer และ AE Sensor

โดยสรุป Thermocouple เหมาะกับโมเดล ARIMA หรือ Autoregression, Accelerometer เหมาะกับ XGBoost และ SVR, ขณะที่ AE Sensor มีแนวโน้มให้ผลลัพธ์แย่กว่า Accelerometer เล็กน้อย แต่ยังสามารถใช้ XGBoost หรือ LSTM ได้ในกรณีที่ต้องการความแม่นยำสูงสุดแม้ต้องแลกกับเวลาการประมวลผลที่เพิ่มขึ้น

XGBoost เป็นโมเดลที่ดีที่สุดในหลายกรณี โดยเฉพาะงานที่ต้องการความรวดเร็วและความแม่นยำในระดับที่ยอมรับได้ สามารถใช้ได้กับ AE Sensor และ Accelerometer

LSTM และ RNN เหมาะสำหรับงานที่มีความซับซ้อนสูง หรือกรณีที่ต้องการโมเดลที่สามารถเรียนรู้รูปแบบข้อมูลเชิงลึก

Linear SVR มีจุดเด่นที่เวลาในการประมวลผลต่ำ แต่มีข้อจำกัดด้านความแม่นยำ

จากผลการทดลอง โดยพิจารณาประสิทธิภาพของโมเดลที่ระดับ 20% ของเวลาทั้งหมด ซึ่งเป็นระดับที่ให้สมดุลที่เหมาะสมระหว่างความแม่นยำและเวลาในการประมวลผล

บทที่ 5
สรุปและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การศึกษานี้ได้นำเสนอแนวทางใหม่ในการพัฒนาอัลกอริทึมที่มีเป้าหมายในการทำนายลักษณะผิวของชิ้นงานที่พิมพ์ 3 มิติ ก่อนที่กระบวนการพิมพ์จะเสร็จสิ้น โดยมีวัตถุประสงค์เพื่อลดการสูญเสียในกระบวนการผลิตด้วยเทคโนโลยีการพิมพ์แบบ Fused Deposition Modeling (FDM) อัลกอริทึมนี้พัฒนาขึ้นโดยใช้ชุดข้อมูลที่รวมพารามิเตอร์ต่าง ๆ และรูปร่างของชิ้นงานที่เก็บจากเครื่องพิมพ์ FDM โมเดลที่ทำนายได้ถูกสร้างขึ้นโดยใช้ข้อมูลจากเซ็นเซอร์สามประเภท ได้แก่ เทอโมคอปเปอร์ เซ็นเซอร์เครื่องเร่งความเร็ว (Accelerometer) และเซ็นเซอร์การปล่อยเสียงอะคูสติก (Acoustic Emission Sensor) โดยใช้เทคนิคทั้งหมดหกอัลกอริทึมในการสร้างโมเดลเหล่านี้ ได้แก่ อ อ Autoregressive: (AR), Autoregressive Integrated Moving Average (ARIMA), XGBoost, Support Vector Regression (SVR), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN) โดยชุดข้อมูลถูกทดสอบในห้าระดับ ได้แก่ร้อยละ 20, 40, 60, 80, และ 100 ของเวลาทั้งหมด

ในการพัฒนาโมเดลทั้งหมด 90 แบบซึ่งการเลือกโมเดลที่เหมาะสมสำหรับการพยากรณ์ข้อมูลจากเซ็นเซอร์แต่ละประเภทนั้นจำเป็นต้องพิจารณาทั้งด้านความแม่นยำของผลลัพธ์และเวลาในการประมวลผล เพื่อให้สามารถนำไปใช้งานได้อย่างมีประสิทธิภาพผล โดยภายในงานวิจัยนี้ได้ใช้ตัวชี้วัดได้แก่ ค่า Root Mean Square Error (RMSE), Mean absolute percentage error(MAPE) และเวลาการประมวลผล (Runtime) โดยผลการทดลองพบว่าโมเดลที่เหมาะสมที่สุดสำหรับแต่ละเซ็นเซอร์มีดังนี้

Thermocouple ที่ 20% ของเวลาทั้งหมดโมเดล ARIMA ให้ค่าความผิดพลาดต่ำที่สุดอยู่ที่ RMSE ± 0.16 μm, MAPE = 8.49% และใช้เวลาในการประมวลผลเพียง 2.30 วินาที ซึ่งต่ำกว่าโมเดลอื่นอย่างชัดเจน โมเดลนี้สามารถเรียนรู้แนวโน้มของข้อมูลอุณหภูมิได้ดีและให้ผลลัพธ์ที่แม่นยำโดยไม่ต้องใช้เวลาประมวลผลสูง

Accelerometerที่ 20% ของเวลาทั้งหมดโมเดล XGBoost ให้ค่า RMSE อยู่ที่ ± 6.35 μm, MAPE 10.82% และใช้เวลาในการประมวลผลเพียง 0.12 วินาที ทำให้เป็นตัวเลือกที่ดีที่สุดสำหรับเซ็นเซอร์ประเภทนี้ โดยเฉพาะสำหรับการใช้งานที่ต้องการความเร็ว

Acoustic Emission เช่นเดียวกับ Accelerometer เซ็นเซอร์ AE ให้ข้อมูลในรูปของ Frequency Domain ซึ่ง XGBoost สามารถพยากรณ์ได้อย่างมีประสิทธิภาพสูงสุด ทั้งในแง่ของ

ความแม่นยำและระยะเวลาในการประมวลผล โดยที่ RMSE อยู่ที่ ±6.58 μm, MAPE 10.45% และใช้เวลาในการประมวลผลเพียง 0.07 วินาที

ซึ่งภายในงานวิจัยนี้ยังชี้ให้เห็นความเป็นไปได้ในการทำความหยาบของผิวของชิ้นงานพิมพ์ที่ผลิตด้วยเทคโนโลยี FDM ได้อย่างแม่นยำ นอกจากนี้ ความยืดหยุ่นของโมเดลยังช่วยให้สามารถทำนายลักษณะผิวของชิ้นส่วนได้ก่อนที่กระบวนการพิมพ์จะเสร็จสิ้น ซึ่งเป็นแนวทางที่มีแนวโน้มดีในการลดการสูญเสียในการผลิตพิมพ์ 3 มิติ

5.2 ข้อเสนอแนะ

5.2.1 การเพิ่มประสิทธิภาพของโมเดล

จากผลการวิจัยพบว่าโมเดลที่สร้างขึ้นโดยใช้ข้อมูลเพียง 20% ของระยะเวลาการผลิตทั้งหมดมีความเหมาะสมสำหรับการฝึกโมเดล โดยข้อมูลที่ได้จากเซ็นเซอร์ Thermocouple พบว่าอัลกอริทึมที่ให้ประสิทธิภาพดีที่สุดในการคาดการณ์คือ Autoregressive (AR) และ ARIMA ทั้งสองอัลกอริทึมสามารถจัดการกับข้อมูลอนุกรมเวลาได้อย่างมีประสิทธิภาพและเหมาะสมสำหรับการพยากรณ์อุณหภูมิในกระบวนการพิมพ์สามมิติ สำหรับเซ็นเซอร์ Accelerometer และ AE Sensor ซึ่งใช้วัดการสั่นสะเทือนและสัญญาณเสียงตามลำดับ อัลกอริทึมที่เหมาะสมที่สุดคือ XGBoost เนื่องจากมีความสามารถสูงในการวิเคราะห์ข้อมูลที่ซับซ้อนและจัดการกับชุดข้อมูลที่หลากหลายได้ดี

นอกจากนี้ วิธีสำคัญที่จะช่วยเพิ่มประสิทธิภาพของโมเดลได้อย่างมากคือการเพิ่มจำนวนตัวอย่างข้อมูล (Data Samples) โดยการรวบรวมข้อมูลจากชิ้นงานที่หลากหลายและครอบคลุมสถานการณ์ที่แตกต่างกัน เช่น การปรับเปลี่ยนพารามิเตอร์การผลิต หรือการจำลองความผิดพลาดที่อาจเกิดขึ้นในกระบวนการผลิต การเพิ่มความหลากหลายของข้อมูลจะช่วยให้โมเดลสามารถเรียนรู้ได้อย่างลึกซึ้งและสามารถคาดการณ์ผลได้อย่างแม่นยำมากยิ่งขึ้นอย่างไรก็ตามการปรับลดขนาดชุดข้อมูลต้องดำเนินการด้วยความระมัดระวัง เพื่อไม่ให้เสียความหลากหลายของข้อมูลที่จำเป็นต่อการเรียนรู้ของโมเดล การเลือกข้อมูลที่มีความสำคัญและเกี่ยวข้องกับพฤติกรรมการทำงานของเซ็นเซอร์ เช่น ข้อมูลช่วงเวลาที่เกิดความเปลี่ยนแปลงที่สำคัญในกระบวนการผลิต จะช่วยให้โมเดลสามารถทำนายผลได้อย่างรวดเร็วขึ้นโดยยังคงความแม่นยำและประสิทธิภาพของการทำงาน

การผสมผสานโมเดล สามารถพัฒนาโมเดลแบบไฮบริดที่รวมข้อดีของโมเดลต่าง ๆ เช่น ARIMA-XGBoost หรือ ARIMA-LSTM เพื่อให้สามารถพยากรณ์ข้อมูลที่มีแนวโน้มเชิงเส้นและไม่เป็นเชิงเส้นได้ดียิ่งขึ้นโมเดลไฮบริดอาจช่วยให้สามารถรับมือกับความซับซ้อนของข้อมูลที่มีแนวโน้มเปลี่ยนแปลงได้ตามเงื่อนไขของกระบวนการผลิต

ในด้านการประเมินผลโมเดล การใช้ Cross Validation หรือ K-Fold Validation เป็นเครื่องมือสำคัญที่ช่วยลดปัญหา Overfitting และเพิ่มความน่าเชื่อถือของโมเดล โดย Standard K-Fold เหมาะกับข้อมูลที่มีการกระจายตัวสม่ำเสมอ เช่น Accelerometer ขณะที่ Stratified K-Fold

เหมาะกับข้อมูลที่มีความไม่สมดุล และ Time Series Split (Rolling Window Validation) เป็นทางเลือกที่ดีที่สุดสำหรับข้อมูลที่มีลำดับเวลา เช่น Thermocouple การเลือกใช้เทคนิคที่เหมาะสมช่วยให้โมเดลสามารถเรียนรู้ข้อมูลได้ดีขึ้นและสามารถนำไปใช้งานจริงได้อย่างมีประสิทธิภาพ

5.2.2 การลดเวลาในการทำนายผลของโมเดล

ผลการศึกษาชี้ให้เห็นว่าการใช้ข้อมูลที่ได้จากเพียง 20% ของระยะเวลาการผลิตทั้งหมดเพียงพอสำหรับการฝึกโมเดล ทั้งนี้ยังมีแนวโน้มว่าสามารถลดขนาดชุดข้อมูลที่ใช้ในการฝึกฝนให้เล็กลงกว่านี้ได้อีก โดยไม่ส่งผลต่อความแม่นยำของโมเดล การลดขนาดของชุดข้อมูลมีข้อดีคือช่วยลดเวลาในการประมวลผลและการทำนายผลได้อย่างมีประสิทธิภาพมากขึ้น Standard K-Fold Cross Validation เหมาะสำหรับชุดข้อมูลที่มีการกระจายตัวอย่างสม่ำเสมอและไม่ได้ขึ้นอยู่กับลำดับของเวลา เช่น ข้อมูลที่มาจากเซ็นเซอร์ Accelerometer ซึ่งสามารถแบ่งข้อมูลออกเป็นหลายส่วนโดยไม่จำเป็นต้องคำนึงถึงลำดับก่อนหลังของข้อมูล

5.2.3 การตรวจสอบ Frequency Domain

การนำ Fourier Series มาใช้ในการวิเคราะห์และพยากรณ์ข้อมูลจากเซ็นเซอร์ เช่น Accelerometer และ AE Sensor ซึ่งอยู่ใน Frequency Domain จำเป็นต้องมีการตรวจสอบความแม่นยำของข้อมูลที่ได้รับหลังจากการแปลงกลับ (Inverse Fourier Transform, IFT) เพื่อให้แน่ใจว่าโมเดลสามารถรักษาคุณลักษณะสำคัญของข้อมูลต้นฉบับได้อย่างถูกต้อง กระบวนการนี้สามารถทำได้โดย เปรียบเทียบข้อมูลก่อนและหลังการแปลง, ปรับจำนวนพจน์ของ Fourier ให้เหมาะสม, ใช้เทคนิคการกรองสัญญาณรบกวน และวิเคราะห์ผลลัพธ์ทางสถิติ ซึ่งจะช่วยให้การใช้ Fourier Series มีความถูกต้องและมีประสิทธิภาพมากขึ้นในการพยากรณ์และวิเคราะห์ข้อมูลเซ็นเซอร์

5.2.4 การปรับปรุงอัลกอริทึมการทำนายความหยาบผิวของชิ้นงานพิมพ์สามมิติ

จากการศึกษาประสิทธิภาพของอัลกอริทึมทั้งหกประเภท ได้แก่ Autoregressive (AR), Autoregressive Integrated Moving Average (ARIMA), Extreme Gradient Boosting (XGBoost), Support Vector Regression (SVR), Long Short-Term Memory (LSTM) และ Recurrent Neural Network (RNN) ในการพยากรณ์ความหยาบผิวของชิ้นงานพิมพ์สามมิติ พบว่ามีจุดแข็งและข้อจำกัดที่แตกต่างกัน ซึ่งสามารถนำไปพัฒนาและปรับปรุงเพื่อเพิ่มความแม่นยำและลดเวลาในการประมวลผลของโมเดลได้ ดังนี้

การปรับปรุงอัลกอริทึม AR และ ARIMA ข้อดีของโมเดลคือเหมาะสำหรับข้อมูลอนุกรมเวลาที่มีแนวโน้มชัดเจนและสามารถพยากรณ์ค่าความหยาบผิวได้อย่างแม่นยำเมื่อใช้กับข้อมูลอุณหภูมิจาก Thermocouple และ มีข้อจำกัดคือไม่สามารถจัดการกับข้อมูลที่มีความซับซ้อนสูง เช่น ความสัมพันธ์ระหว่างแรงสั่นสะเทือนของเครื่องพิมพ์และเสียงที่เกิดขึ้นระหว่างกระบวนการพิมพ์ ข้อเสนอแนะในการปรับปรุง ได้แก่ ทดลองใช้ Hybrid Model โดยรวม ARIMA กับอัลกอริทึม Machine Learning เช่น XGBoost หรือ SVR เพื่อเพิ่มความแม่นยำ, ใช้ Feature Engineering โดย

แยกองค์ประกอบแนวโน้มและความผันผวนของข้อมูลออกมาเพื่อช่วยให้โมเดลเรียนรู้ได้ดีขึ้น และ ใช้ Grid Search หรือ Bayesian Optimization ในการปรับค่าพารามิเตอร์ (p, d, q) ของ ARIMA ให้เหมาะสมกับข้อมูลที่ใช้

การปรับปรุงอัลกอริทึม XGBoost ข้อดีของโมเดลคือแสดงประสิทธิภาพสูงสุดเมื่อใช้กับข้อมูลจาก Accelerometer และ Acoustic Emission Sensor โดยสามารถจับความสัมพันธ์เชิงซับซ้อนของข้อมูลได้ดี และ ข้อจำกัดคืออาจเกิด Overfitting ได้หากพารามิเตอร์ไม่ถูกปรับแต่งอย่างเหมาะสม และไม่สามารถจัดการกับข้อมูลอนุกรมเวลาได้โดยตรง ข้อเสนอแนะในการปรับปรุง ได้แก่ ใช้ Time Series Cross Validation (TSCV) แทนการแบ่งชุดข้อมูลแบบสุ่ม เพื่อลดปัญหาความคลาดเคลื่อนของโมเดล, เพิ่ม Feature Engineering โดยใช้ Fourier Transform หรือ Wavelet Transform เพื่อแปลงข้อมูลอนุกรมเวลาให้อยู่ในรูปแบบที่เหมาะกับการวิเคราะห์ของ XGBoost และ ปรับพารามิเตอร์หลัก เช่น learning rate, max_depth, และ number of estimators โดยใช้ Grid Search เพื่อเพิ่มความแม่นยำและลด Overfitting

การปรับปรุงอัลกอริทึม SVR ข้อดีของโมเดลคือเหมาะกับข้อมูลที่มีความซับซ้อนสูงและสามารถใช้ Kernel Trick เพื่อสร้างเส้นการทำนายที่แม่นยำ และ มีข้อจำกัดในด้านการใช้เวลาในการประมวลผลสูงมาก โดยเฉพาะเมื่อจำนวนข้อมูลเพิ่มขึ้น และต้องการการปรับแต่ง Kernel Function อย่างเหมาะสม ข้อเสนอแนะในการปรับปรุง ทดลองใช้ Non-linear Kernel Functions เช่น RBF หรือ Polynomial Kernel เพื่อปรับให้เข้ากับข้อมูลจากเซ็นเซอร์, ลดมิติของข้อมูลโดยใช้ Principal Component Analysis (PCA) เพื่อลดระยะเวลาในการประมวลผล และ เปรียบเทียบ SVR กับโมเดล XGBoost และ LSTM เพื่อเลือกอัลกอริทึมที่ให้ผลลัพธ์ที่ดีที่สุดสำหรับแต่ละประเภทของเซ็นเซอร์

การปรับปรุงอัลกอริทึม LSTM ข้อดีของโมเดลคือสามารถเรียนรู้ลำดับของข้อมูลอนุกรมเวลาได้ดี และเหมาะกับข้อมูลที่มีความสัมพันธ์ระยะยาว และมีข้อจำกัดในด้านการใช้เวลาฝึกโมเดลนานและต้องการข้อมูลจำนวนมากเพื่อให้สามารถเรียนรู้ได้อย่างมีประสิทธิภาพ ข้อเสนอแนะในการปรับปรุงใช้ Attention Mechanism เพื่อให้โมเดลสามารถโฟกัสไปที่ช่วงเวลาที่สำคัญที่สุดของข้อมูล, ใช้ Dropout และ Batch Normalization เพื่อลด Overfitting และเพิ่มความเสถียรของโมเดล และทดลองใช้ Bidirectional LSTM (Bi-LSTM) ซึ่งสามารถเรียนรู้ข้อมูลได้ทั้งจากอดีตและอนาคต เพื่อเพิ่มความแม่นยำของการทำนาย

การปรับปรุงอัลกอริทึม RNN ข้อดีของโมเดลคือสามารถเรียนรู้ลำดับของข้อมูลได้ และมีโครงสร้างที่ง่ายกว่ารุ่นอื่น เช่น LSTM และ ข้อจำกัดคือมีปัญหา Vanishing Gradient ทำให้ไม่สามารถจดจำข้อมูลระยะยาวได้ดี ข้อเสนอแนะในการปรับปรุงคือ เปลี่ยนไปใช้ Gated Recurrent Unit (GRU) ซึ่งเป็นเวอร์ชันที่ปรับปรุงของ RNN ที่สามารถเรียนรู้ข้อมูลระยะยาวได้ดีขึ้น และใช้ Dropout และ L2 Regularization เพื่อลด Overfitting

รายการอ้างอิง

Alim, M., Ye, G. H., Guan, P., Huang, D. S., Zhou, B. S., & Wu, W. (2020). Comparison of ARIMA model and XGBoost model for prediction of human brucellosis in mainland China: a time-series study. *BMJ open, 10*(12), e039676. (pp. 155–161).

Asadi-Eydivand, M., Solati-Hashjin, M., Farzad, A., & Osman, N. A. A. (2016). Effect of technical parameters on porous structure and strength of 3D printed calcium sulfate prototypes. *Robotics and Computer-Integrated Manufacturing*, 37, (pp. 57-67).

Asiltürk, İ., Kuntoğlu, M., Binali, R., Akkuş, H., & Salur, E. (2023). A comprehensive analysis of surface roughness, vibration, and acoustic emissions based on machine learning during hard turning of AISI 4140 *steel. Metals*, 13(2), 437.

Ayrilmis, N. (2018). Effect of layer thickness on surface properties of 3D printed materials produced from wood flour/PLA filament. *Polymer testing*, 71, (pp. 163-166).

Barrios, J. M., & Romero, P. E. (2019). Decision tree methods for predicting surface roughness in fused deposition modeling parts. *Materials*, 12(16), 2574.

Byun, H. S., & Lee, K. H. (2006). Determination of the optimal build direction for different rapid prototyping processes using multi-criterion decision making. *Robotics and Computer-Integrated Manufacturing*, 22(1), (pp. 69-80).

Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3), (pp. 1-27).

Costa, S. F., Duarte, F. M., & Covas, J. A. (2017). Estimation of filament temperature and adhesion development in fused deposition techniques. *Journal of Materials Processing Technology*, 245, (pp. 167-179).

D.L. Bourell, M.C. Leu, D.W. Rosen. (2009). Roadmap for Additive Manufacturing: Identifying the Future of Freeform Processing. *The University of Texas at Austin, Austin, TX*. (pp. 11–15).

Everton, S. K., Hirsch, M., Stravroulakis, P., Leach, R. K. , & Clare, A. T. (2016). Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. *Materials & Design*, 95, (pp. 431-445).

Galantucci, L. M., Lavecchia, F., & Percoco, G. (2009). Experimental study aiming to enhance the surface finish of fused deposition modeled parts. *CIRP annals*, 58(1), (pp. 189-192).

Geng, P., Zhao, J., Wu, W., Ye, W., Wang, Y., Wang, S., & Zhang, S. (2019). Effects of extrusion speed and printing speed on the 3D printing stability of extruded PEEK filament. *Journal of Manufacturing Processes*, 37, (pp. 266-273).

Gibson, I., Rosen, D., Stucker, B., Khorasani, M., Rosen, D., Stucker, B., & Khorasani, M. (2021). *Additive manufacturing technologies* (Vol. 17 , pp. 16 0-186). Cham, Switzerland: Springer.

Guin, A. (2006 , September). Travel time prediction using a seasonal autoregressive integrated moving average time series model. In 2006 *IEEE intelligent transportation systems conference*. (pp. 493-498). IEEE.

Li, Z., Zhang, Z., Shi, J., & Wu, D. (2019). Prediction of surface roughness in extrusion-based additive manufacturing with machine learning. *Robotics and Computer-Integrated Manufacturing*, 57, (pp. 488-495).

Nam, J., Jo, N., Kim, J. S., & Lee, S. W. (2020). Development of a health monitoring and diagnosis framework for fused deposition modeling process based on a machine learning algorithm. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 234(1-2), (pp. 324-332). doi: 10.1177/0954405419855224

Reutzel, E. W., & Nassar, A. R. (2015). A survey of sensing and control systems for machine and process monitoring of directed-energy, metal-based additive manufacturing. *Rapid Prototyping Journal*, 21(2), (pp. 159-167).

Sahoo, B. B., Jha, R., Singh, A., & Kumar, D. (2019). Application of support vector regression for modeling low flow time series. *KSCE Journal of Civil Engineering*, 23, (pp. 923-934).

Singh, V., Pencina, M., Einstein, A. J., Liang, J. X., Berman, D. S., & Slomka, P. (2021). Impact of train/test sample regimen on performance estimate stability of machine learning in cardiovascular imaging. *Scientific reports*, 11(1), 14490.

Strano, G., Hao, L., Everson, R. M., & Evans, K. E. (2013). Surface roughness analysis, modelling and prediction in selective laser melting. *Journal of Materials Processing Technology*, 213(4), (pp. 589-597).

Tapia, G., & Elwany, A. (2014). A review on process monitoring and control in metal-based additive manufacturing. *Journal of Manufacturing Science and Engineering*, 136(6), 060801.

Tlegenov, Y., Hong, G. S., & Lu, W. F. (2018). Nozzle condition monitoring in 3D printing. *Robotics and Computer-Integrated Manufacturing*, 54, (pp. 45-55).

Turner, B. N., & Gold, S. A. (2015). A review of melt extrusion additive manufacturing processes: II. Materials, dimensional accuracy, and surface roughness. *Rapid Prototyping Journal*, 21(3), (pp. 250-261).

Wang, J., Li, X., Li, J., Sun, Q., & Wang, H. (2022). NGCU: A new RNN model for time-series data prediction. *Big Data Research*, 27, 100296.

Wang, J. Y., Xu, D. D., Sun, W., Du, S. M., Guo, J. J., & Xu, G. J. (2019, February). Effects of nozzle-bed distance on the surface quality and mechanical properties of fused filament fabrication parts. In *IOP Conference Series: Materials Science and Engineering* (Vol. 479, No. 1, p. 012094). IOP Publishing. , doi:10.1088/1757-899X/479/1/012094

Wu, D., Wei, Y., & Terpenny, J. (2019). Predictive modelling of surface roughness in fused deposition modelling using data fusion. *International Journal of Production Research*, 57(12), 3992-4006.

