การประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องโดยไม่ต้องเขียนโค้ด สำหรับการตรวจจับการหลอกลวงผ่านข้อความสั้น

นายนิวัฒน์ นาคจันทร์

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

การประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องโดยไม่ต้องเขียนโค้ด สำหรับการตรวจจับการหลอกลวงผ่านข้อความสั้น

นายนิวัฒน์ นาคจันทร์

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

ใบรับรองการค้นคว้าอิสระ

บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

เรื่อง การประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องโดยไม่ต้องเขียนโค้ด สำหรับการตรวจจับการหลอกลวงผ่านข้อความสั้น

โดย นายนิวัฒน์ นาคจันทร์

ได้รับอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ

คณบดีบัณฑิตวิทยาลัย / หัวหน้าภาควิชา

คณะกรรมการสอบการค้นคว้าอิสระ

..... ประธานกรรมการ

(รองศาสตราจารย์ ดร.สุมิตรา นวลมีศรี)

..... อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.สุนันฑา สดสี)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.นวพร วิสิฐพงศ์พันธ์)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุนันฑา สดสี)

ชื่อ : นายนิวัฒน์ นาคจันทร์
ชื่อการค้นคว้าอิสระ : การประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องโดยไม่ต้องเขียนโค้ด สำหรับการตรวจจับการหลอกลวงผ่านข้อความสั้น
สาขาวิชา : การบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก : ผู้ช่วยศาสตราจารย์ ดร.สุนันฑา สดสี
ปีการศึกษา : 2567

บทคัดย่อ

การศึกษานี้มีวัตถุประสงค์เพื่อประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องที่ไม่ต้องเขียนโค้ดและใช้โค้ดน้อยในการตรวจจับข้อความฟิชชิ่งผ่าน SMS โดยเริ่มต้นด้วยการแปลงข้อความSMS ให้เป็นการแทนข้อมูลในรูปแบบเวกเตอร์ จากนั้น Azure ML Studio ถูกนำมาใช้ในการดำเนินการศึกษานี้ โดยเลือกใช้โมเดลไม่ต้องเขียนโค้ดจำนวนสองอัลกอริทึม และโมเดลใช้โค้ดน้อยอีกสองสองอัลกอริทึม ได้แก่ LogisticRegression, MultinomialNaiveBayes, และ Multiclass Neural Network, Multiclass Logistic Regression ตามลำดับ ซึ่งโมเดลที่ไม่ต้องเขียนโค้ดจะถูกจัดอยู่ใน AutoML ในขณะที่โมเดลที่ใช้โค้ดน้อยจะถูกจัดอยู่ในDesigner สุดท้ายการจำลองการตรวจจับจะดำเนินการโดยการเปรียบเทียบประสิทธิภาพระหว่างโมเดลที่ไม่ต้องเขียนโค้ดและโมเดลที่ใช้โค้ดน้อย ผลลัพธ์ที่ได้จากการศึกษาชี้ให้เห็นว่าโมเดล LogisticRegression สามารถให้ค่าความแม่นยำ (Accuracy) ที่ร้อยละ 93 ความเที่ยงตรง (Precision) ที่ร้อยละ 93 และระยะเวลาในการคำนวณที่ 6 ชั่วโมง 15 นาที 6.276 วินาที จากผลการศึกษา สรุปได้ว่าโมเดล LogisticRegression เป็นโมเดลที่มีประสิทธิภาพและเหมาะสมในการใช้ตรวจจับ SMS ฟิชชิ่ง เหตุจากความแม่นยำ ความเที่ยงตรง และเวลาการดำเนินการ

(มีจำนวนทั้งสิ้น 47 หน้า)

คำสำคัญ : การเรียนรู้ของเครื่อง การหลอกลวงผ่านข้อความสั้น การตรวจจับ ไม่ต้องเขียนโค้ด

_____ อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก

Name : Mr. Niwat Nakchan
Independent Study Title : APPLYING NOCODE MACHINE LEARNING MODELS FOR DETECTING SMISHING
Major Field : Network and Information Security Management
King Mongkut's University of Technology North Bangkok
Independent Study Advisor : Assistant Professor Dr. Sunantha Sodsee
Academic Year : 2024

ABSTRACT

This study is aiming to apply no-code and low-code machine learning models to detect SMS phishing. First, the SMS is converted to vector representation. Next, Azure ML Studio is applied in this work by using two no-code models and two low-code models, which are LogisticRegression, MultinomialNaiveBayes, and Multiclass Neural Network, Multiclass Logistic Regression respectively. Herein, the no-code models are included in AutoML, as well as the low-code models are in the Designer. Finally, the detection simulations are conducted by comparing no-code and low-code models. The results show that the LogisticRegression model with Accuracy 93%, Precision 93% and 6h 15m 6.276s of compute duration time. It concludes that for detecting SMS Phishing, the LogisticRegression model is efficient and suitable for it because of accuracy, precision, and execution time