Yoon, J., He, D., & Van Hecke, B. (2014, September). A PHM approach to additive manufacturing equipment health monitoring, fault diagnosis, and quality control. *In Annual Conference of the PHM Society* (Vol. 6, No. 1).

Zhang, X., Zhou, B., Zeng, Y., & Gu, P. (2002). Model layout optimization for solid ground curing rapid prototyping processes. *Robotics and Computer-Integrated Manufacturing*, 18(1), (pp. 41-51).

ภาคผนวก ก

Code ที่ใช้ในการดำเนินงาน

Code ที่ใช้สำหรับtrain Temperature data

Import Library

```
import pandas as pd
import numpy as np
import time
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.ar_model import AutoReg
from xgboost import XGBRegressor
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, SimpleRNN, Dense
from tensorflow.keras.optimizers import Adam
```

Data Loading

```
df20 = pd.read_csv("/content/20_.csv")
df40 = pd.read_csv("/content/40_.csv")
df60 = pd.read_csv("/content/60_.csv")
df80 = pd.read_csv("/content/80_.csv")
df100 = pd.read_csv("/content/100_.csv")
print("Shape of df20:", df20.shape)
print("Shape of df40:", df40.shape)
print("Shape of df60:", df60.shape)
print("Shape of df80:", df80.shape)
print("Shape of df100:", df100.shape)
```

Data Preprocessing

```
df20 = df20.iloc[:120000]
df40 = df40.iloc[:250000]
df60 = df60.iloc[:370000]
df80 = df80.iloc[:510000]
```

```
df100 = df100.iloc[:620000]
print("New shape of df20:", df20.shape)
print("New shape of df40:", df40.shape)
print("New shape of df60:", df60.shape)
print("New shape of df80:", df80.shape)
print("New shape of df100:", df100.shape)
def rename_columns(df):
    df.rename(columns={df.columns[0]: "Tem", df.columns[1]: "Surface"}, inplace=True)
rename_columns(df20)
rename_columns(df40)
rename_columns(df60)
rename_columns(df80)
rename_columns(df100)
for df in [df20, df40, df60, df80, df100]:
df['Surface'] = df['Surface'].astype(str).str.replace(',', '').astype(float)
def split_and_drop(df, test_size=0.3):
    train, test = train_test_split(df, test_size=test_size, random_state=42)
    train = df.loc[df.index.difference(test.index)]
    return train, test
train20, test20 = split_and_drop(df20)
train40, test40 = split_and_drop(df40)
train60, test60 = split_and_drop(df60)
train80, test80 = split_and_drop(df80)
train100, test100 = split_and_drop(df100)
print("Train and test shapes:")
print("df20 - Train:", train20.shape, "Test:", test20.shape)
print("df40 - Train:", train40.shape, "Test:", test40.shape)
print("df60 - Train:", train60.shape, "Test:", test60.shape)
print("df80 - Train:", train80.shape, "Test:", test80.shape)
print("df100 - Train:", train100.shape, "Test:", test100.shape)
dataframes = {'df20': df20, 'df40': df40, 'df60': df60, 'df80': df80, 'df100': df100}
for name, df in dataframes.items():
```

```
    max_value = df["Tem"].max()
    print(f"Max value of 'Tem' in {name}: {max_value}")
df100
x = df60.iloc[:, 0].values
y = df60.iloc[:, 1].values
import numpy as np
# Assuming y is your array
greater_than_30000 = y[y > 250]
print(greater_than_30000)
greater_than_30000.shape
Model
from sklearn.preprocessing import MinMaxScaler
import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.ar_model import AutoReg
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, SimpleRNN, Dense
from tensorflow.keras.optimizers import Adam
from xgboost import XGBRegressor
from sklearn.svm import SVR
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
def train_and_evaluate(df, df_name):
    # Extract input and target values
    x = df.iloc[:, 0].values
    y = df.iloc[:, 1].values
    # Drop rows where y > 250
    mask = y <= 250
    x = x[mask]
```

```
    y = y[mask]
    # Train-test split (70:30)
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)
    # Scaling y for models sensitive to magnitude differences
    y_scaler = MinMaxScaler()
    y_train_scaled = y_scaler.fit_transform(y_train.reshape(-1, 1)).flatten()
    y_test_scaled = y_scaler.transform(y_test.reshape(-1, 1)).flatten()
    # Store results
    results = {}
    # ARIMA Model
    start_time = time.time()
    arima_model = ARIMA(y_train_scaled, order=(5, 1, 0))
    arima_model_fit = arima_model.fit()
    arima_predictions = arima_model_fit.forecast(steps=len(y_test_scaled))
    arima_rmse = np.sqrt(mean_squared_error(y_test_scaled, arima_predictions))
    results['ARIMA'] = {'RMSE': arima_rmse, 'Runtime': time.time() - start_time}
    # Autoregressive Model
    start_time = time.time()
    ar_model = AutoReg(y_train_scaled, lags=5)
    ar_model_fit = ar_model.fit()
    ar_predictions = ar_model_fit.predict(start=len(y_train_scaled),
end=len(y_train_scaled) + len(y_test_scaled) - 1)
    ar_rmse = np.sqrt(mean_squared_error(y_test_scaled, ar_predictions))
    results['Autoregressive'] = {'RMSE': ar_rmse, 'Runtime': time.time() - start_time}
    # LSTM Model
    start_time = time.time()
    x_train_lstm = np.expand_dims(x_train, axis=1)
    x_test_lstm = np.expand_dims(x_test, axis=1)
    lstm_model = Sequential([
        LSTM(50, activation='relu', input_shape=(1, 1)),
        Dense(1)
    ])
```

```
    lstm_model.compile(optimizer=Adam(learning_rate=0.001), loss='mse')
    lstm_model.fit(x_train_lstm, y_train_scaled, epochs=10, batch_size=32, verbose=0)
    lstm_predictions = y_scaler.inverse_transform(lstm_model.predict(x_test_lstm))
    lstm_rmse = np.sqrt(mean_squared_error(y_test, lstm_predictions))
    results['LSTM'] = {'RMSE': lstm_rmse, 'Runtime': time.time() - start_time}
    # RNN Model
    start_time = time.time()
    rnn_model = Sequential([
        SimpleRNN(50, activation='relu', input_shape=(1, 1)),
        Dense(1)
    ])
    rnn_model.compile(optimizer=Adam(learning_rate=0.001), loss='mse')
    rnn_model.fit(x_train_lstm, y_train_scaled, epochs=10, batch_size=32, verbose=0)
    rnn_predictions = y_scaler.inverse_transform(rnn_model.predict(x_test_lstm))
    rnn_rmse = np.sqrt(mean_squared_error(y_test, rnn_predictions))
    results['RNN'] = {'RMSE': rnn_rmse, 'Runtime': time.time() - start_time}
    # XGBoost Model
    start_time = time.time()
    xgb_model = XGBRegressor(n_estimators=100, learning_rate=0.1, max_depth=5)
    xgb_model.fit(x_train.reshape(-1, 1), y_train)
    xgb_predictions = xgb_model.predict(x_test.reshape(-1, 1))
    xgb_rmse = np.sqrt(mean_squared_error(y_test, xgb_predictions))
    results['XGBoost'] = {'RMSE': xgb_rmse, 'Runtime': time.time() - start_time}
    # Linear SVR
    start_time = time.time()
    scaler = StandardScaler()
    x_train_scaled = scaler.fit_transform(x_train.reshape(-1, 1))
    x_test_scaled = scaler.transform(x_test.reshape(-1, 1))
    svr_model = SVR(kernel='linear', C=1.0)
    svr_model.fit(x_train_scaled, y_train_scaled)
    svr_predictions                                                    =
y_scaler.inverse_transform(svr_model.predict(x_test_scaled).reshape(-1, 1))
```

```
    svr_rmse = np.sqrt(mean_squared_error(y_test, svr_predictions))
    results['Linear SVR'] = {'RMSE': svr_rmse, 'Runtime': time.time() - start_time}
    # Plot Actual vs Predicted for all models with temperature (x) as x-axis
    models = ['ARIMA', 'Autoregressive', 'LSTM', 'RNN', 'XGBoost', 'Linear SVR']
    predictions = [y_scaler.inverse_transform(arima_predictions.reshape(-1, 1)).flatten(),
                   y_scaler.inverse_transform(ar_predictions.reshape(-1, 1)).flatten(),
                   lstm_predictions.flatten(),
                   rnn_predictions.flatten(),
                   xgb_predictions,
                   svr_predictions.flatten()]
    for i, model in enumerate(models):
        plt.figure(figsize=(10, 6))
        plt.scatter(x_test, y_test, label='Actual', alpha=0.6, color='blue')  # Set Actual to
blue
        plt.scatter(x_test, predictions[i], label='Predicted', alpha=0.6, color='red')  # Set
Predicted to red
        plt.title(f"{model} Predictions ({df_name})")
        plt.xlabel("Temperature (celsius)")
        plt.ylabel("Surface roughness (μm)")
        plt.legend()
        # Save each model plot as PNG
        plt.savefig(f"{df_name}_{model}_predictions.png")
        plt.close()  # Close the plot to avoid overlap with the next one
    print(f"Results for {df_name}:")
    for model, metrics in results.items():
        print(f"{model} - RMSE: {metrics['RMSE']:.4f}, Runtime: {metrics['Runtime']:.2f}
seconds")
dataframes = {
    "df20": df20,
    "df40": df40,
    "df60": df60,
    "df80": df80,
```

```
    "df100": df100
}
for df_name, df in dataframes.items():
    train_and_evaluate(df, df_name)
```

Code สำหรับ Training Acceleration และ Ae senor data

Import Library

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from statsmodels.tsa.arima.model import ARIMA
from sklearn.linear_model import LinearRegression
from xgboost import XGBRegressor
from sklearn.svm import LinearSVR
from keras.models import Sequential
from keras.layers import Dense, LSTM, SimpleRNN
from sklearn.metrics import mean_squared_error
import time
import matplotlib.pyplot as plt
import time
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.ar_model import AutoReg
import xgboost as xgb
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVR
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, SimpleRNN
from sklearn.metrics import mean_squared_error
import pandas as pd
```

```
from sklearn.model_selection import train_test_split

1 Loading Data
df = pd.read_csv('Data_acae/ALL_acae.csv')
df.head(2)
df.isnull().sum()
df.info()
df.isnull().sum()
2 Data Preprocessing
2.1 Changing Data type
df_cleaned = df[df['Accelerometer(g)'] != '#VALUE!']
# Function to check if a value can be converted to a float
def is_convertible_to_float(value):
    try:
        float(value)
        return True
    except ValueError:
        return False
# Filter out non-convertible values
df_cleaned                                                                    =
df_cleaned[df_cleaned['Accelerometer(g)'].apply(is_convertible_to_float)]
# Convert the remaining values to float
df_cleaned['Accelerometer(g)'] = df_cleaned['Accelerometer(g)'].astype(float)
df_cleaned.info()
df.shape
df_cleaned.shape
df.info()
df.head(2)
df_cleaned['Acoustic(dB)'] = df_cleaned['Acoustic(dB)'].astype(float)
df = df_cleaned.copy()
df.info()
```

2.2 Changing Value Format

```
df['Surface'] = df['Surface'].apply(lambda x: str(x).replace(',', '.'))
df['Surface'] = df['Surface'].astype(float)
df['Sample'] = df['Sample'].astype(int)
df['No'] = df['No'].astype(int)
df.head(2)
```

2.3 Convert time-series data to frequency domain(Fourier Transform)

```
def convert_to_frequency_domain(data):
    return np.abs(np.fft.fft(data))
df['Accelerometer_freq'] =
df.groupby(['No'])['Accelerometer(g)'].transform(convert_to_frequency_domain)
```

2.4 Group data

```
df.info()
unique_values_no = df['No'].unique()
print("Unique values in the 'No' column:", unique_values_no)
df20 = df[df['No'] == 20]
df40 = df[df['No'] == 40]
df60 = df[df['No'] == 60]
df80 = df[df['No'] == 80]
df100 = df[df['No'] == 100]
df20
```

2.5 Cut off Data

```
print("Shape of df20:", df20.shape)
print("Shape of df40:", df40.shape)
print("Shape of df60:", df60.shape)
print("Shape of df80:", df80.shape)
print("Shape of df100:", df100.shape)
rows_df20 = 128006
rows_df40 = 253495
rows_df60 = 373581
rows_df80 = 518655
rows_df100 = 626325
```

```
# Slice the original DataFrame to create each smaller DataFrame
df20 = df20.iloc[:rows_df20, :]
df40 = df40.iloc[:rows_df40, :]
df60 = df60.iloc[:rows_df60, :]
df80 = df80.iloc[:rows_df80, :]
df100 = df100.iloc[:rows_df100, :]
print("Shape of df20:", df20.shape)
print("Shape of df40:", df40.shape)
print("Shape of df60:", df60.shape)
print("Shape of df80:", df80.shape)
print("Shape of df100:", df100.shape)
df40
2.6 split data acceleration and ae sensor
dataframes = [df20, df40, df60, df80, df100]
columns_to_keep = ['Accelerometer(g)', 'Surface']
df_names = ['df20', 'df40', 'df60', 'df80', 'df100']
filtered_dataframes = {}
for df, name in zip(dataframes, df_names):
    filtered_df = df[columns_to_keep]
    filtered_dataframes[name] = filtered_df
df20_ae = filtered_dataframes['df20']
df40_ae = filtered_dataframes['df40']
df60_ae = filtered_dataframes['df60']
df80_ae = filtered_dataframes['df80']
df100_ae = filtered_dataframes['df100']
df20_ae
dataframes = [df20, df40, df60, df80, df100]
columns_to_keep = ['Acoustic(dB)', 'Surface']
df_names = ['df20', 'df40', 'df60', 'df80', 'df100']
filtered_dataframes = {}
for df, name in zip(dataframes, df_names):
    filtered_df = df[columns_to_keep]
```

```
    filtered_dataframes[name] = filtered_df
df20_ac = filtered_dataframes['df20']
df40_ac = filtered_dataframes['df40']
df60_ac = filtered_dataframes['df60']
df80_ac = filtered_dataframes['df80']
df100_ac = filtered_dataframes['df100']
df20_ac
```

2.7 Split Data train and test 70:30

```
def split_and_drop(df, test_size=0.3):
    train, test = train_test_split(df, test_size=test_size, random_state=42)
    train = df.loc[df.index.difference(test.index)]
    return train, test
train20_ac, test20_ac = split_and_drop(df20_ac)
train40_ac, test40_ac = split_and_drop(df40_ac)
train60_ac, test60_ac = split_and_drop(df60_ac)
train80_ac, test80_ac = split_and_drop(df80_ac)
train100_ac, test100_ac = split_and_drop(df100_ac)
train20_ae, test20_ae = split_and_drop(df20_ae)
train40_ae, test40_ae = split_and_drop(df40_ae)
train60_ae, test60_ae = split_and_drop(df60_ae)
train80_ae, test80_ae = split_and_drop(df80_ae)
train100_ae, test100_ae = split_and_drop(df100_ae)
print("Train and test shapes:")
print("df20_ac - Train:", train20_ac.shape, "Test:", test20_ac.shape)
print("df40_ac - Train:", train40_ac.shape, "Test:", test40_ac.shape)
print("df60_ac - Train:", train60_ac.shape, "Test:", test60_ac.shape)
print("df80_ac - Train:", train80_ac.shape, "Test:", test80_ac.shape)
print("df10_ac0_ac - Train:", train100_ac.shape, "Test:", test100_ac.shape)
```

2.8 Definding x and y

```
data_train_ac = [train20_ac, train40_ac, train60_ac, train80_ac, train100_ac]
data_test_ac = [test20_ac, test40_ac, test60_ac, test80_ac, test100_ac]
X_train_list = []
```

```
y_train_list = []
X_test_list = []
y_test_list = []
for train_df, test_df in zip(data_train_ac, data_test_ac):
    X_train = train_df.drop(columns=['Surface'])
    y_train = train_df['Surface']
    X_test = test_df.drop(columns=['Surface'])
    y_test = test_df['Surface']
    X_train_list.append(X_train)
    y_train_list.append(y_train)
    X_test_list.append(X_test)
    y_test_list.append(y_test)
X_train_20_ac = X_train_list[0].values
y_train_20_ac = y_train_list[0].values
X_test_20_ac = X_test_list[0].values
y_test_20_ac = y_test_list[0].values
X_train_40_ac = X_train_list[1].values
y_train_40_ac = y_train_list[1].values
X_test_40_ac = X_test_list[1].values
y_test_40_ac = y_test_list[1].values
X_train_60_ac = X_train_list[2].values
y_train_60_ac = y_train_list[2].values
X_test_60_ac = X_test_list[2].values
y_test_60_ac = y_test_list[2].values
X_train_80_ac = X_train_list[3].values
y_train_80_ac = y_train_list[3].values
X_test_80_ac = X_test_list[3].values
y_test_80_ac = y_test_list[3].values
X_train_100_ac = X_train_list[4].values
y_train_100_ac = y_train_list[4].values
X_test_100_ac = X_test_list[4].values
y_test_100_ac = y_test_list[4].values
```

```
data_train_ae = [train20_ae, train40_ae, train60_ae, train80_ae, train100_ae]
data_test_ae = [test20_ae, test40_ae, test60_ae, test80_ae, test100_ae]
X_train_list = []
y_train_list = []
X_test_list = []
y_test_list = []
for train_df, test_df in zip(data_train_ae, data_test_ae):
    X_train = train_df.drop(columns=['Surface'])
    y_train = train_df['Surface']
    X_test = test_df.drop(columns=['Surface'])
    y_test = test_df['Surface']
    X_train_list.append(X_train)
    y_train_list.append(y_train)
    X_test_list.append(X_test)
    y_test_list.append(y_test)
X_train_20_ae = X_train_list[0].values
y_train_20_ae = y_train_list[0].values
X_test_20_ae = X_test_list[0].values
y_test_20_ae = y_test_list[0].values
X_train_40_ae = X_train_list[1].values
y_train_40_ae = y_train_list[1].values
X_test_40_ae = X_test_list[1].values
y_test_40_ae = y_test_list[1].values
X_train_60_ae = X_train_list[2].values
y_train_60_ae = y_train_list[2].values
X_test_60_ae = X_test_list[2].values
y_test_60_ae = y_test_list[2].values
X_train_80_ae = X_train_list[3].values
y_train_80_ae = y_train_list[3].values
X_test_80_ae = X_test_list[3].values
y_test_80_ae = y_test_list[3].values
X_train_100_ae = X_train_list[4].values
```

```
y_train_100_ae = y_train_list[4].values
X_test_100_ae = X_test_list[4].values
y_test_100_ae = y_test_list[4].values
3 Ploting
def plot_predictions(X_test, y_test, y_pred, title):
    plt.figure(figsize=(12, 6))
    # Use the first column of X_test if it's 2D, or X_test itself if it's 1D
    x_values = X_test[:, 0] if X_test.ndim > 1 else X_test
    # Sort the values for smooth plotting
    sorted_indices = np.argsort(x_values)
    x_sorted = x_values[sorted_indices]
    y_pred_sorted = y_pred[sorted_indices]
    # Plot actual values
    plt.scatter(x_values, y_test, label='Actual', color='blue', alpha=0.5, s=10)
    # Plot predicted values as a trend line
    plt.scatter(x_sorted, y_pred_sorted, label='Predicted Trend', color='red', alpha=0.5,
s=5, marker=(5, 1))
    plt.title(title)
    plt.xlabel('Feature Value')
    plt.ylabel('Surface Area')
    plt.legend()
    plt.grid(True, linestyle='--', alpha=0.7)
    plt.show()
4 Model training
def evaluate_models(X_train, y_train, X_test, y_test, label):
    # Dictionary to store RMSE and runtime
    results = {}
    # ARIMA Model
    start_time = time.time()
    arima_model = ARIMA(y_train, order=(0, 1, 0))
    arima_result = arima_model.fit()
    y_pred_arima = arima_result.forecast(steps=len(y_test))
```

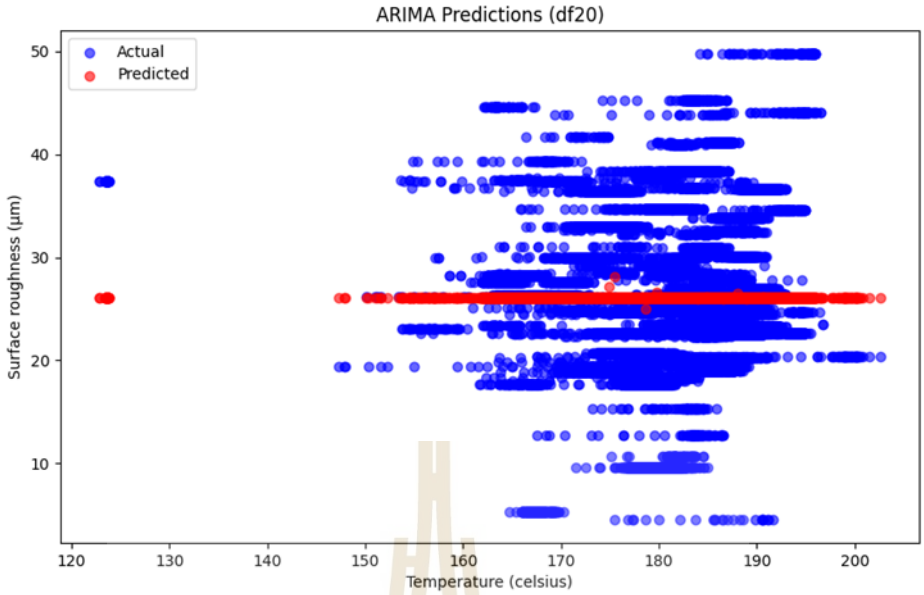
```
end_time = time.time()
arima_runtime = end_time - start_time
rmse_arima = np.sqrt(mean_squared_error(y_test, y_pred_arima))
results['ARIMA'] = (rmse_arima, arima_runtime)
print(f"ARIMA ({label}) RMSE: {rmse_arima}, Runtime: {arima_runtime:.2f} seconds")
plot_predictions(X_test, y_test, y_pred_arima, f"ARIMA Predictions ({label})")
# Autoregressive Model
start_time = time.time()
ar_model = AutoReg(y_train, lags=5)
ar_result = ar_model.fit()
y_pred_ar = ar_result.predict(start=len(y_train), end=len(y_train) + len(y_test) - 1)
end_time = time.time()
ar_runtime = end_time - start_time
rmse_ar = np.sqrt(mean_squared_error(y_test, y_pred_ar))
results['Autoregressive'] = (rmse_ar, ar_runtime)
print(f"Autoregressive ({label}) RMSE: {rmse_ar}, Runtime: {ar_runtime:.2f} seconds")
plot_predictions(X_test, y_test, y_pred_ar, f"Autoregressive Predictions ({label})")
# XGBoost Model
start_time = time.time()
xgb_model = xgb.XGBRegressor(objective='reg:squarederror')
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)
end_time = time.time()
xgb_runtime = end_time - start_time
rmse_xgb = np.sqrt(mean_squared_error(y_test, y_pred_xgb))
results['XGBoost'] = (rmse_xgb, xgb_runtime)
print(f"XGBoost ({label}) RMSE: {rmse_xgb}, Runtime: {xgb_runtime:.2f} seconds")
plot_predictions(X_test, y_test, y_pred_xgb, f"XGBoost Predictions ({label})")
# Linear SVR Model
start_time = time.time()
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
```