(Total 47 Pages)

Keywords: Machine Learning Smishing Detection NoCode

_____ Advisor

กิตติกรรมประกาศ

การค้นคว้าอิสระฉบับนี้สำเร็จไปได้ด้วยความกรุณาและการสนับสนุนกับผู้วิจัย ขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.สุนันทา สดสี อาจารย์ที่ปรึกษาหลักการค้นคว้าอิสระ ที่ให้คำแนะนำ องค์ความรู้ และข้อเสนอแนะที่เป็นประโยชน์ตลอดระยะเวลาการทำวิจัยฉบับนี้ ขอขอบคุณคณะอาจารย์ในสาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ ภาควิชาวิชาการบริหารเครือข่ายดิจิทัลและความมั่นคงปลอดภัยสารสนเทศ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือที่ให้ความรู้และสนับสนุนในด้านต่าง ๆ รวมถึงการให้โอกาสผู้วิจัยในการศึกษาค้นคว้าและพัฒนาวิจัยนี้ ขอแสดงความขอบคุณไปยังเพื่อนร่วมงานและผู้ที่ให้ความช่วยเหลือในการให้คำปรึกษาและกำลังใจในช่วงเวลาต่าง ๆ ของการศึกษา สุดท้ายนี้ผู้วิจัยขอขอบคุณ บิดา มารดา และครอบครัว ที่ให้การสนับสนุนและเป็นกำลังใจที่สำคัญที่สุดเสมอมา ที่ได้ให้กำลังใจแก่ผู้วิจัยเสมอมาจนทำการค้นคว้าอิสระฉบับนี้สำเร็จไปได้ด้วยดี

นิวัฒน์ นาคจันทร์

สารบัญ

หน้า

สารบัญ

สารบัญตาราง

สารบัญรูปภาพ (ถ้ามี)

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบัน เทคโนโลยีมีบทบาทสำคัญและถูกนำมาใช้งานอย่างแพร่หลายในชีวิตประจำวัน โดยช่วยเพิ่มความสะดวกสบายและเสริมประสิทธิภาพในการดำเนินกิจกรรมต่างๆ ตามรายงาน Digital 2022 Global Overview ประจำเดือนมกราคม พ.ศ. 2565 พบว่าจำนวนผู้ใช้อินเทอร์เน็ตทั่วโลกเพิ่มขึ้นเป็น 4.95 พันล้านคน คิดเป็นร้อยละ 62.5 ของประชากรทั้งหมดทั่วโลก โดยมีอัตราการเติบโต 192 ล้านคน หรือเพิ่มขึ้นร้อยละ 4.0 ภายในปีที่ผ่านมา นอกจากนี้ รายงานจากสถาบัน IMC ยังระบุว่า ประเทศไทยติดอันดับต้นๆ ของโลกในด้านการใช้เทคโนโลยีดิจิทัล โดยมีอัตราส่วนการเข้าถึงอินเทอร์เน็ตอยู่ที่ร้อยละ 77.8 ของประชากรทั้งหมด และใช้เวลาในการเล่นอินเทอร์เน็ตเฉลี่ย 9 ชั่วโมง 6 นาทีต่อวัน ซึ่งจัดอยู่ในอันดับที่ 7 ของโลก หนึ่งในเทคโนโลยีที่มีบทบาทสำคัญและเปรียบเสมือนปัจจัยที่ห้าของมนุษย์ คือ เทคโนโลยีโทรศัพท์เคลื่อนที่ [1]

เทคโนโลยีโทรศัพท์เคลื่อนที่ (Mobile Technology) [2] ได้กลายเป็นส่วนหนึ่งของชีวิตประจำวัน โดยมีการใช้งานอย่างแพร่หลายในระดับโลก จากรายงานของ Ovum พบว่าร้อยละ 73 ของประชากรในประเทศที่พัฒนาแล้วและกำลังพัฒนาครอบครองโทรศัพท์เคลื่อนที่แบบสมาร์ตโฟน นอกจากนี้ ในภูมิภาคอเมริกาเหนือร้อยละ 55 ของการโทรเข้า Call Center มาจากโทรศัพท์เคลื่อนที่ ซึ่งสะท้อนให้เห็นว่า สมาร์ตโฟนได้กลายเป็นอุปกรณ์สำคัญที่ขาดไม่ได้ ในยุคดิจิทัล

ข้อได้เปรียบของเทคโนโลยีโทรศัพท์เคลื่อนที่คือ การพกพาและการเชื่อมต่อที่หลากหลาย ผู้ใช้สามารถติดต่อกันได้ไม่เพียงแค่ผ่านการโทรด้วยเสียง แต่ยังรวมถึง ข้อความสั้น (SMS), การสนทนาออนไลน์ (Chat), อีเมล (Email), สื่อสังคมออนไลน์ (Social Media) และแอปพลิเคชันบนอุปกรณ์เคลื่อนที่ (Mobile Applications) นอกจากนี้ สมาร์ตโฟนและแท็บเล็ต ยังช่วยขับเคลื่อนการใช้งานเทคโนโลยีสารสนเทศในหมู่ผู้บริโภคอย่างมหาศาล โดยกลายเป็นเครื่องมือหลักในการเข้าถึงข้อมูลและข่าวสารผ่านเครือข่ายอินเทอร์เน็ต