```
    X_test_scaled = scaler.transform(X_test)
    linear_svr_model = LinearSVR()
    linear_svr_model.fit(X_train_scaled, y_train)
    y_pred_linear_svr = linear_svr_model.predict(X_test_scaled)
    end_time = time.time()
    linear_svr_runtime = end_time - start_time
    rmse_linear_svr = np.sqrt(mean_squared_error(y_test, y_pred_linear_svr))
    results['LinearSVR'] = (rmse_linear_svr, linear_svr_runtime)
    print(f"LinearSVR ({label}) RMSE: {rmse_linear_svr}, Runtime: {linear_svr_runtime:.2f}
seconds")
    plot_predictions(X_test, y_test, y_pred_linear_svr, f"LinearSVR Predictions ({label})")
    # LSTM Model
    start_time = time.time()
    X_train_lstm = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
    X_test_lstm = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))
    lstm_model = Sequential()
    lstm_model.add(LSTM(50, activation='relu', input_shape=(X_train_lstm.shape[1],
1)))
    lstm_model.add(Dense(1))
    lstm_model.compile(optimizer='adam', loss='mse')
    lstm_model.fit(X_train_lstm, y_train, epochs=200, batch_size=32, verbose=0)
    y_pred_lstm = lstm_model.predict(X_test_lstm)
    end_time = time.time()
    lstm_runtime = end_time - start_time
    rmse_lstm = np.sqrt(mean_squared_error(y_test, y_pred_lstm))
    results['LSTM'] = (rmse_lstm, lstm_runtime)
    print(f"LSTM ({label}) RMSE: {rmse_lstm}, Runtime: {lstm_runtime:.2f} seconds")
    plot_predictions(X_test[:, 0], y_test, y_pred_lstm, f"LSTM Predictions ({label})")
    # RNN Model
    start_time = time.time()
    rnn_model = Sequential()
```

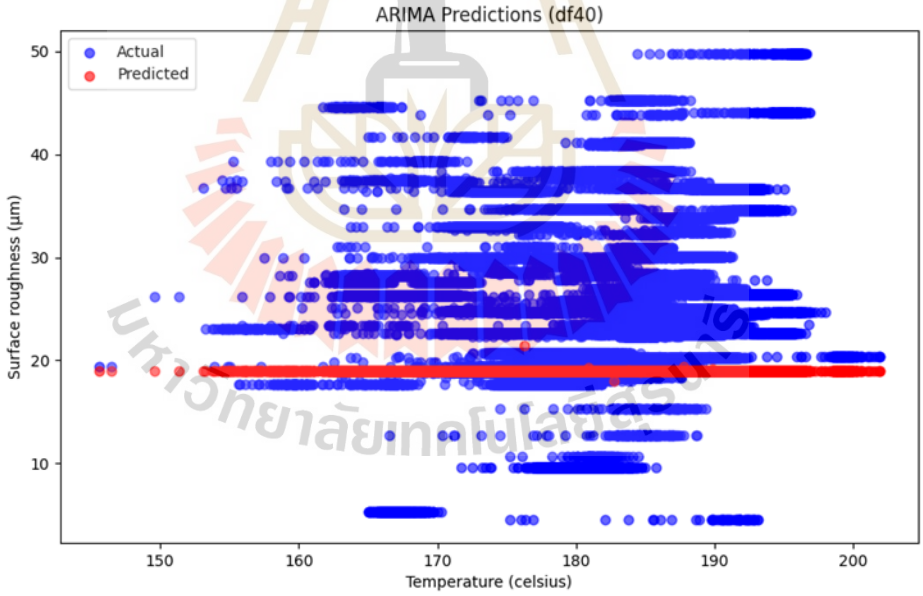
```
    rnn_model.add(SimpleRNN(50, activation='relu', input_shape=(X_train_lstm.shape[1], 1)))
    rnn_model.add(Dense(1))
    rnn_model.compile(optimizer='adam', loss='mse')
    rnn_model.fit(X_train_lstm, y_train, epochs=200, batch_size=32, verbose=0)
    y_pred_rnn = rnn_model.predict(X_test_lstm)
    end_time = time.time()
    rnn_runtime = end_time - start_time
    rmse_rnn = np.sqrt(mean_squared_error(y_test, y_pred_rnn))
    results['RNN'] = (rmse_rnn, rnn_runtime)
    print(f"RNN ({label}) RMSE: {rmse_rnn}, Runtime: {rnn_runtime:.2f} seconds")
    plot_predictions(X_test[:, 0], y_test, y_pred_rnn, f"RNN Predictions ({label})")
    return results
datasets_ac = [
    ('20_ac', X_train_20_ac, y_train_20_ac, X_test_20_ac, y_test_20_ac),
    ('40_ac', X_train_40_ac, y_train_40_ac, X_test_40_ac, y_test_40_ac),
    ('60_ac', X_train_60_ac, y_train_60_ac, X_test_60_ac, y_test_60_ac),
    ('80_ac', X_train_80_ac, y_train_80_ac, X_test_80_ac, y_test_80_ac),
    ('100_ac', X_train_100_ac, y_train_100_ac, X_test_100_ac, y_test_100_ac),
]
datasets_ae = [
    ('20_ae', X_train_20_ae, y_train_20_ae, X_test_20_ae, y_test_20_ae),
    ('40_ae', X_train_40_ae, y_train_40_ae, X_test_40_ae, y_test_40_ae),
    ('60_ae', X_train_60_ae, y_train_60_ae, X_test_60_ae, y_test_60_ae),
    ('80_ae', X_train_80_ae, y_train_80_ae, X_test_80_ae, y_test_80_ae),
    ('100_ae', X_train_100_ae, y_train_100_ae, X_test_100_ae, y_test_100_ae),
]
5 Model Evaluation
all_results_ac = {}
for label, X_train, y_train, X_test, y_test in datasets_ac:
    print(f"\nEvaluating models for {label}% of data...")
    results = evaluate_models(X_train, y_train, X_test, y_test, label)
```

```
    all_results_ac[label] = results
all_results_ae = {}
for label, X_train, y_train, X_test, y_test in datasets_ae:
    print(f"\nEvaluating models for {label}% of data...")
    results = evaluate_models(X_train, y_train, X_test, y_test, label)
    all_results_ae[label] = results
for label, result in all_results_ac.items():
    print(f"\nResults for {label}% data:")
    for model_name, (rmse, runtime) in result.items():
        print(f"{model_name} RMSE: {rmse}, Runtime: {runtime:.2f} seconds")
```

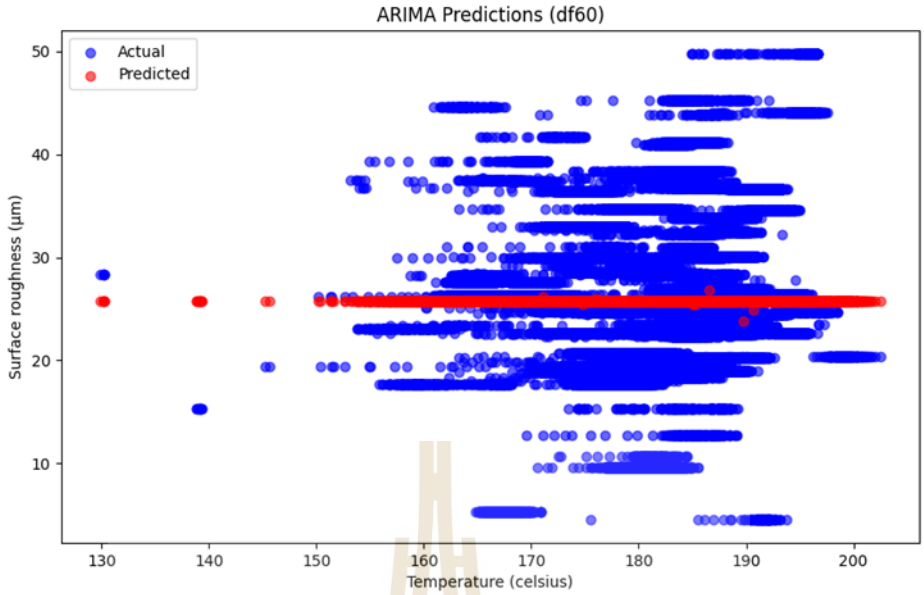
ภาคผนวก ข
การทำงานของโมเดล



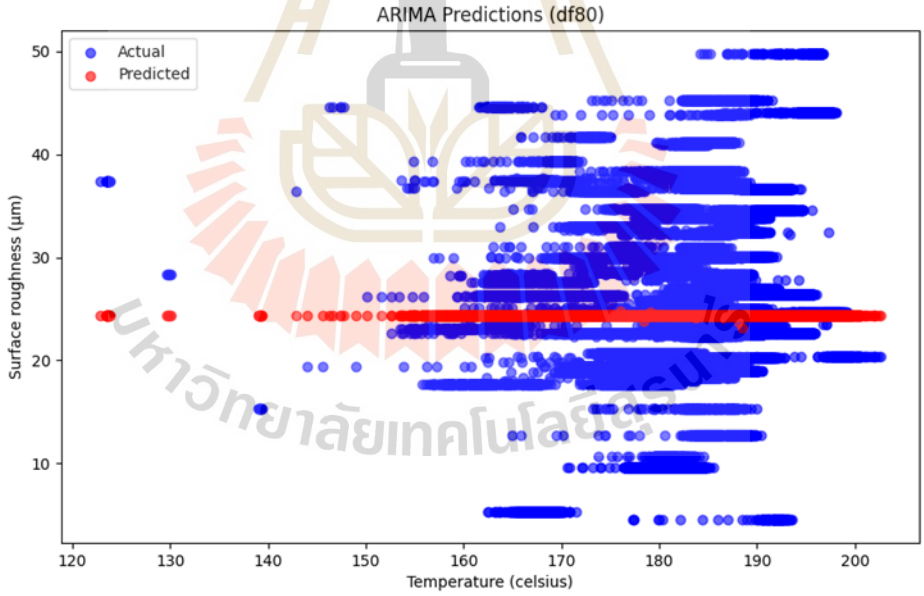
รูปที่ ข.1 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



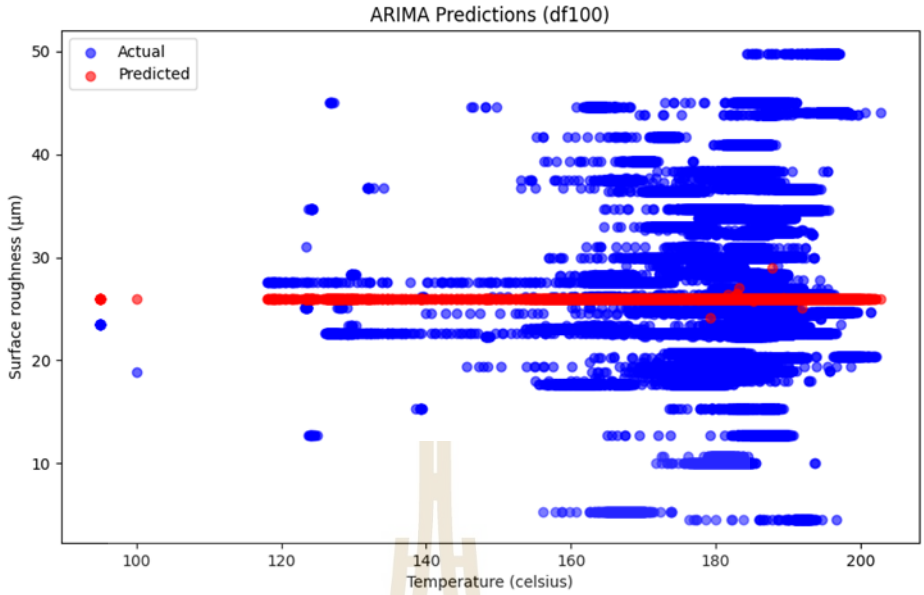
รูปที่ ข.2 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



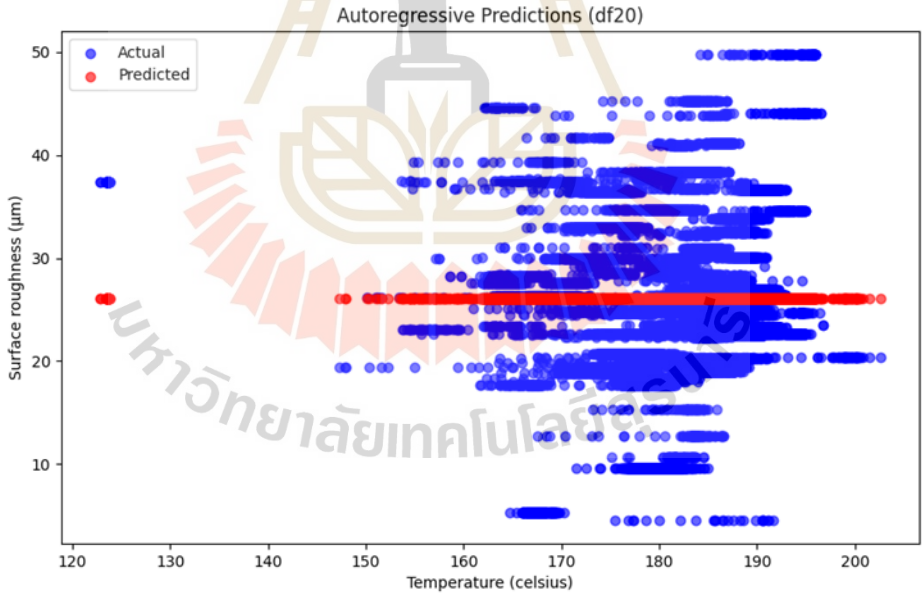
รูปที่ ข.3 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



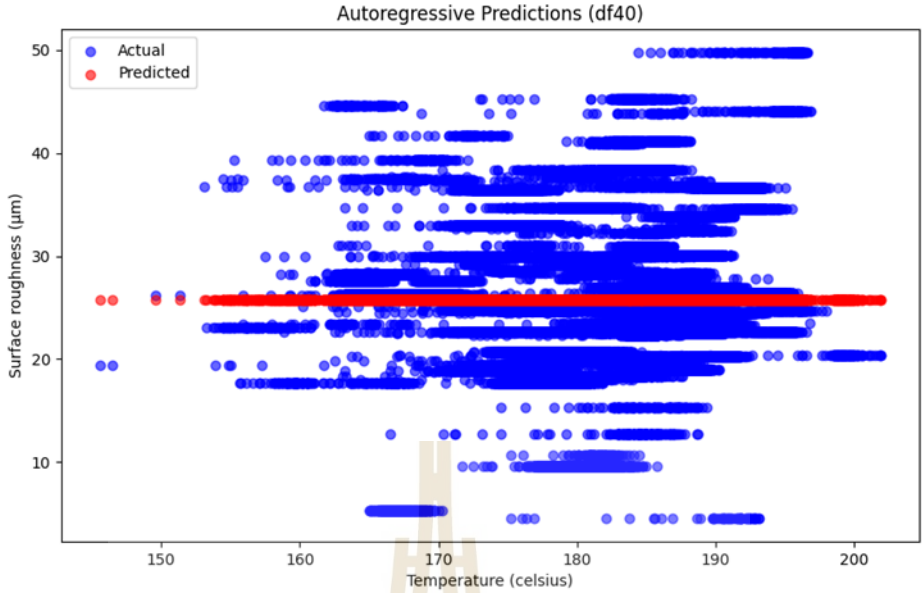
รูปที่ ข.4 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



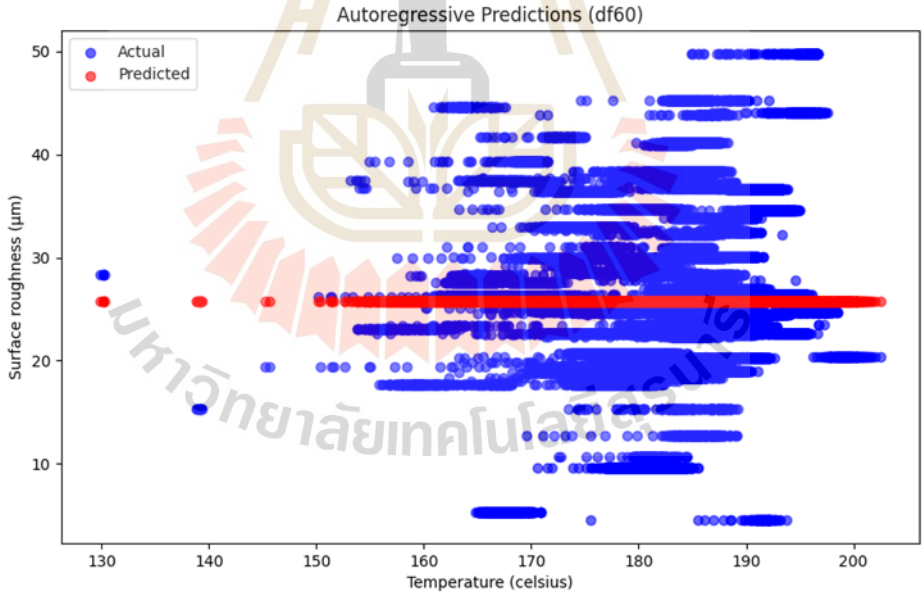
รูปที่ ข.5 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



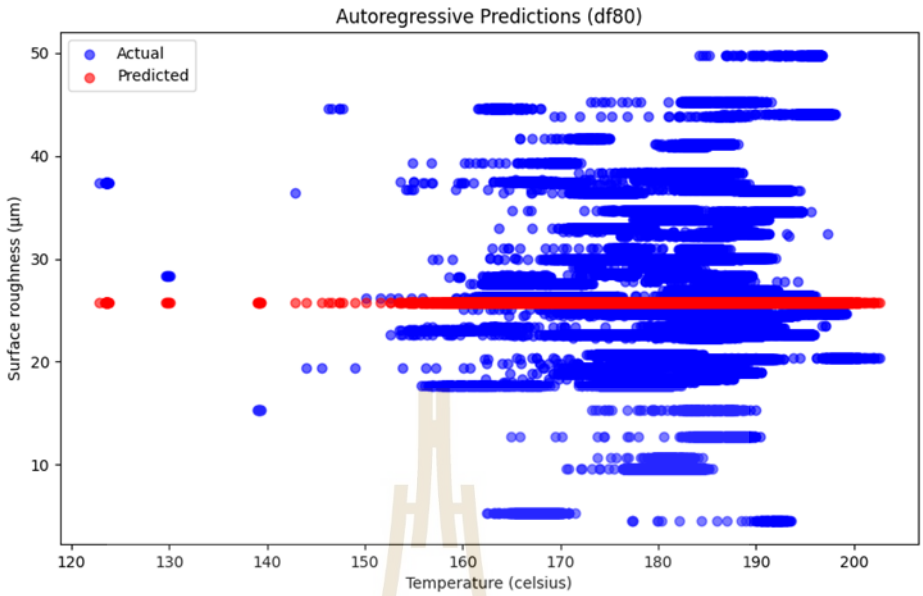
รูปที่ ข.6 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



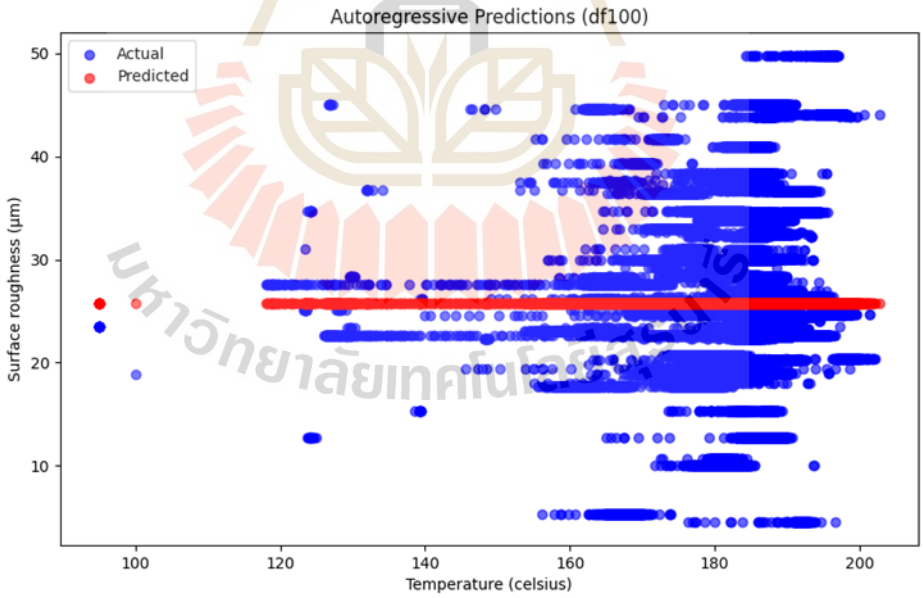
รูปที่ ข.7 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



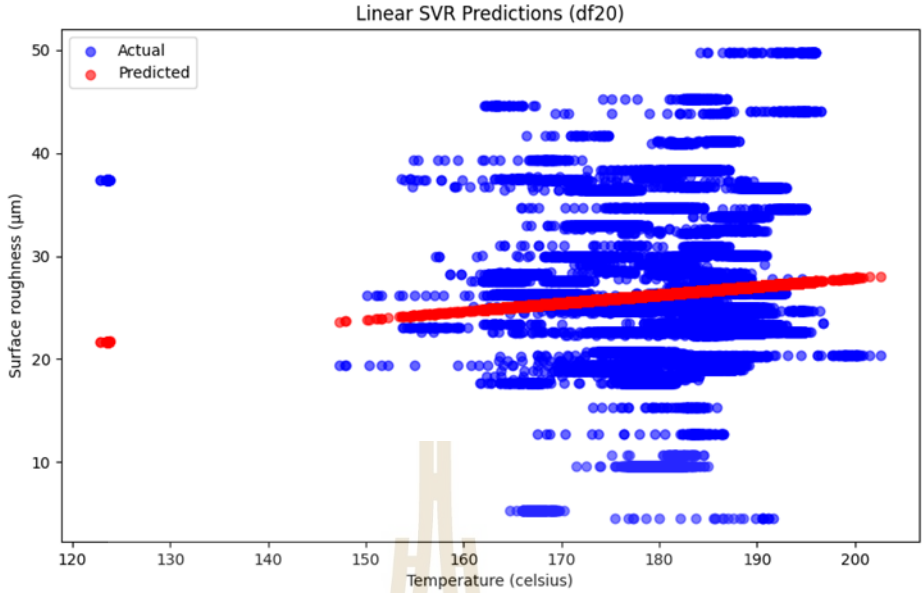
รูปที่ ข.8 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



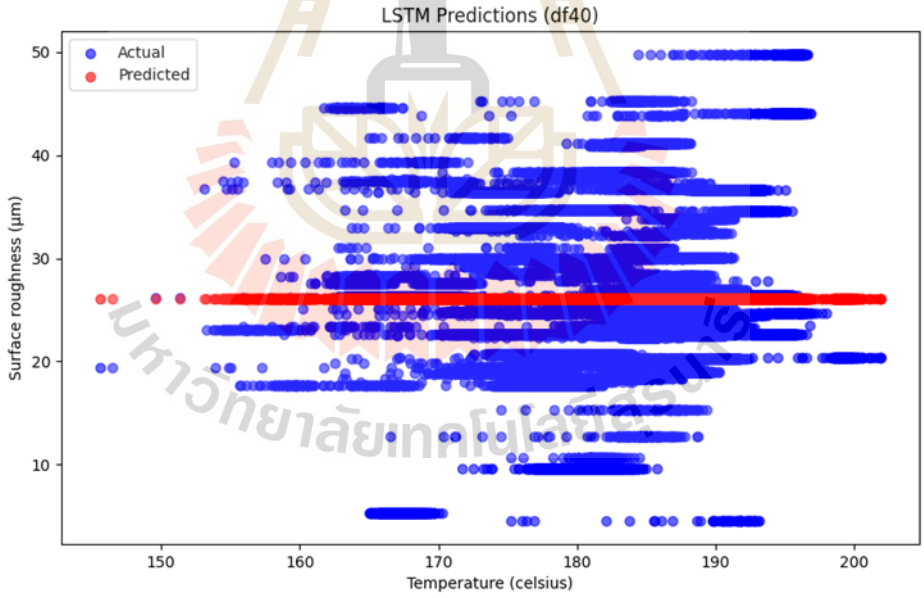
รูปที่ ข.9 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



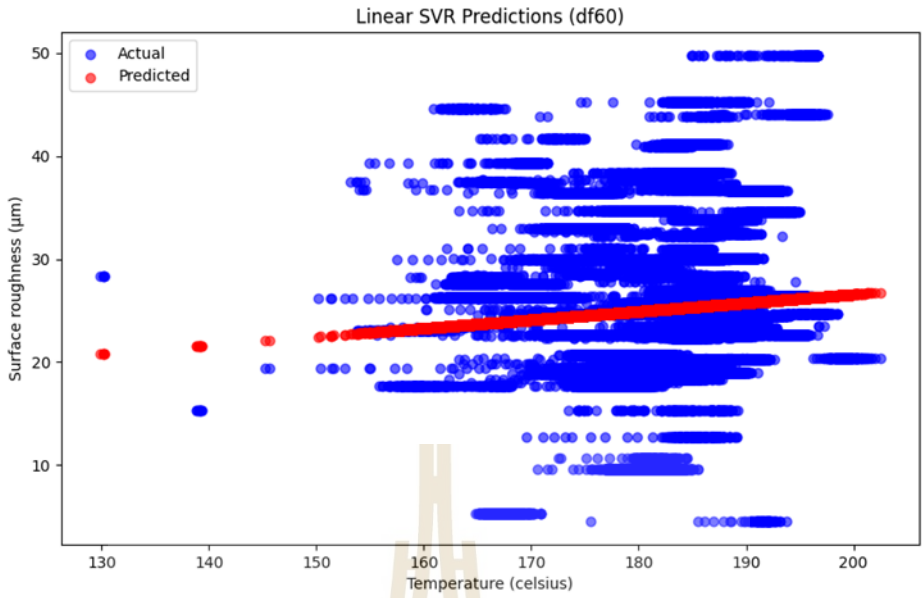
รูปที่ ข.10 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



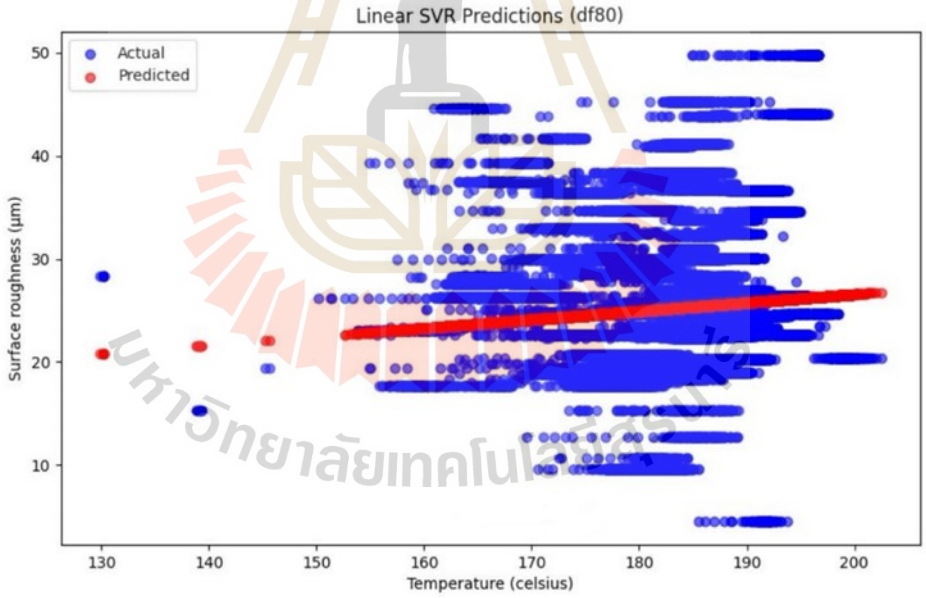
รูปที่ ข.11 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



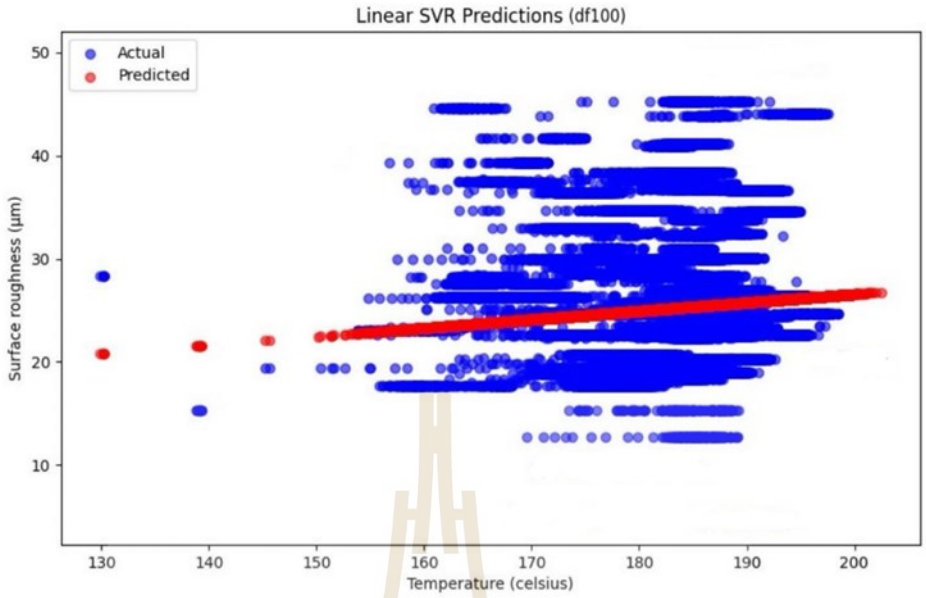
รูปที่ ข.12 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



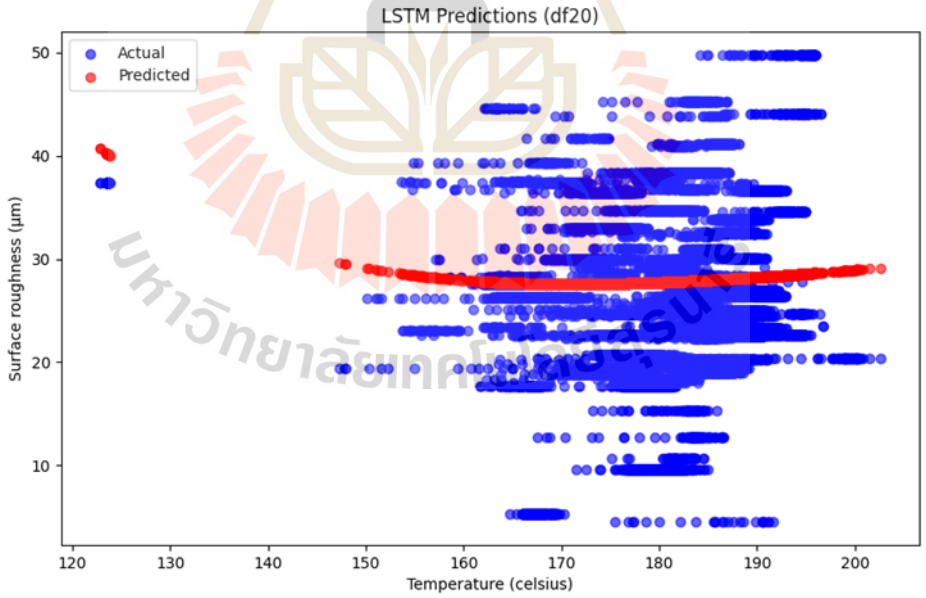
รูปที่ ข.13 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



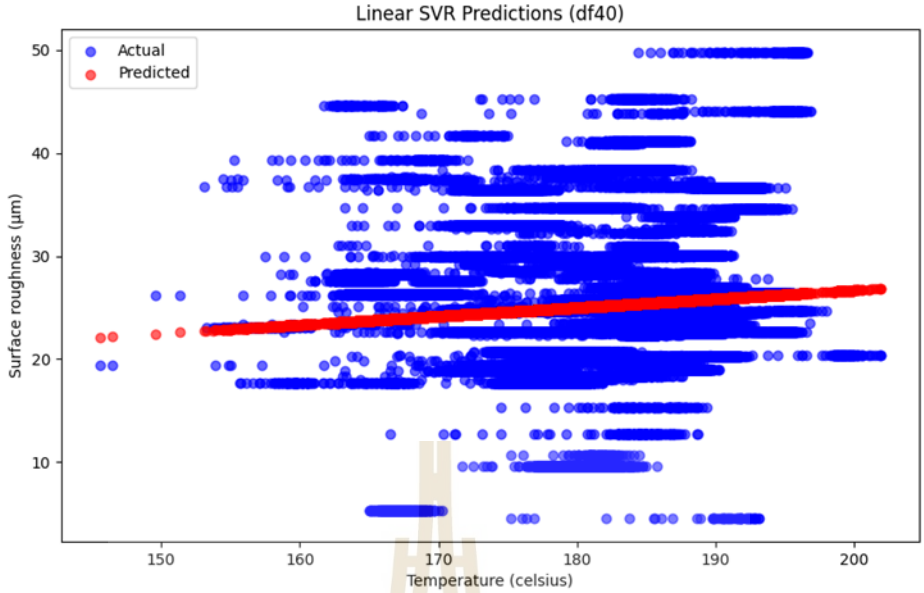
รูปที่ ข.14 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



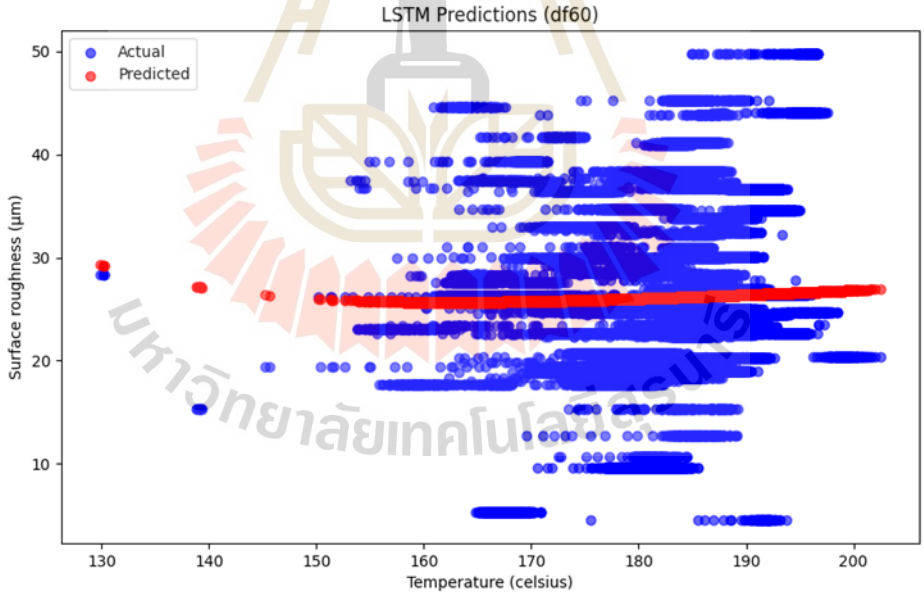
รูปที่ ข.15 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมด และใช้อัลกอริทึม Linear SVR



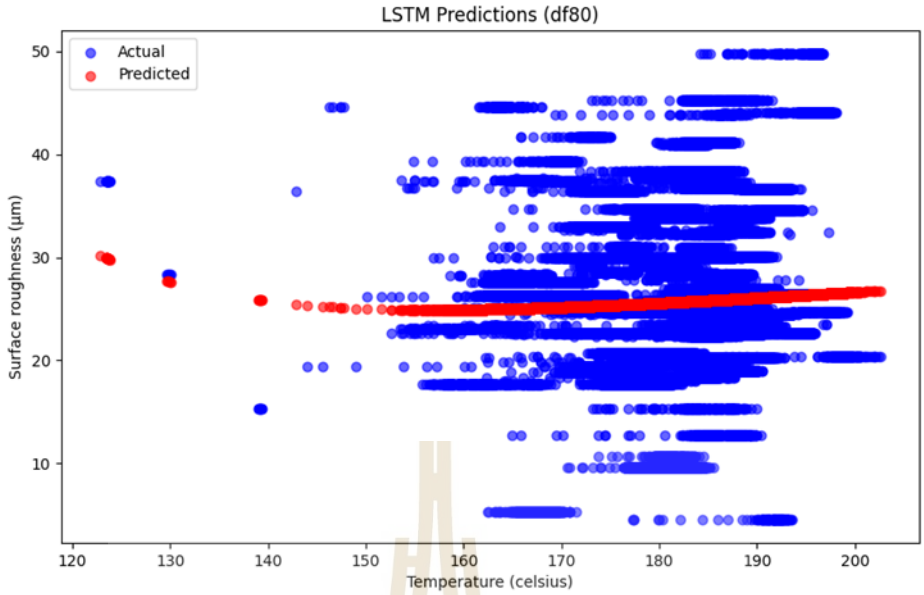
รูปที่ ข.16 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



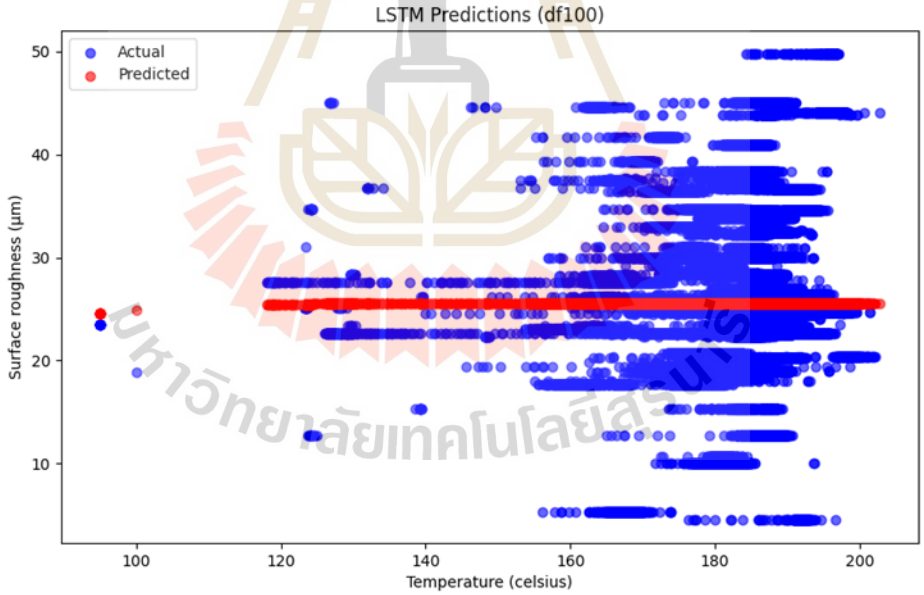
รูปที่ ข.17 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



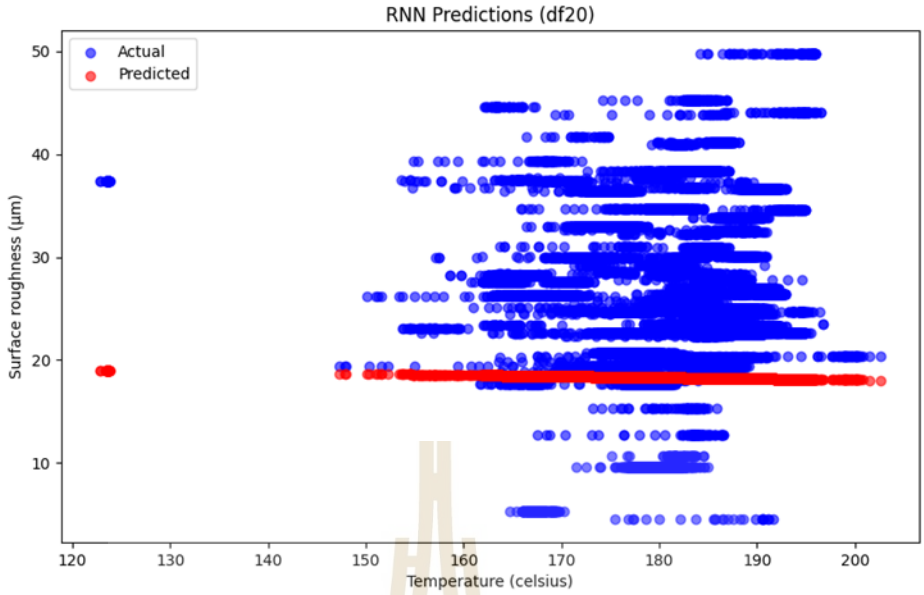
รูปที่ ข.18 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



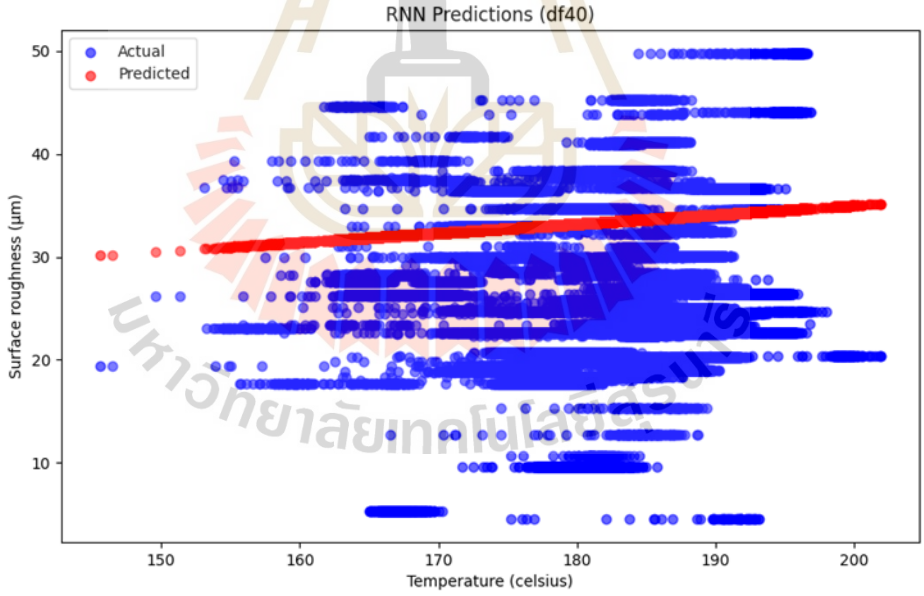
รูปที่ ข.19 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



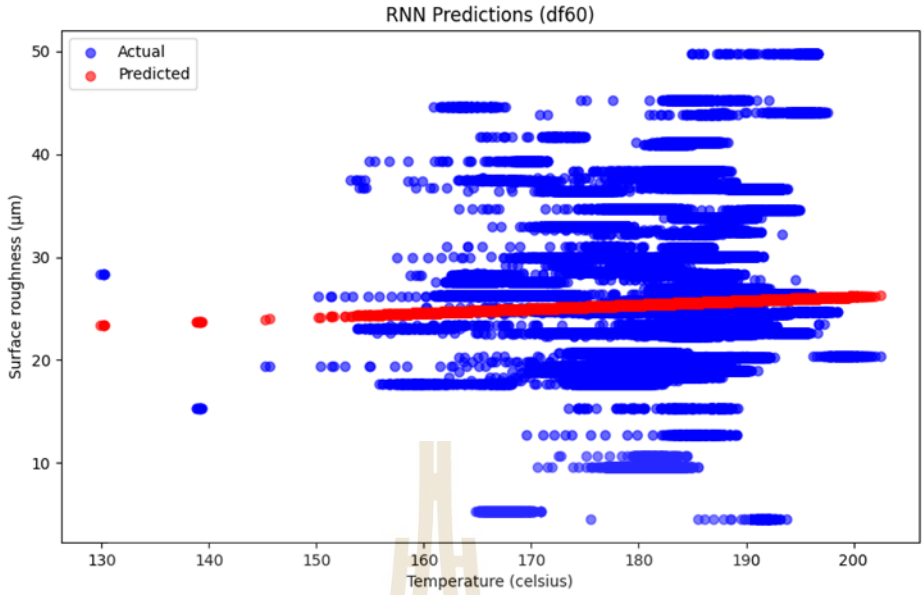
รูปที่ ข.20 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



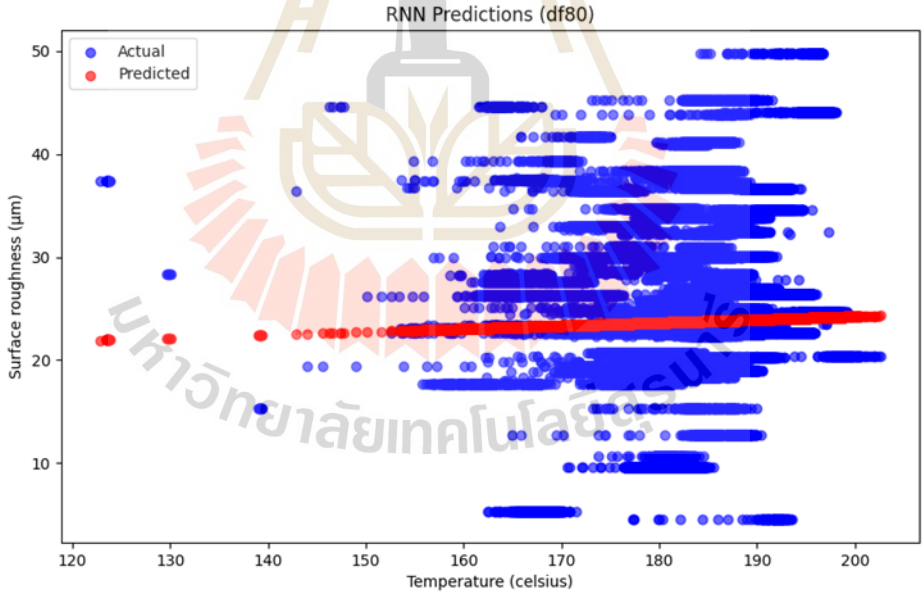
รูปที่ ข.21 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



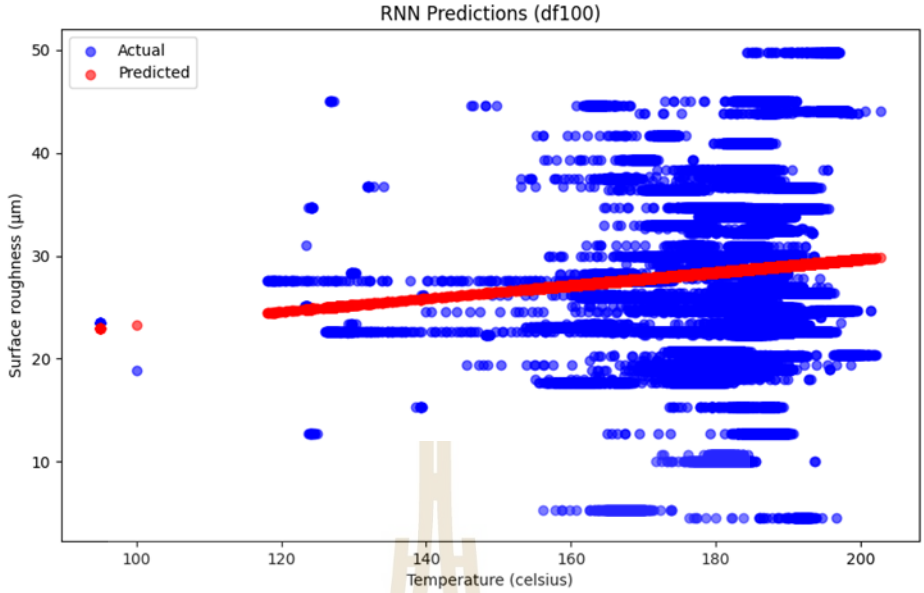
รูปที่ ข.22 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



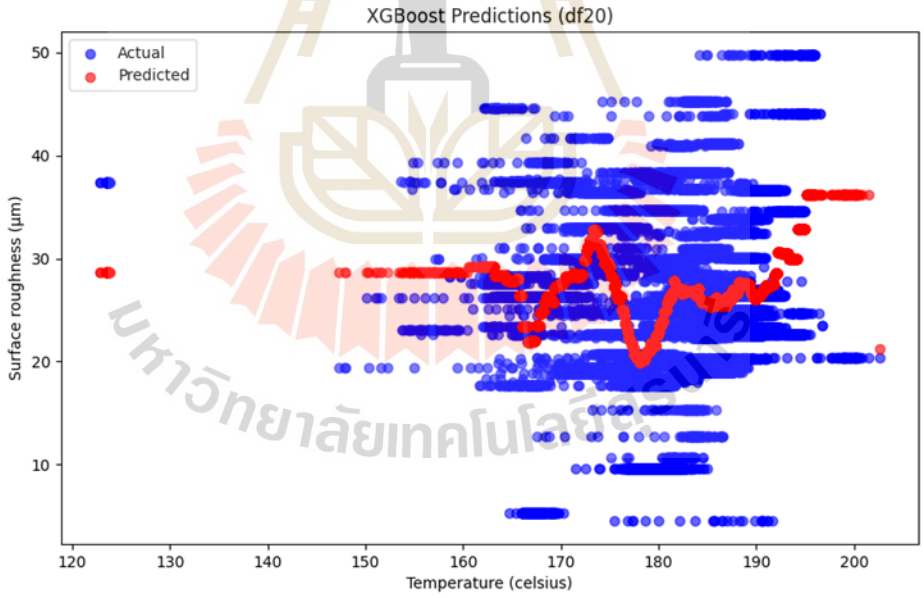
รูปที่ ข.23 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



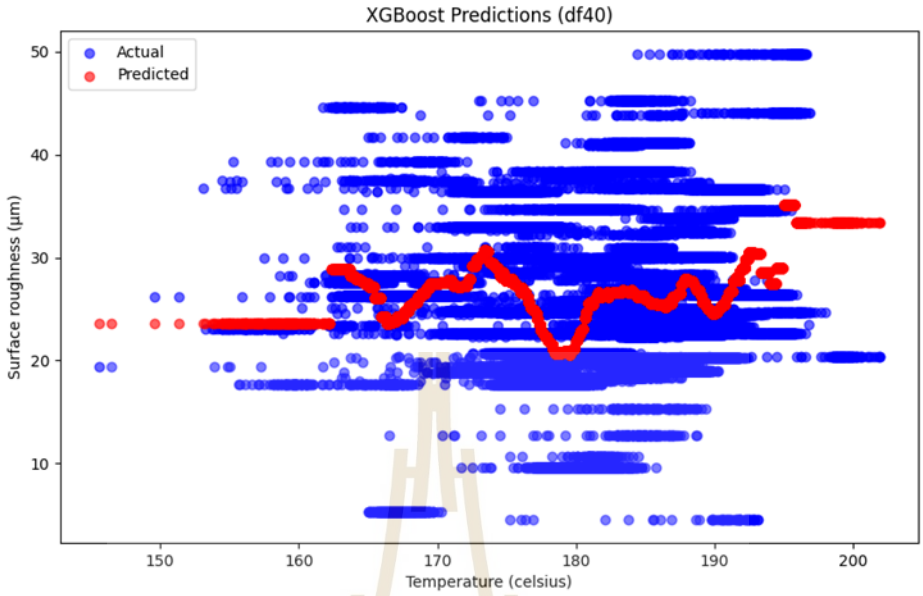
รูปที่ ข.24 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



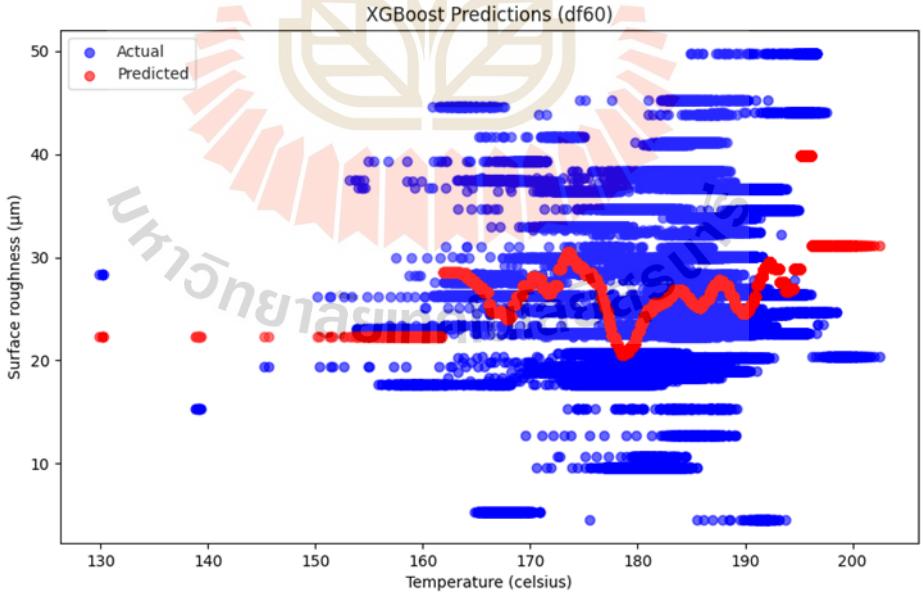
รูปที่ ข.25 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



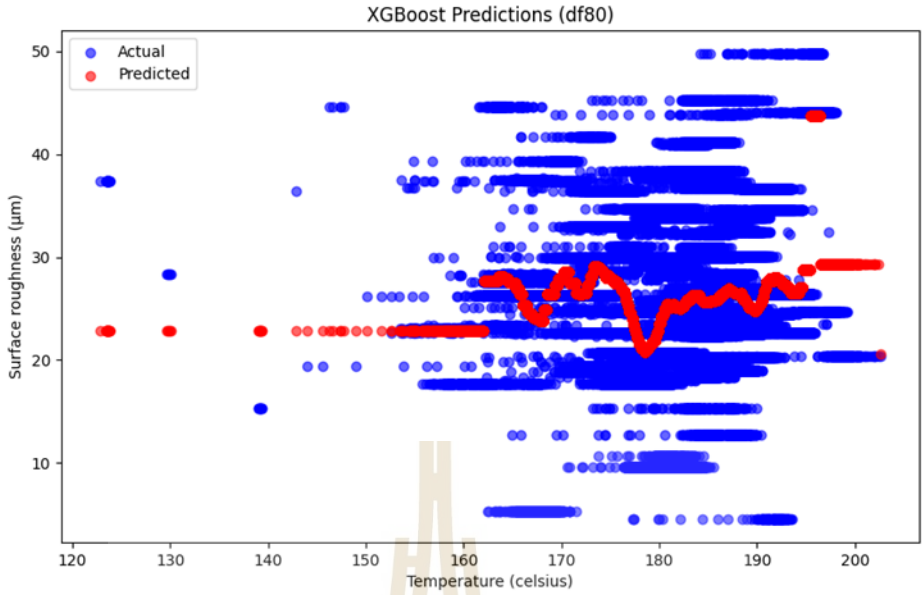
รูปที่ ข.26 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



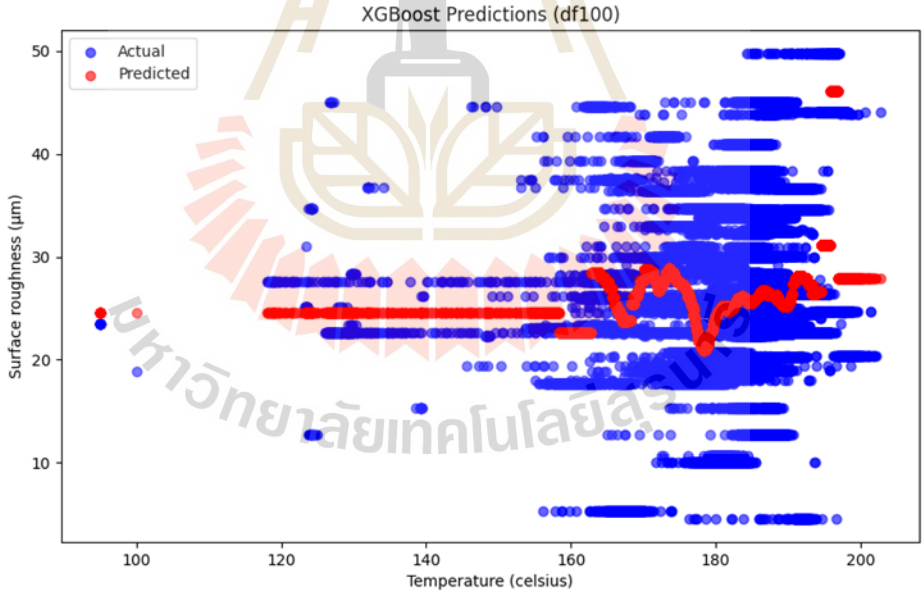
รูปที่ ข.27 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



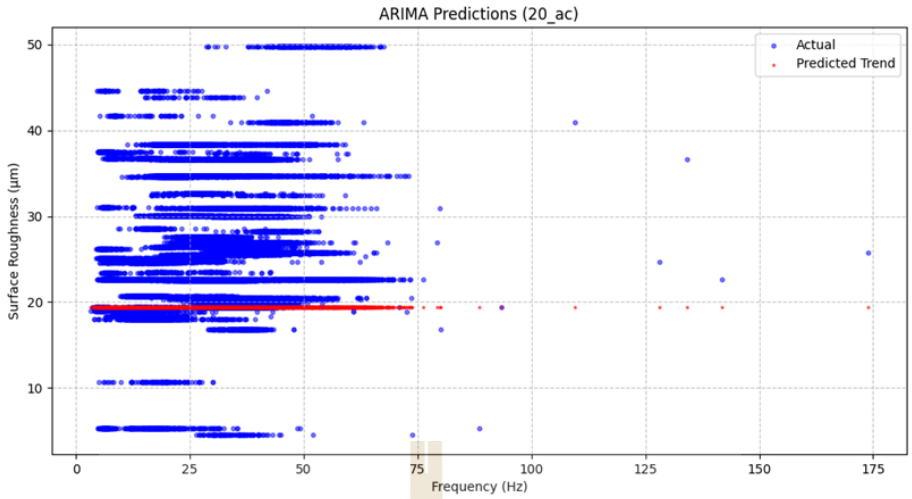
รูปที่ ข.28 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



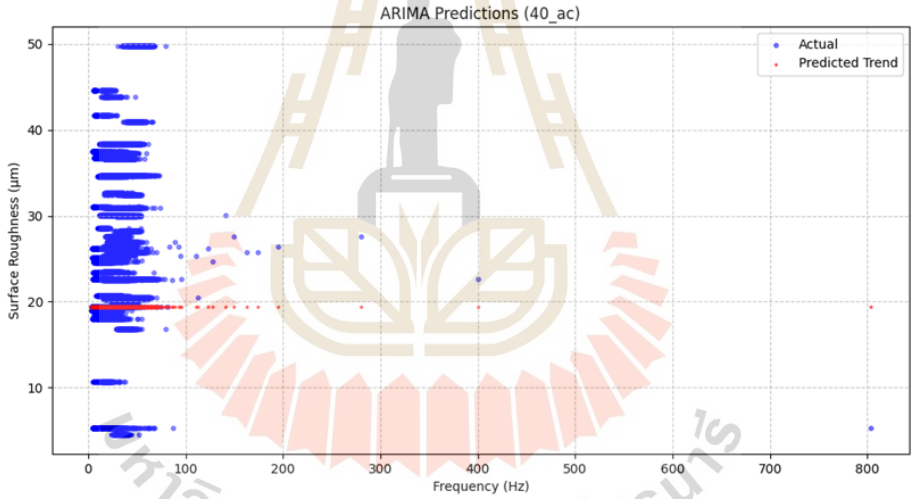
รูปที่ ข.29 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



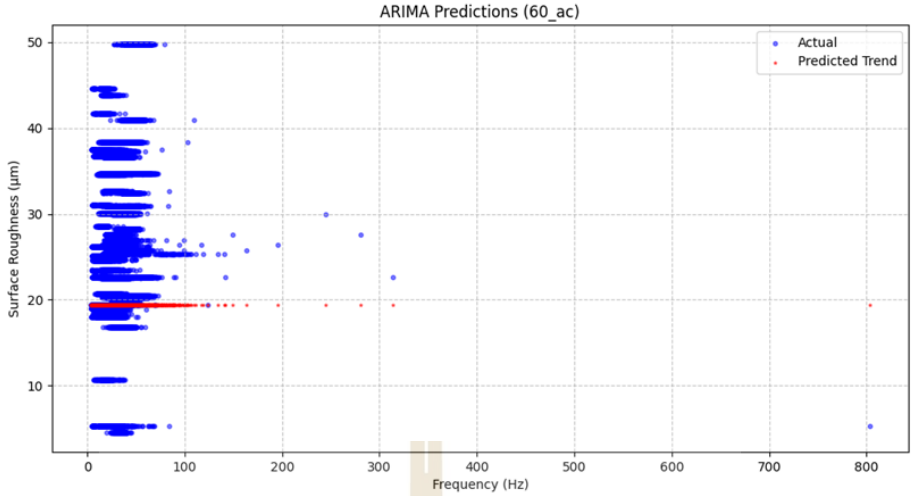
รูปที่ ข. 30 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Thermocouple ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



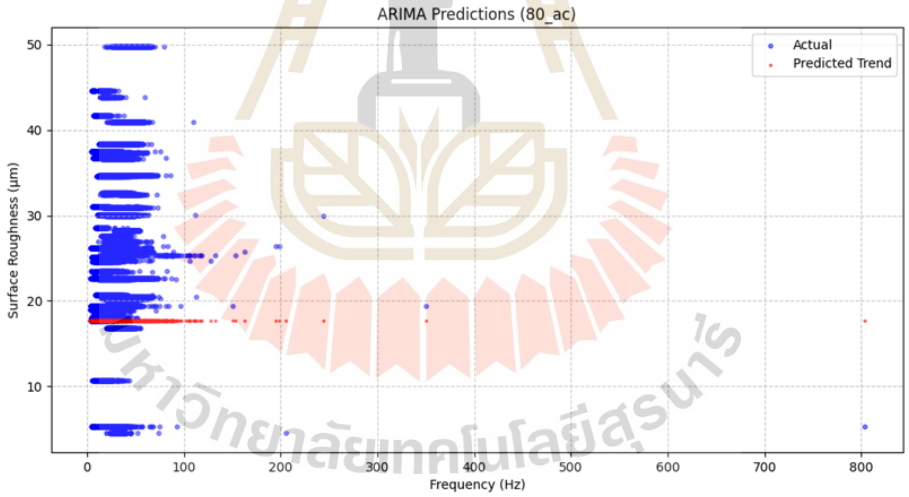
รูปที่ ข. 31 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



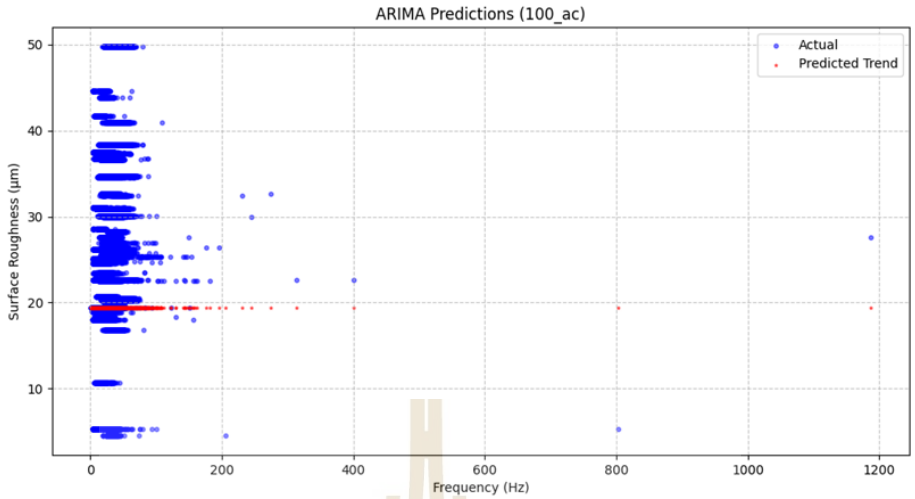
รูปที่ ข.32 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



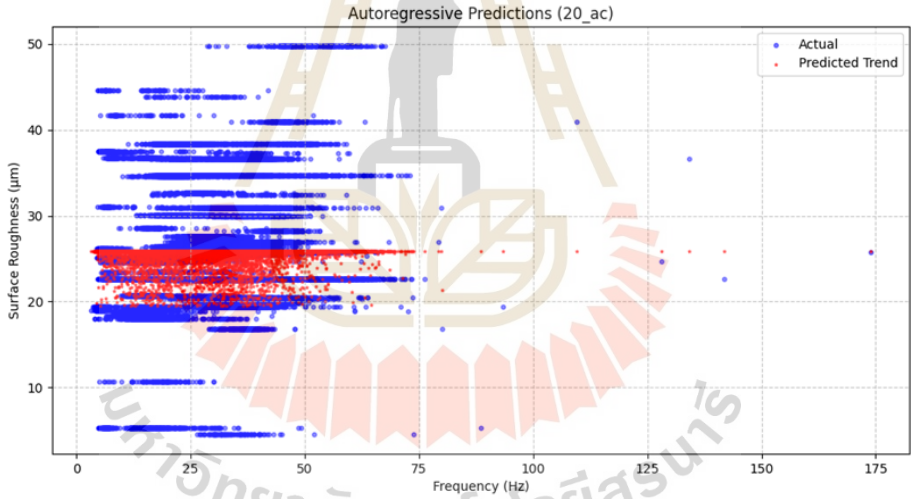
รูปที่ ข.33 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



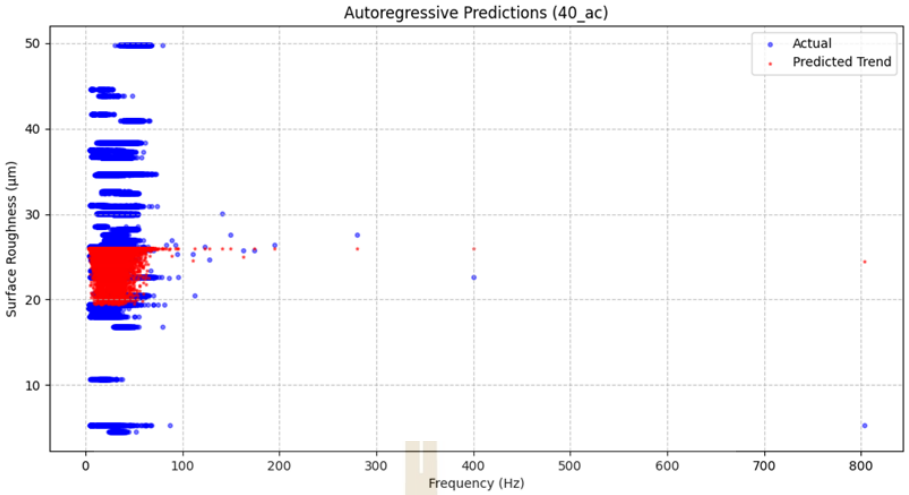
รูปที่ ข. 34 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



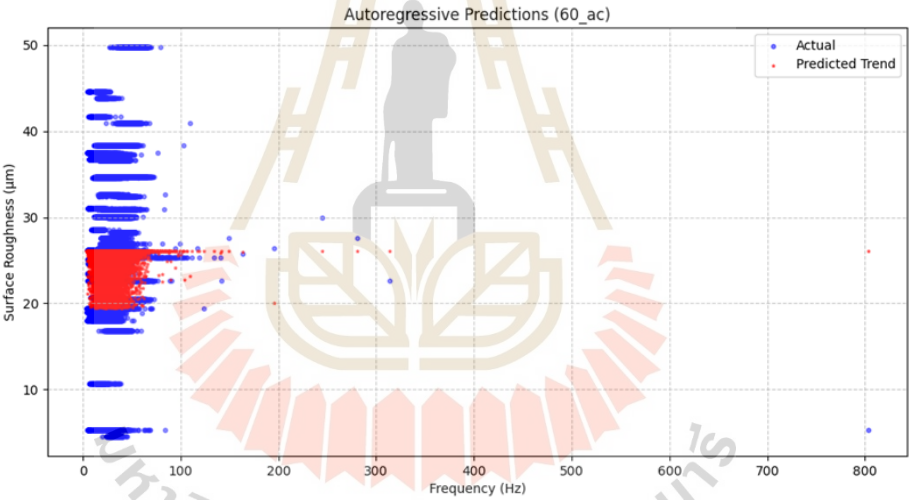
รูปที่ ข.35 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



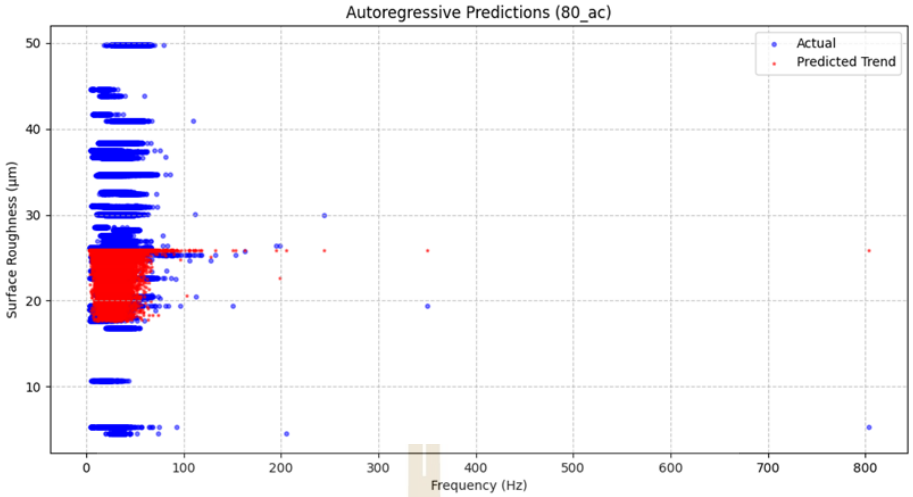
รูปที่ ข. 36 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



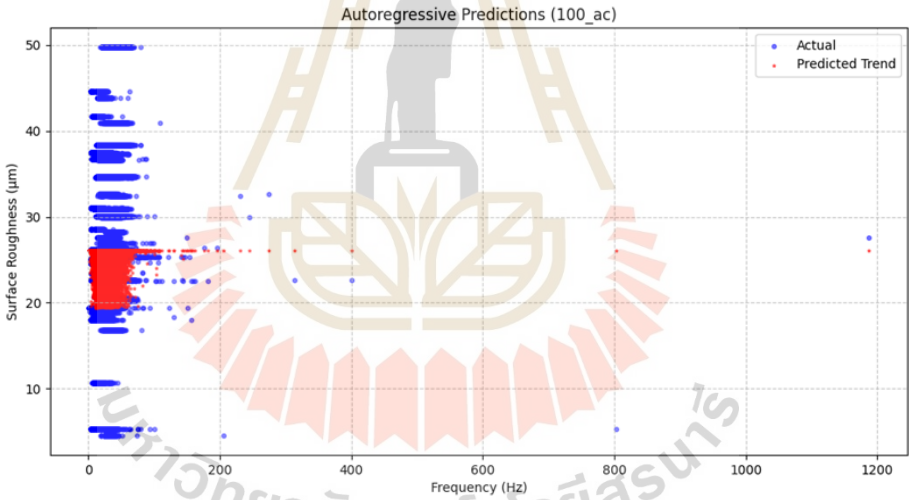
รูปที่ ข.37 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



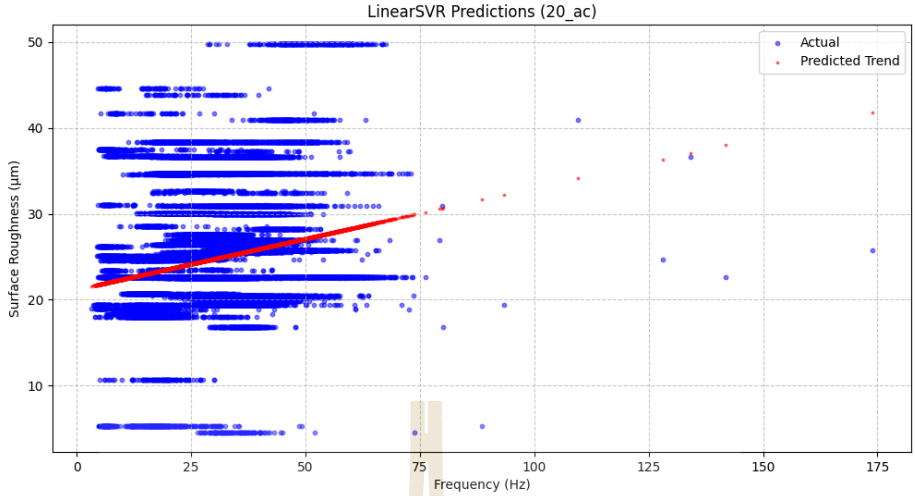
รูปที่ ข.38 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



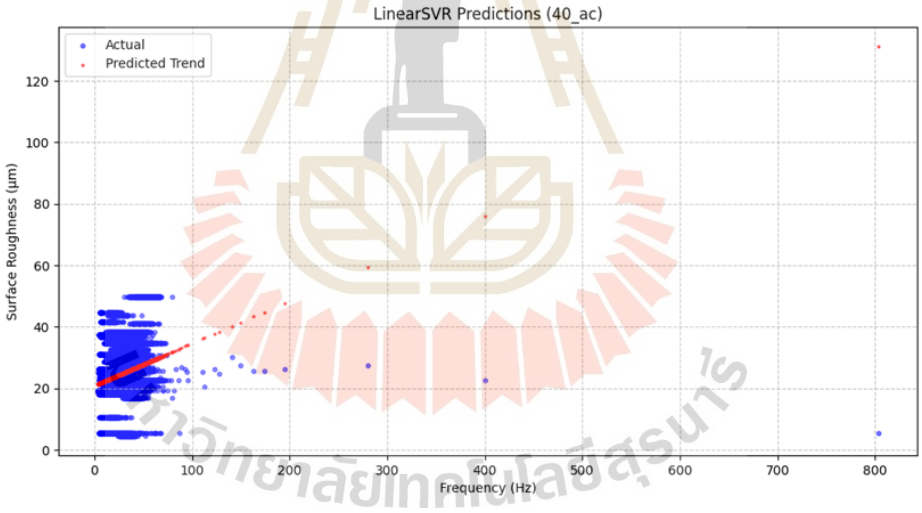
รูปที่ ข.39 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



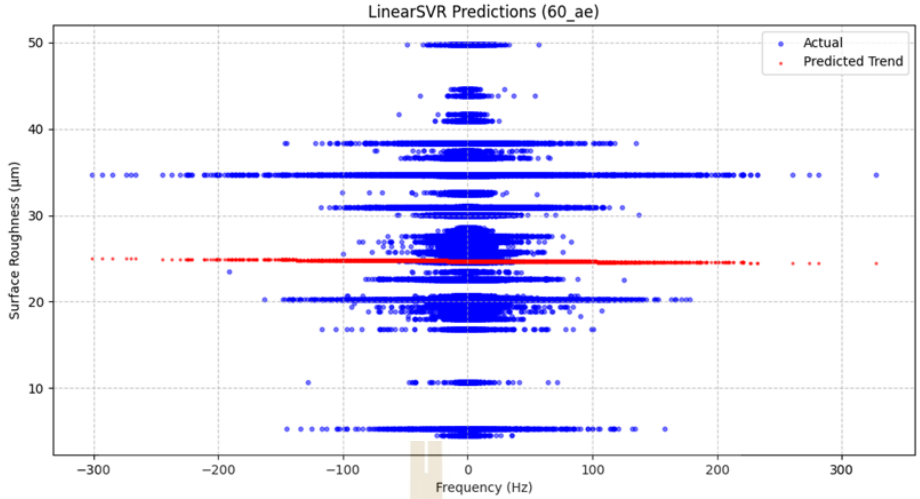
รูปที่ ข.40 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



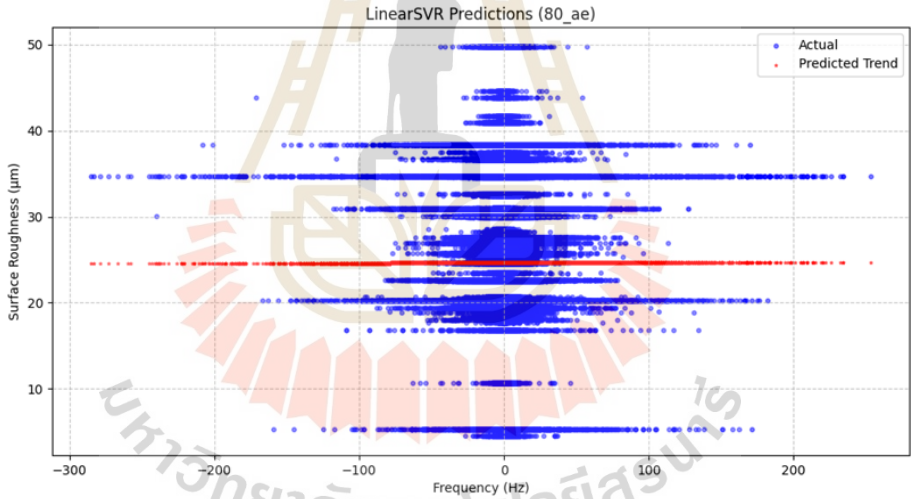
รูปที่ ข.41 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



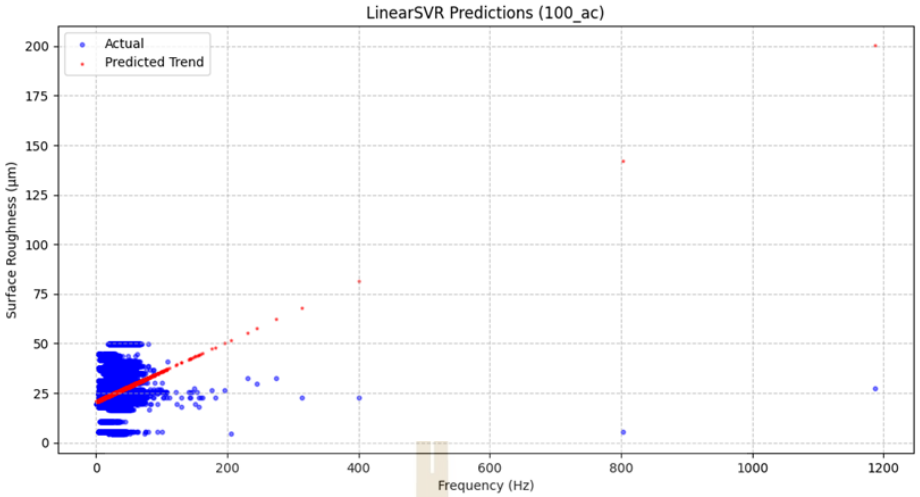
รูปที่ ข.42 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



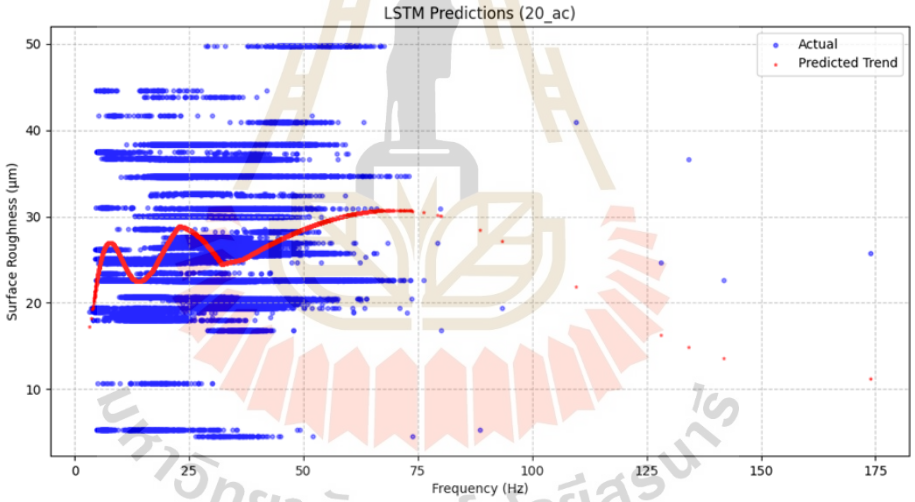
รูปที่ ข.43 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



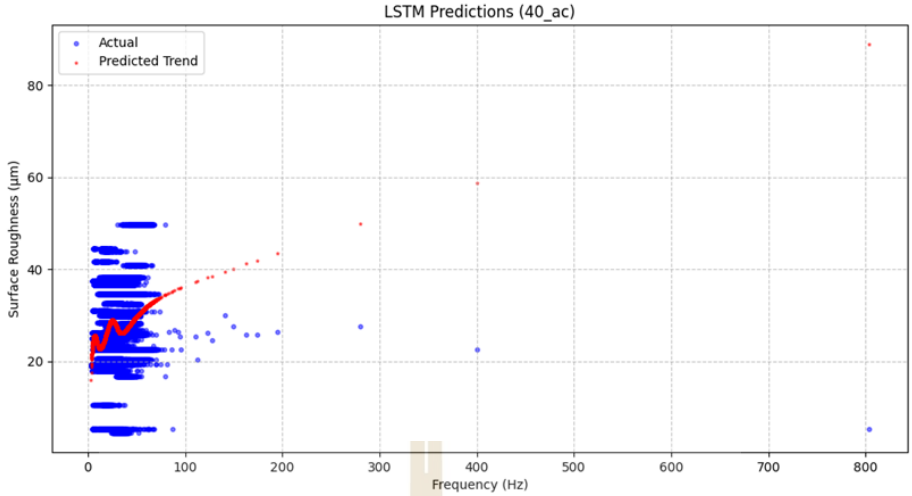
รูปที่ ข.44 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



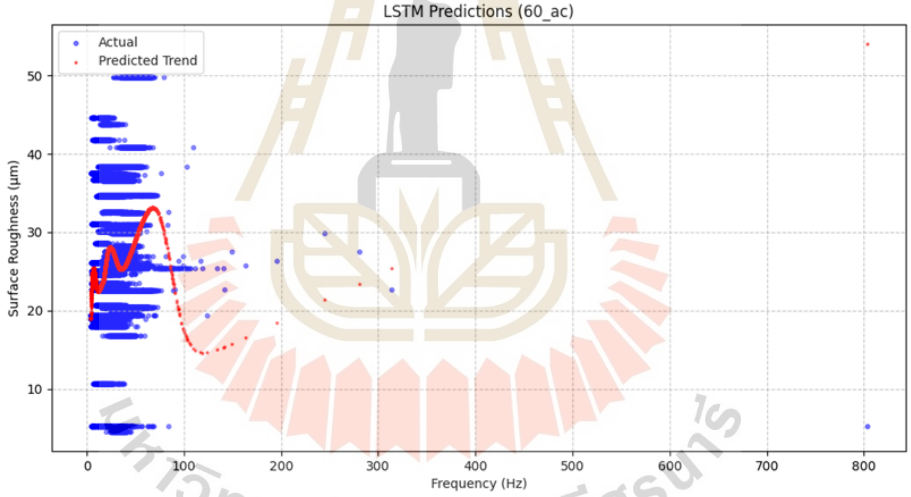
รูปที่ ข.45 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม Linear SVR



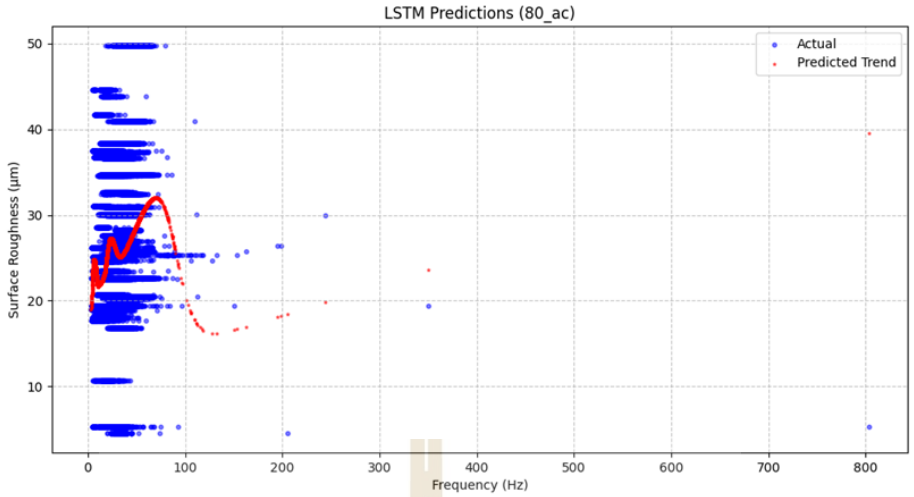
รูปที่ ข.46 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



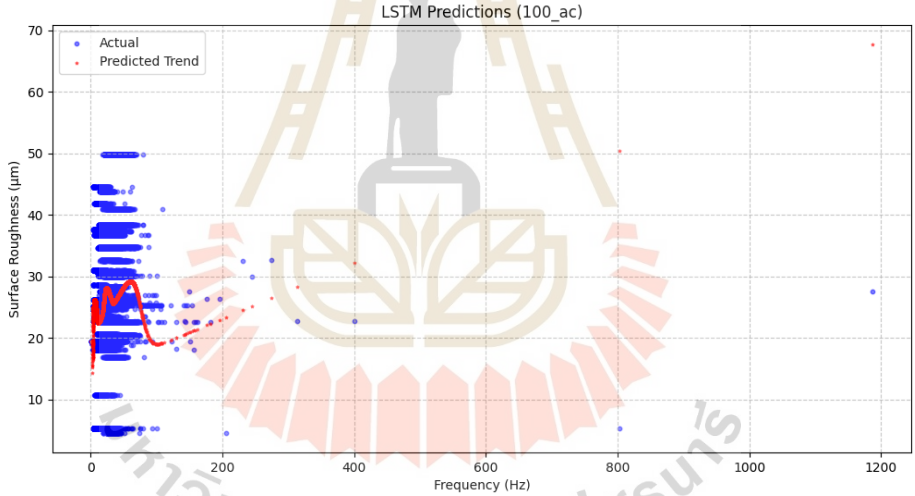
รูปที่ ข.47 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



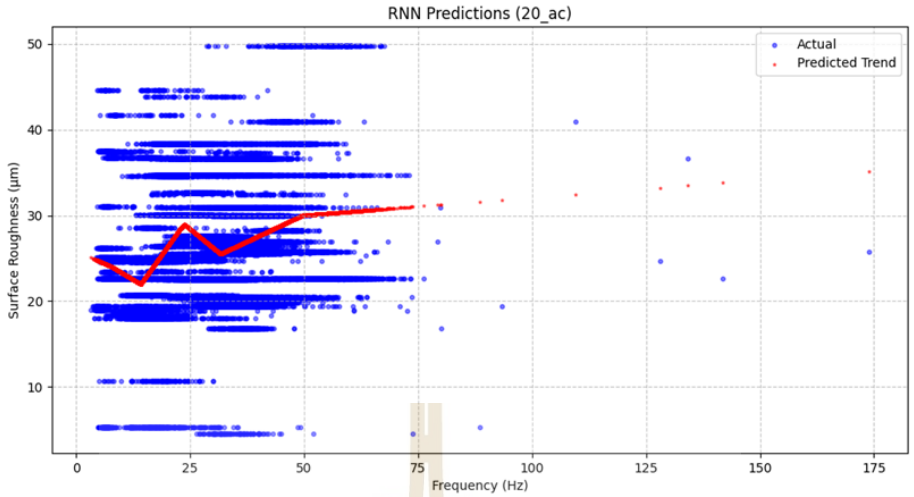
รูปที่ ข.48 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



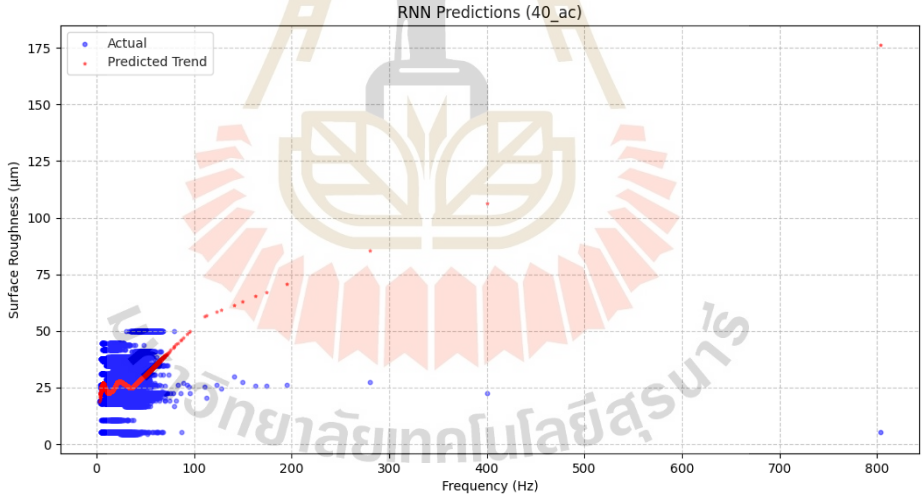
รูปที่ ข.49 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



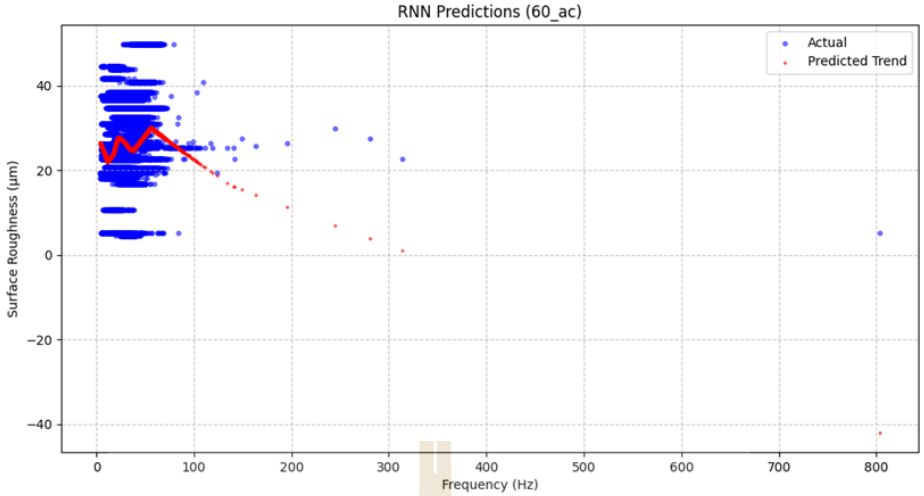
รูปที่ ข.50 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



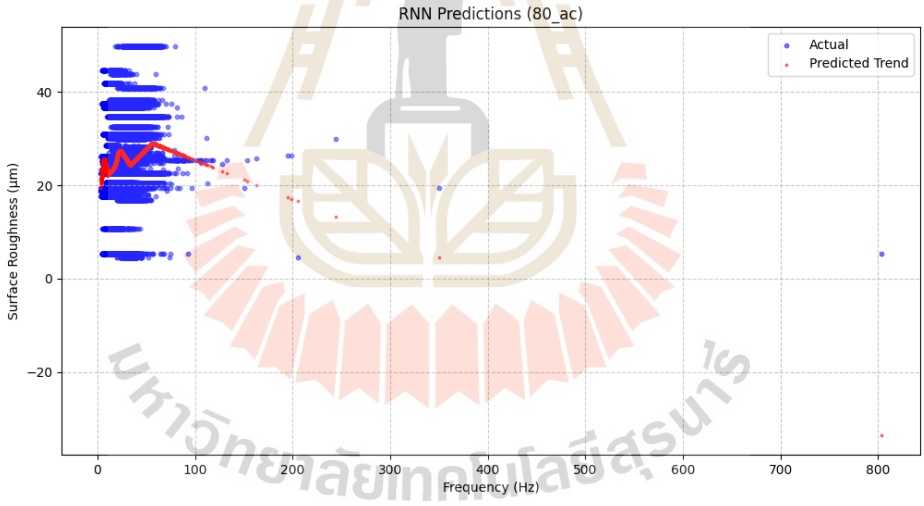
รูปที่ ข.51 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



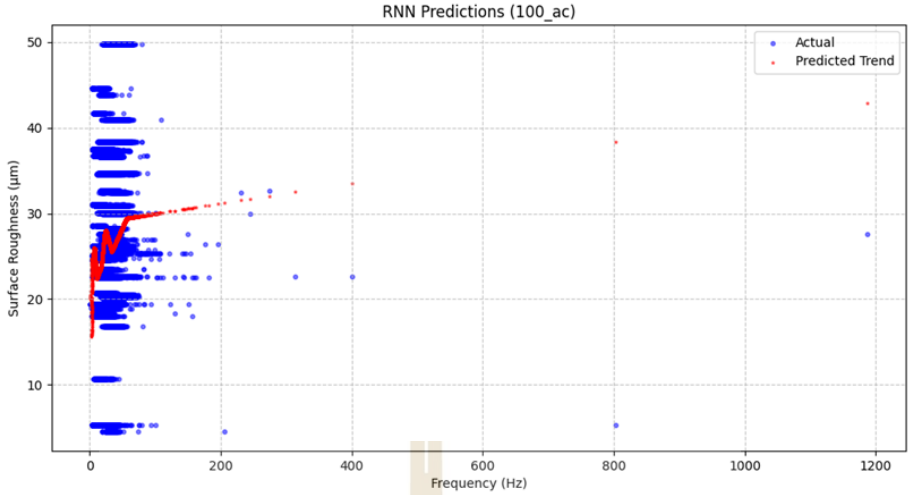
รูปที่ ข.52 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



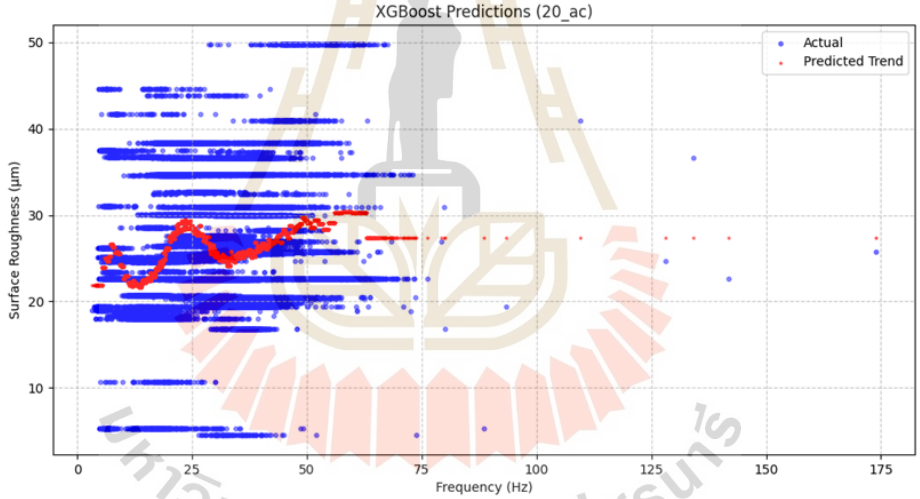
รูปที่ ข.53 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



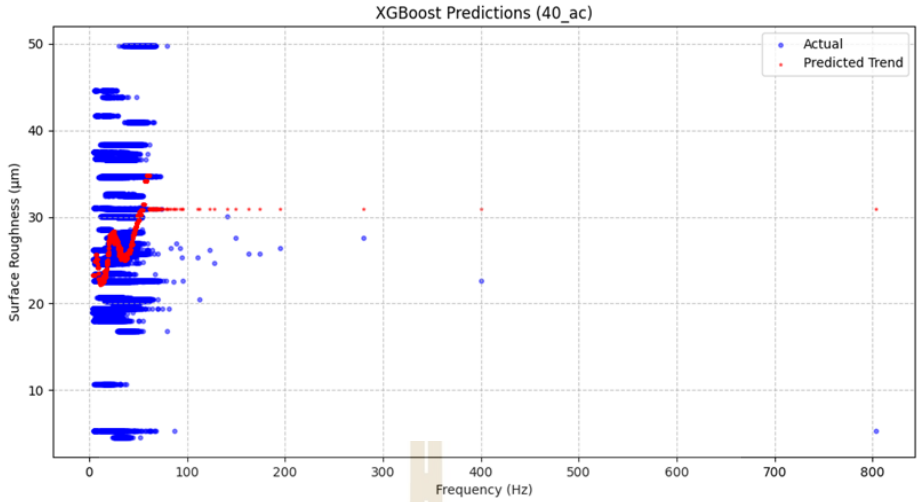
รูปที่ ข.54 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



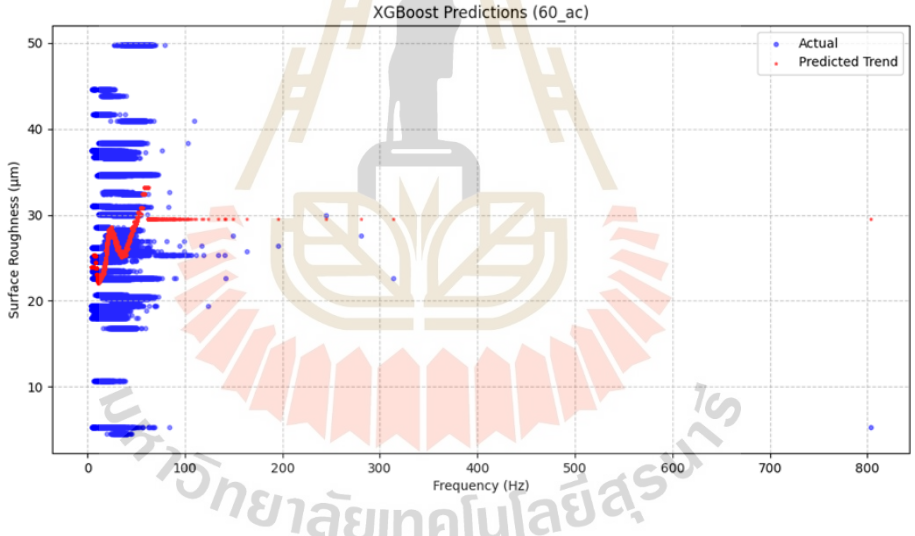
รูปที่ ข.55 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



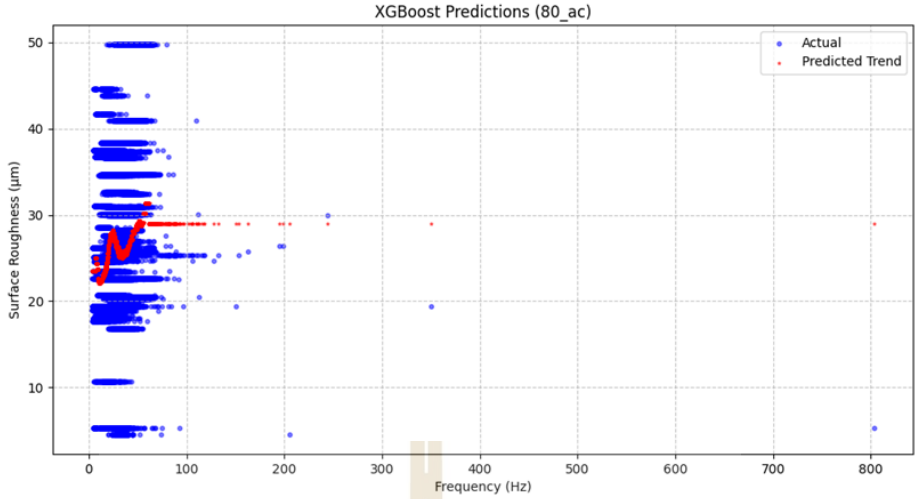
รูปที่ ข.56 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



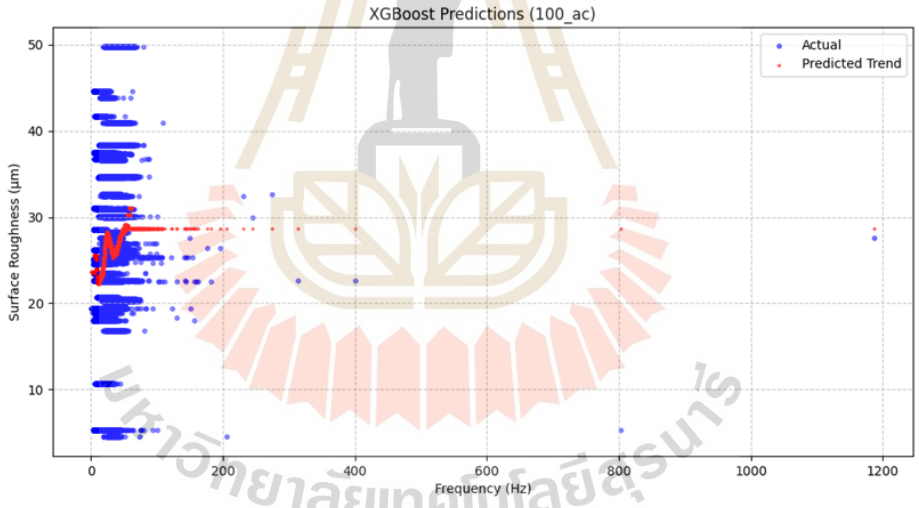
รูปที่ ข.57 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



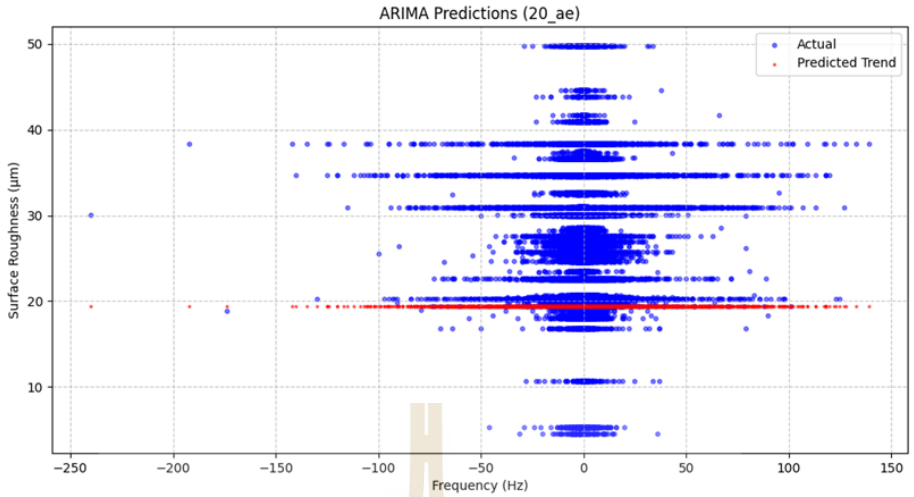
รูปที่ ข. 58. การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost.



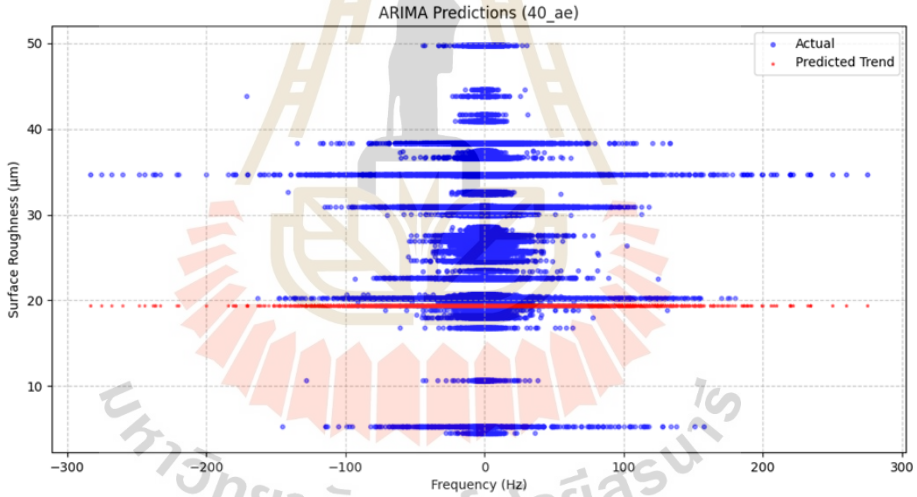
รูปที่ ข.59 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



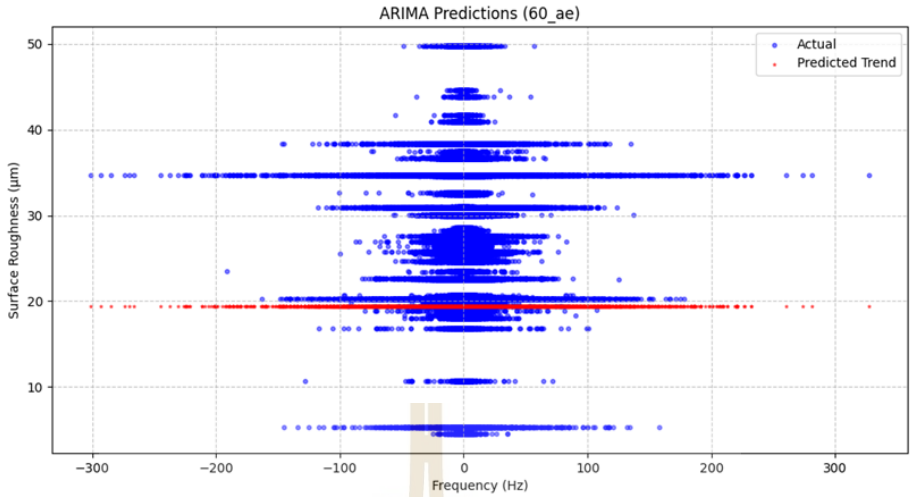
รูปที่ ข.60 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ Acceleration ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



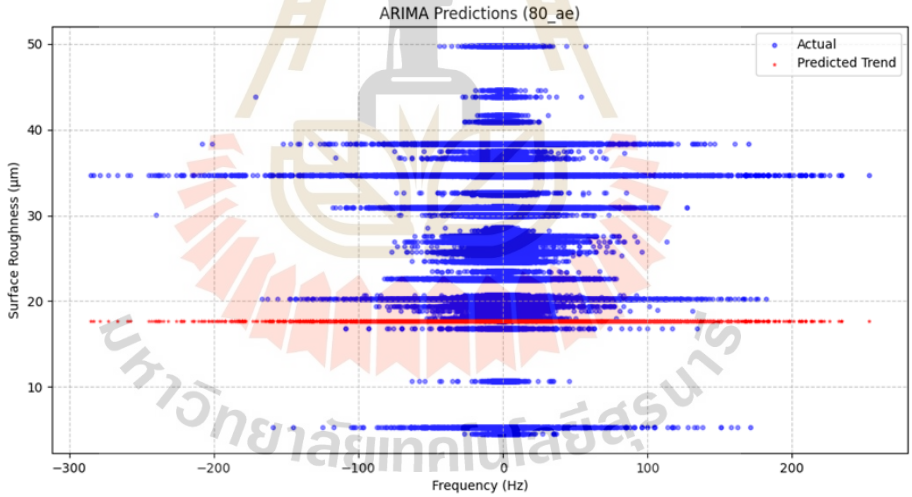
รูปที่ ข.61 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



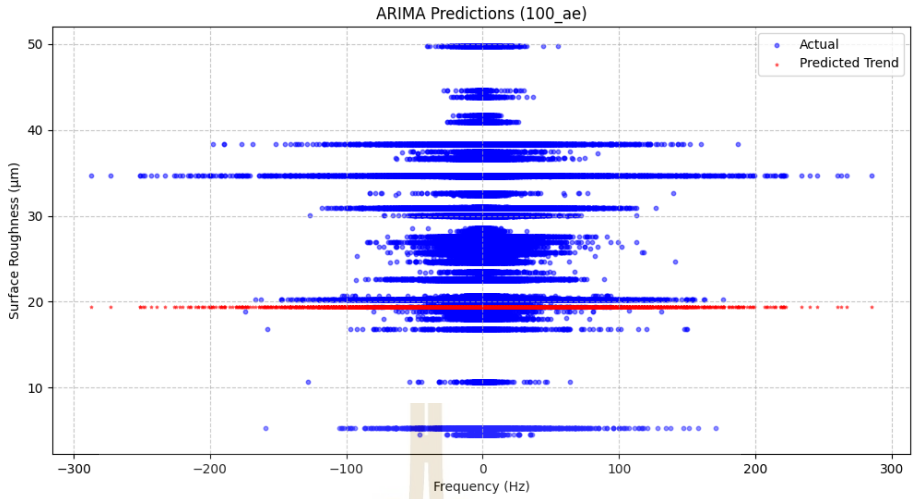
รูปที่ ข. 62 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



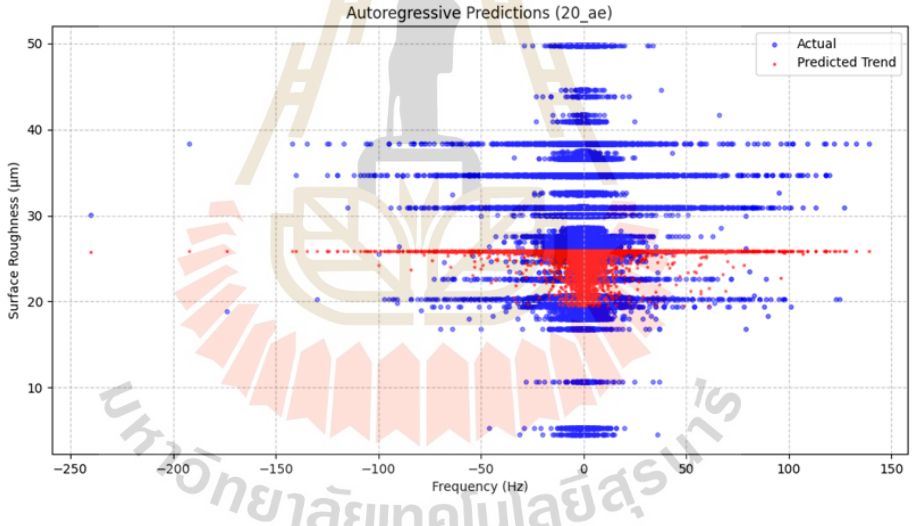
รูปที่ ข.63 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



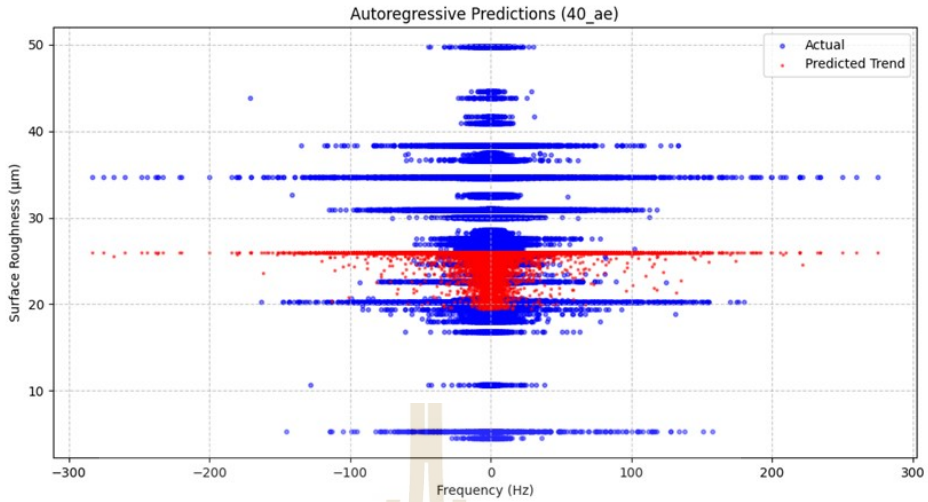
รูปที่ ข.64 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA



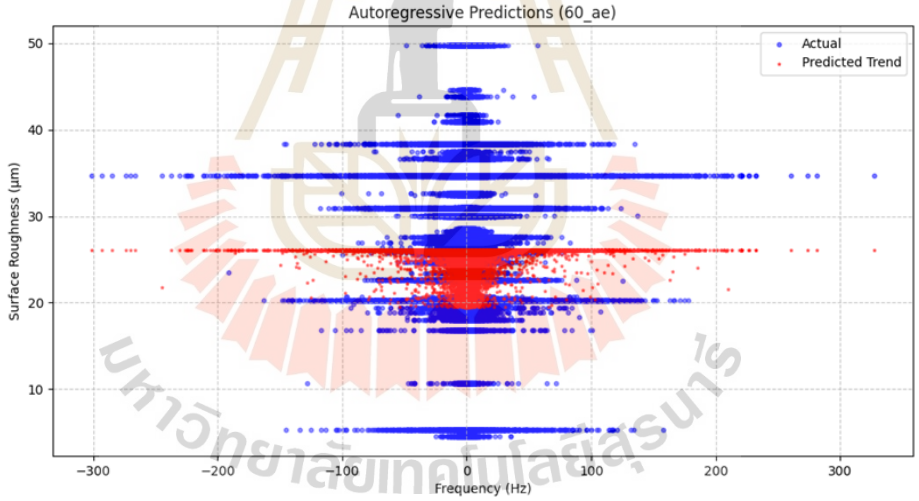
รูปที่ ข.65 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมดและใช้อัลกอริทึม ARIMA.



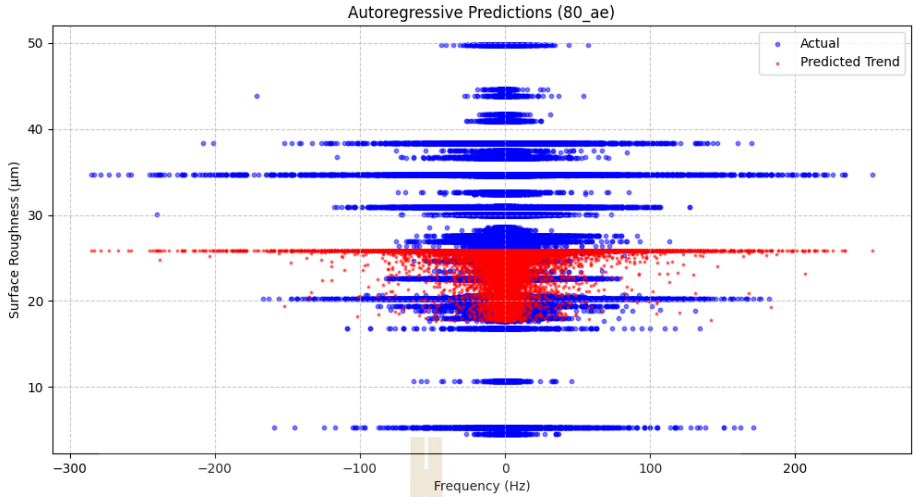
รูปที่ ข.66 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



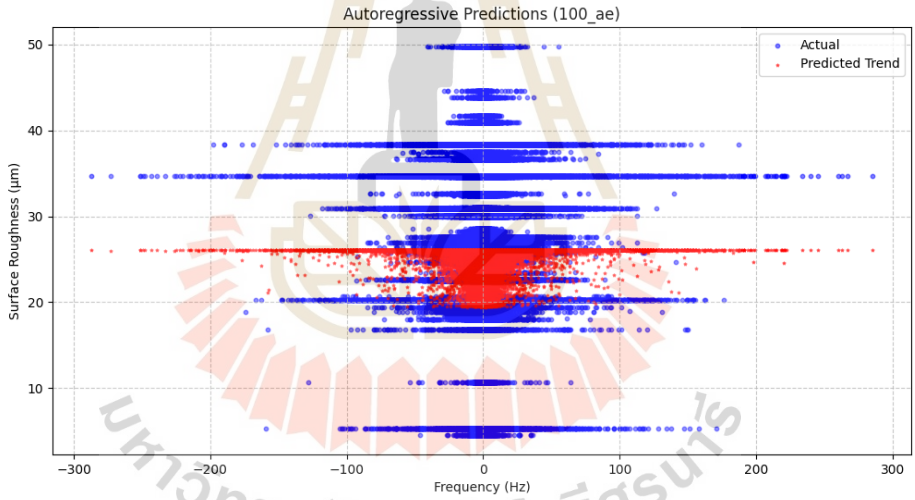
รูปที่ ข.67 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



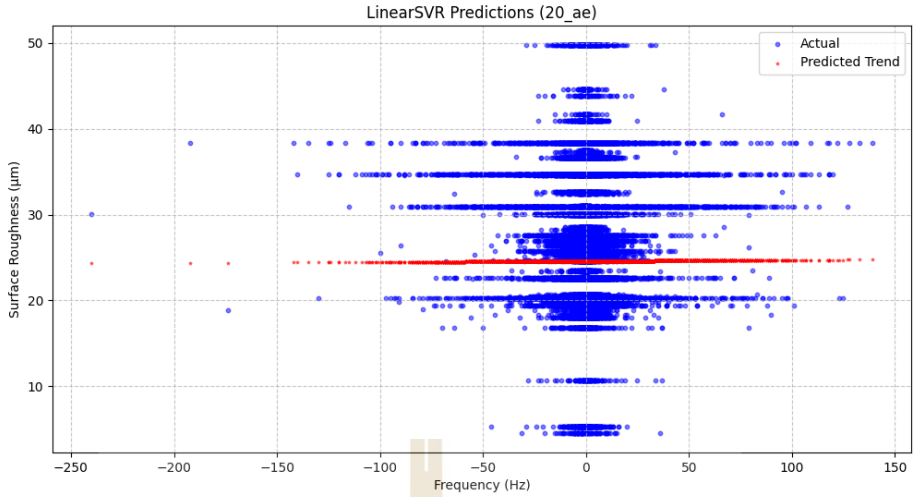
รูปที่ ข.68 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



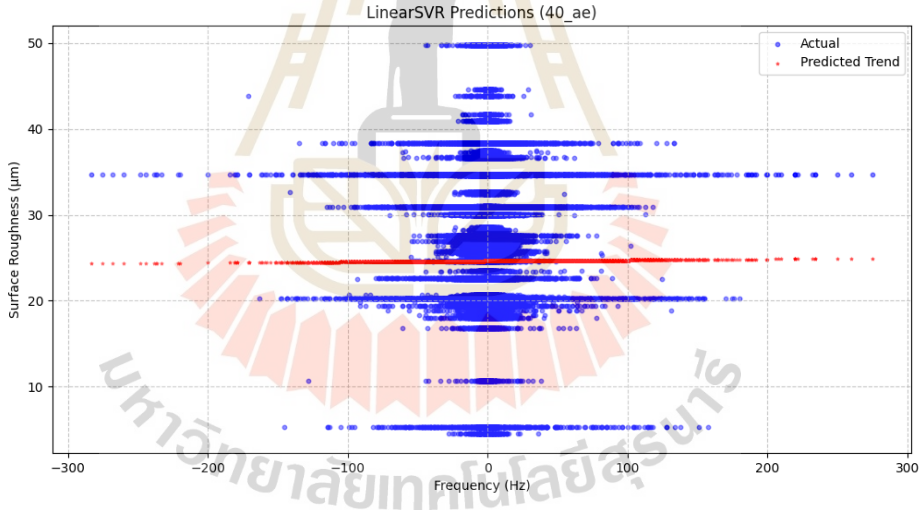
รูปที่ ข.69 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



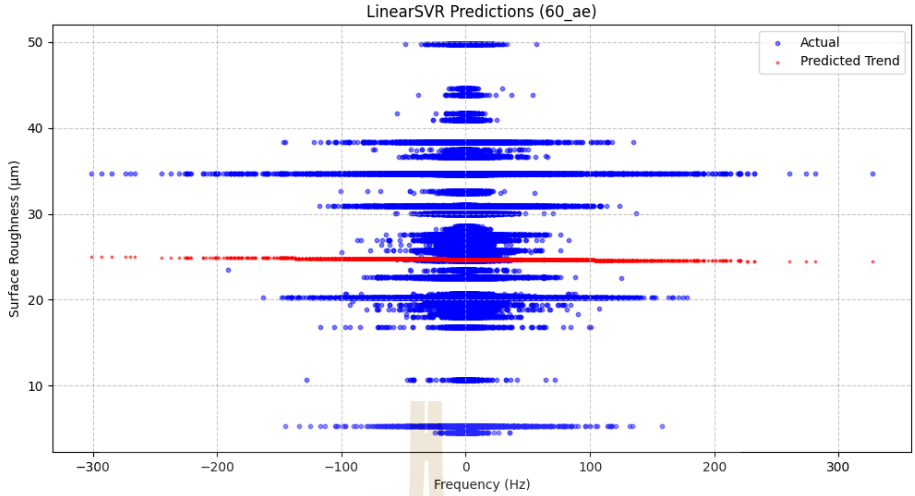
รูปที่ ข.70 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมดและใช้อัลกอริทึม Autoregressive



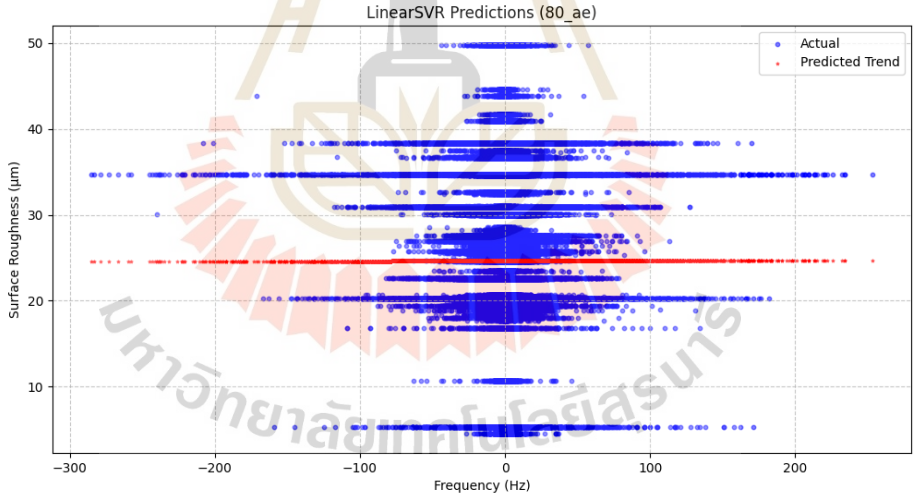
รูปที่ ข.71 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม LinearSVR



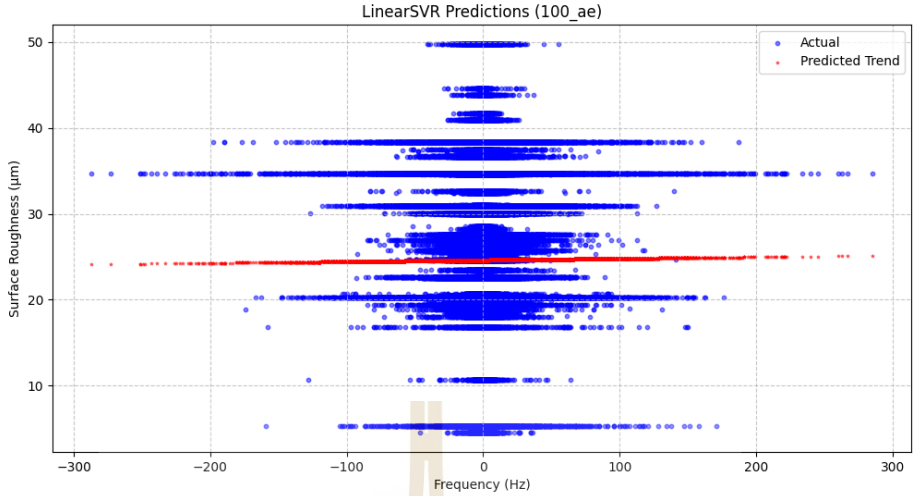
รูปที่ ข.72 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม LinearSVR



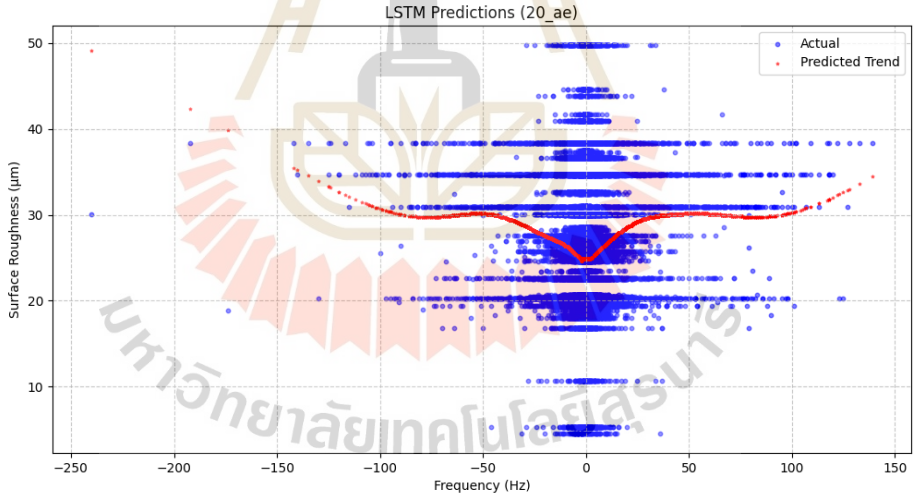
รูปที่ ข.73 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม LinearSVR



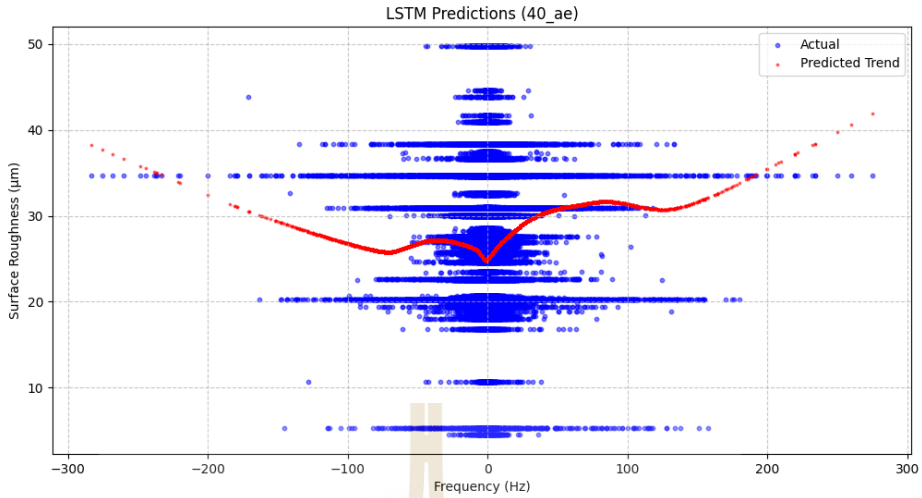
รูปที่ ข.74 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม LinearSVR



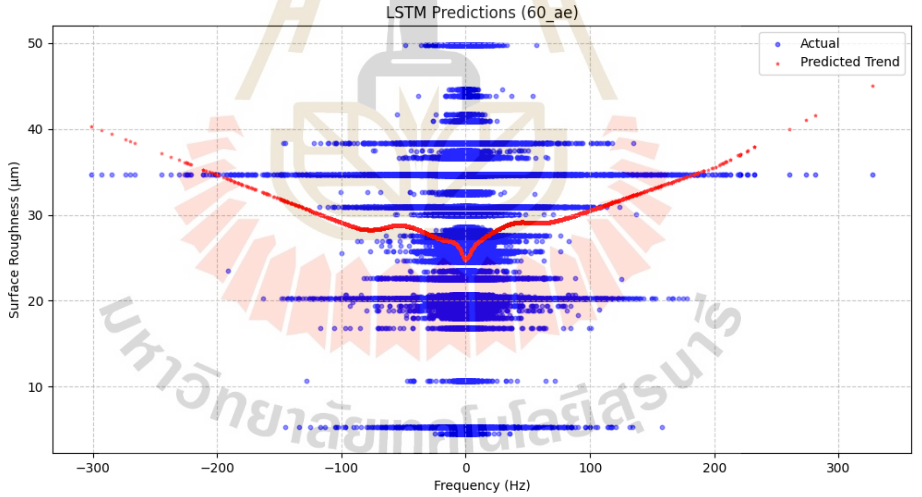
รูปที่ ข.75 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมด และใช้อัลกอริทึม LinearSVR



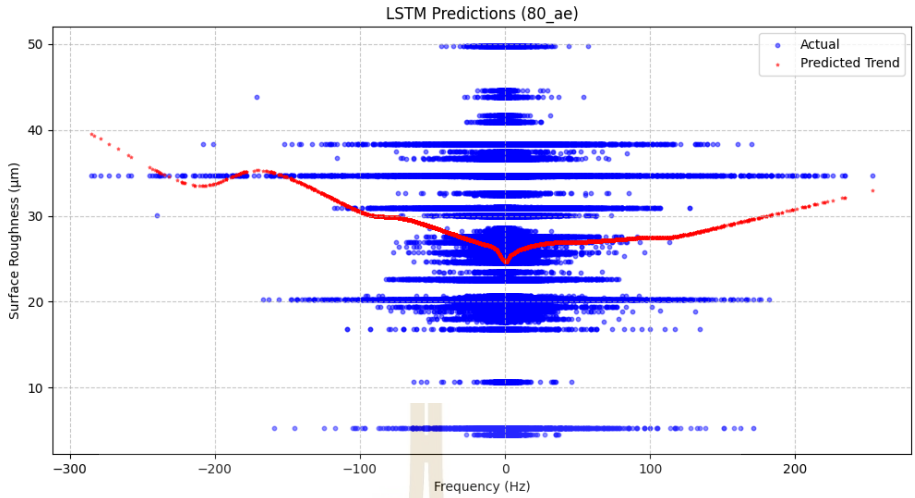
รูปที่ ข.76 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



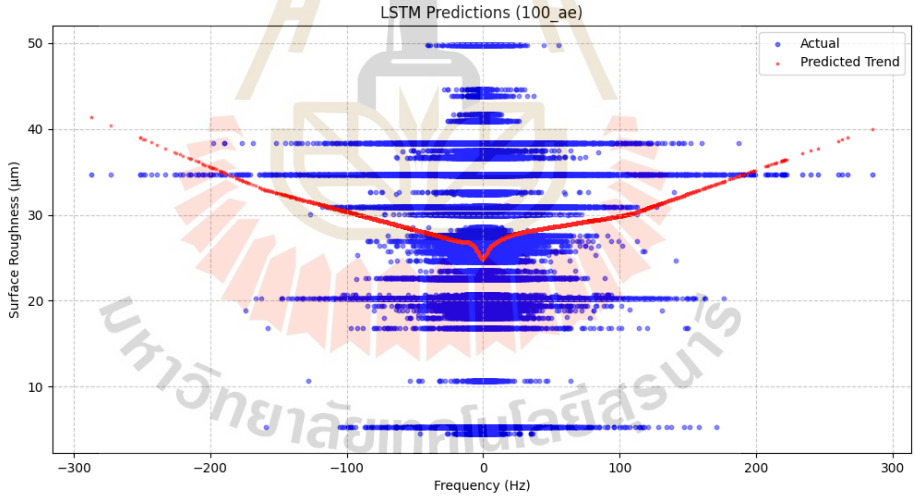
รูปที่ ข.77 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



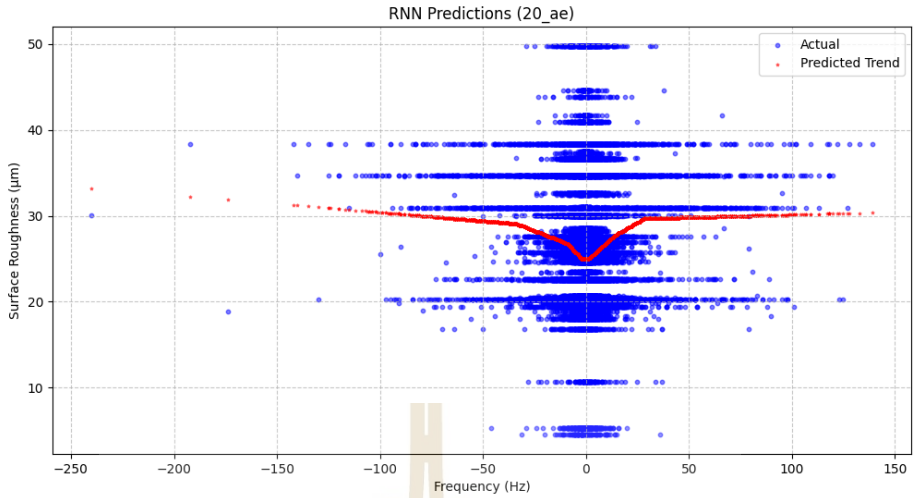
รูปที่ ข.78 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



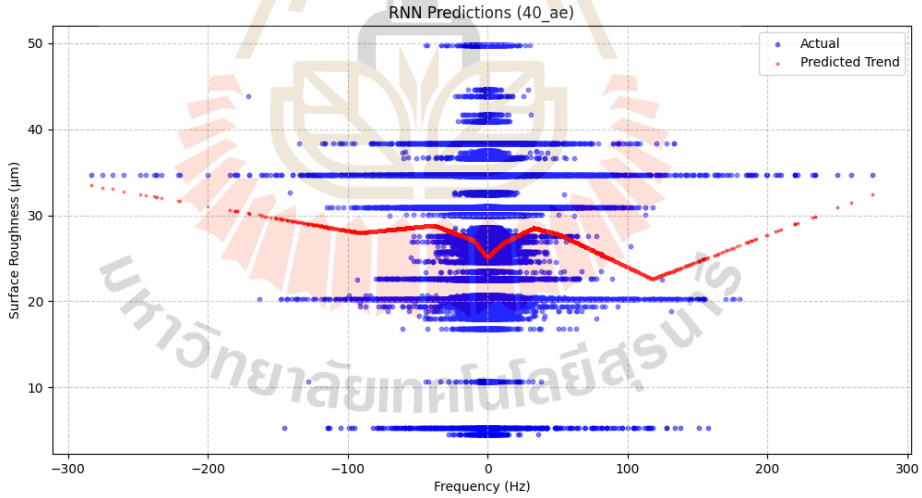
รูปที่ ข.79 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



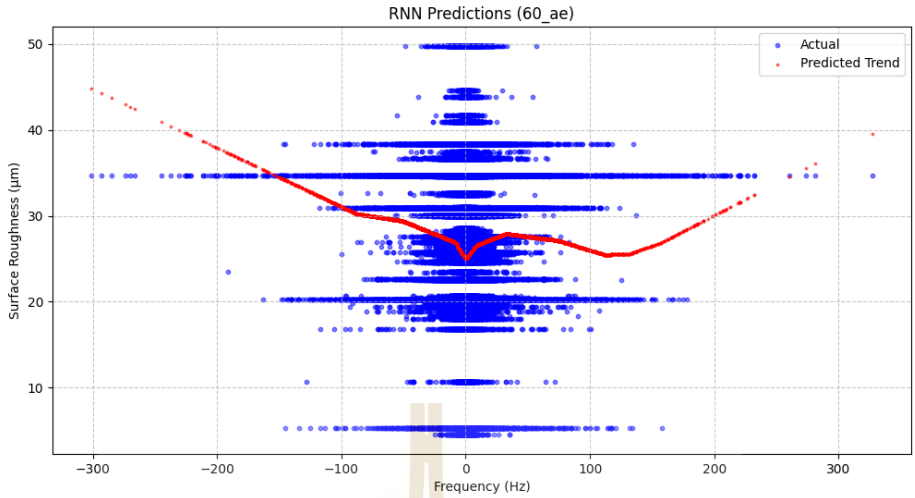
รูปที่ ข.80 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมดและใช้อัลกอริทึม LSTM



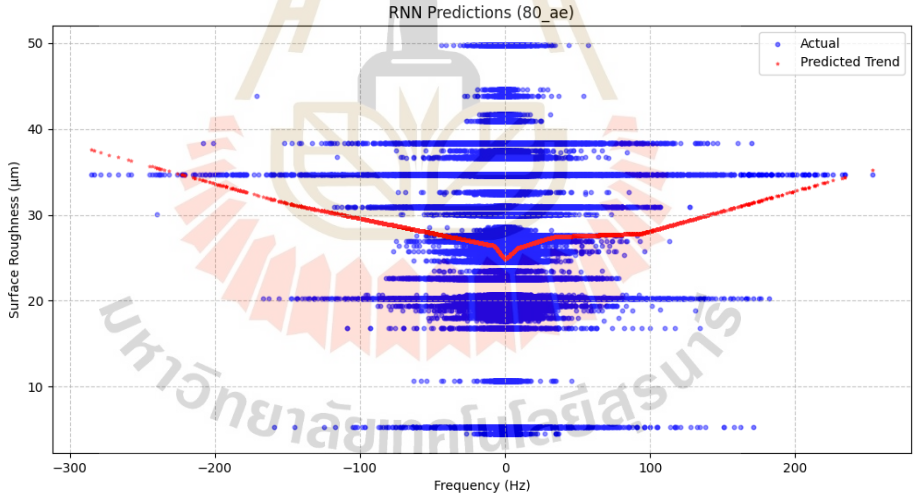
รูปที่ ข.81 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



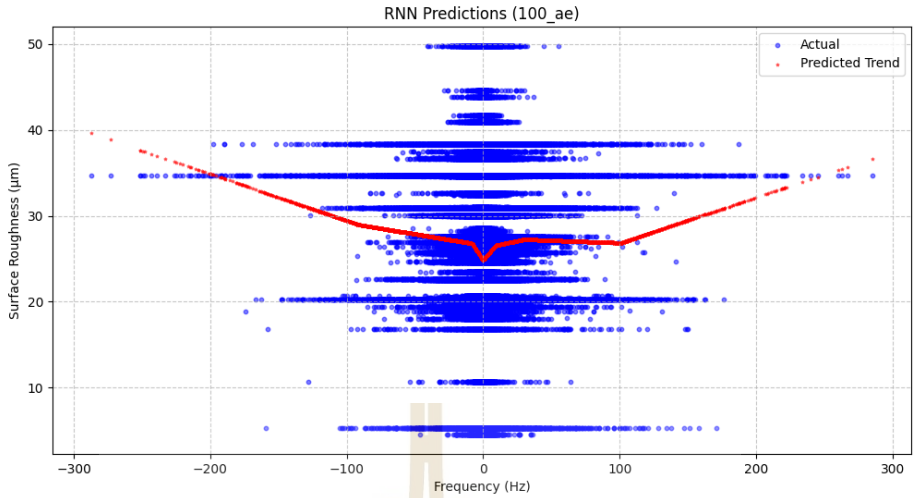
รูปที่ ข.82 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



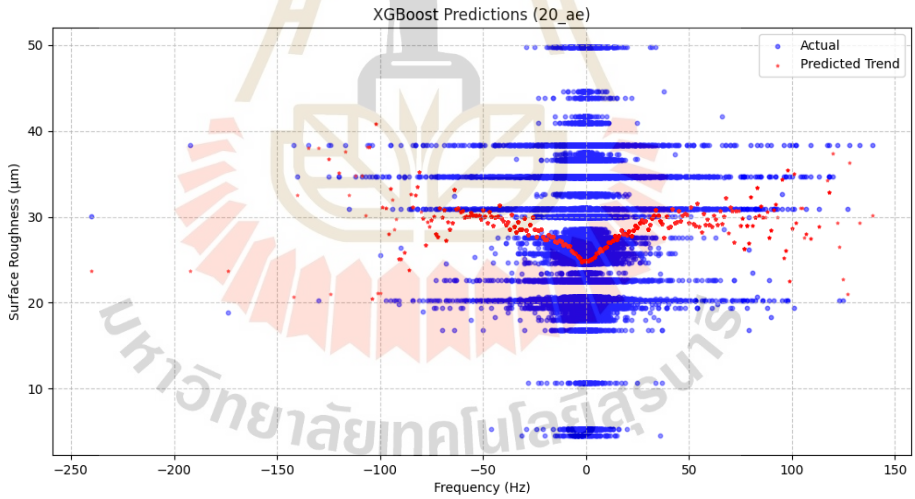
รูปที่ ข.83 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



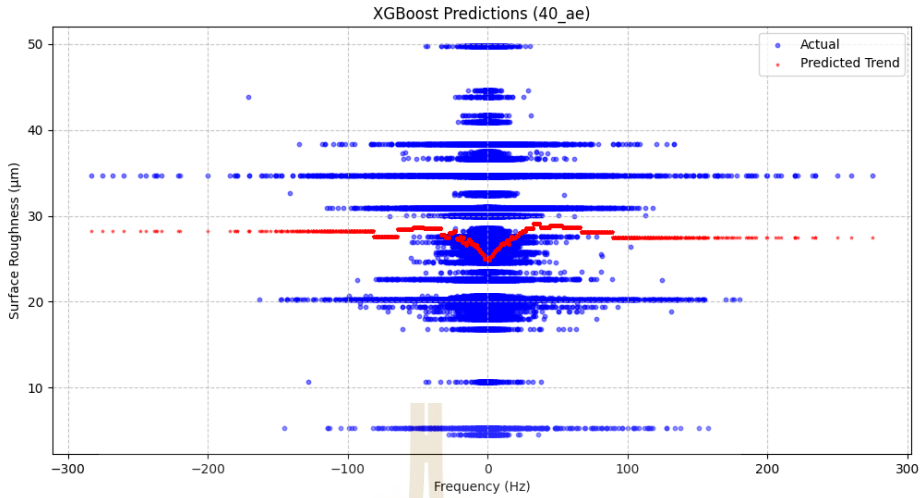
รูปที่ ข.84 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



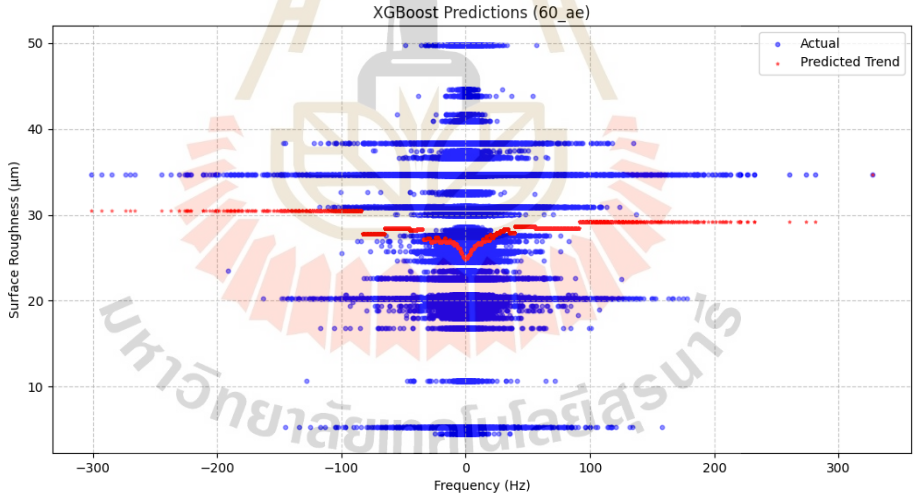
รูปที่ ข.85 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมดและใช้อัลกอริทึม RNN



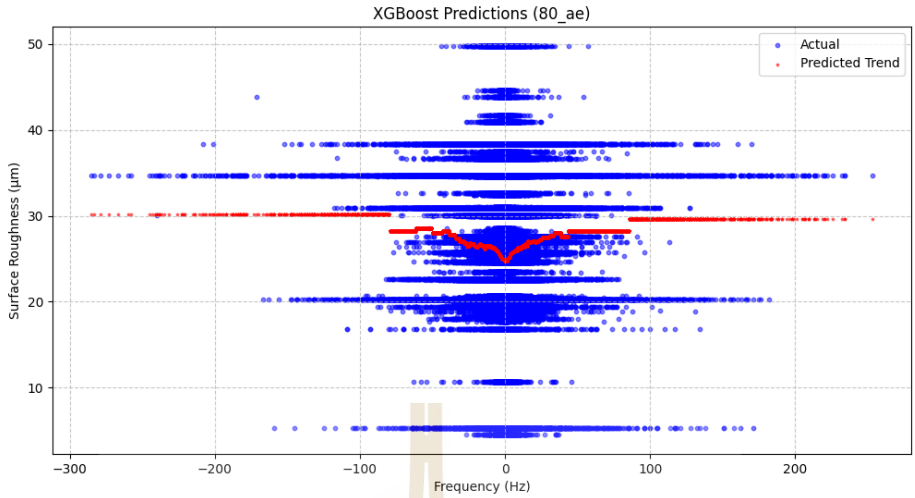
รูปที่ ข.86 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 20 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



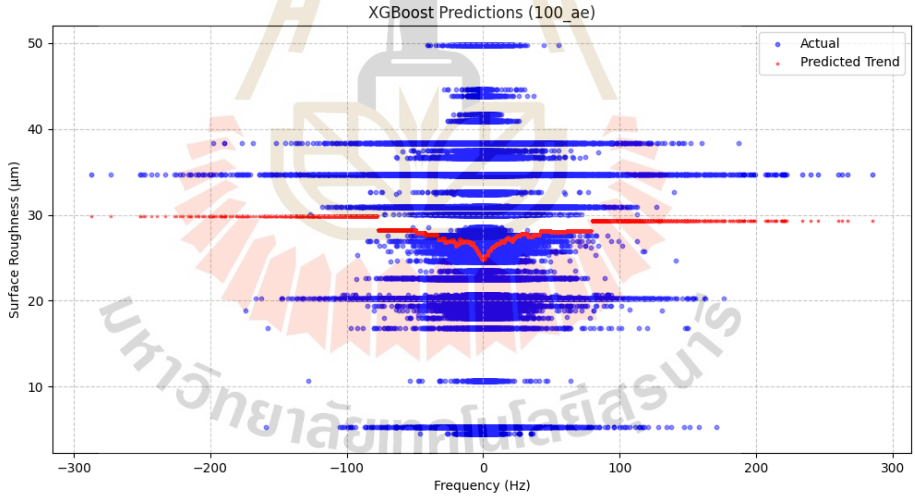
รูปที่ ข.87 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 40 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



รูปที่ ข.88 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 60 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



รูปที่ ข.89 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ร้อยละ 80 ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost



รูปที่ ข.90 การทำงานของโมเดลที่สร้างขึ้นด้วยการเก็บข้อมูลของ AE sensor ของเวลาทั้งหมดและใช้อัลกอริทึม XGBoost

ประวัติผู้เขียน

นายอภิวัฒน์ อินทร์ชู เกิดเมื่อวันที่ 6 ตุลาคม พ.ศ. 2540 ที่อำเภอเมืองจังหวัดนครราชสีมา เริ่มต้นการศึกษาในระดับอนุบาลศึกษาถึงประถมศึกษปีที่ 3 ที่โรงเรียนประสารวิทยา จังหวัดนครราชสีมา ได้เข้าศึกษาในระดับประถมศึกษาปีที่ 4 ตลอดจนสำเร็จการศึกษาในระดับมัธยมศึกษาสายวิทย์คณิต ที่โรงเรียนมารีย์วิทยา จังหวัดนครราชสีมา ได้สำเร็จการศึกษาวิศวกรรมศาสตร์บัณฑิต (วิศวกรรมการผลิตอัตโนมัติและหุ่นยนต์) สำนักวิชาวิศวกรรมศาสตร์มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2563 และในปีเดียวกันได้ศึกษาต่อในระดับวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิศวกรรมเครื่องกลและระบบกระบวนการ