การเติบโตของ Mobile Technology ยังมีส่วนช่วยลดช่องว่างทางดิจิทัล (Digital Divide) ทำให้ผู้คนทุกเพศทุกวัยสามารถเข้าถึงอินเทอร์เน็ตและซอฟต์แวร์ต่างๆ ได้สะดวกขึ้น นอกจากนี้ ยังช่วยเพิ่มศักยภาพในการทำงานและการดำเนินธุรกิจจากระยะไกล (Remote Work & Business Management) [3] ผ่านสมาร์ตโฟนและแท็บเล็ตที่เชื่อมต่ออินเทอร์เน็ต ซึ่งสามารถรองรับการใช้งานได้ทั้งในเชิงธุรกิจและชีวิตประจำวัน

อย่างไรก็ตาม การใช้งานเทคโนโลยีโทรศัพท์เคลื่อนที่ที่เพิ่มสูงขึ้น นำไปสู่ความเสี่ยงด้านความปลอดภัย ทางไซเบอร์ที่เพิ่มขึ้นตามไปด้วย ปัจจุบันพบว่าภัยคุกคามทางไซเบอร์มีหลายรูปแบบ

โดยหนึ่งในภัยคุกคามที่แพร่หลายคือ การหลอกลวงผ่านข้อความสั้น (Smishing) ซึ่งมักอยู่ในรูปแบบของข้อความที่มีลิงก์อันตรายเพื่อหลอกลวงให้เหยื่อคลิกเข้าไป ซึ่งอาจนำไปสู่การโจรกรรมข้อมูลส่วนบุคคล หรือการดาวน์โหลดมัลแวร์โดยไม่รู้ตัว

การศึกษาพบว่า อัลกอริทึมการเรียนรู้ของเครื่อง (Machine Learning: ML) [4] สามารถนำมาใช้ในการตรวจจับความผิดปกติของข้อความและลิงก์ที่อาจเป็นภัยคุกคามได้อย่างมีประสิทธิภาพ งานวิจัยในด้านนี้มุ่งเน้นไปที่ การวิเคราะห์ข้อมูลกิจกรรมเครือข่ายและพฤติกรรมของแอปพลิเคชัน เพื่อช่วยตัดสินใจก่อนที่ผู้ใช้จะดำเนินการกับข้อความหรือลิงก์ที่ได้รับ

ระบบที่ได้รับการพัฒนาอาจอยู่ในรูปแบบของ ระบบแจ้งเตือนภัยคุกคามทางไซเบอร์ (Cyber Threat Alert System) ซึ่งสามารถวิเคราะห์ความเสี่ยงได้โดยอัตโนมัติ โดยใช้ทรัพยากรของอุปกรณ์ในปริมาณต่ำ เพื่อลดภาระการประมวลผลและเพิ่มความรวดเร็วในการตรวจจับภัยคุกคาม นอกจากนี้ ระบบดังกล่าวยังสามารถเป็นแนวทางสำหรับนักพัฒนาและนักวิจัยที่สนใจศึกษาเพิ่มเติมเกี่ยวกับ การประยุกต์ใช้ Machine Learning เพื่อเสริมสร้างความปลอดภัยบนอุปกรณ์เคลื่อนที่

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 เพื่อสร้างแบบจำลอง สำหรับการตรวจจับภัยคุกคามทางไซเบอร์ทางข้อความบนโทรศัพท์เคลื่อนที่ได้อย่างอัตโนมัติ

1.2.2 เพื่อประเมินประสิทธิภาพโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ด สำหรับตรวจจับภัยคุกคามทางไซเบอร์ทางข้อความบนโทรศัพท์เคลื่อนที่

1.3 ขอบเขตการวิจัย

1.3.1 ศึกษาโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ด เพื่อใช้สำหรับตรวจจับภัยคุกคามทางไซเบอร์ในรูปแบบของ Smishing (SMS Phishing)

1.3.2 พัฒนาโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ด เพื่อใช้สำหรับตรวจจับภัยคุกคามทางไซเบอร์ในรูปแบบของ Smishing (SMS Phishing)

1.3.3 เปลี่ยนแปลงรูปแบบข้อความเป็นเวกเตอร์เพื่อใช้ใน Azure Machine Learning Designer

1.3.4 ทดสอบโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ด เพื่อใช้สำหรับตรวจจับภัยคุกคามทางไซเบอร์ในรูปแบบของ Smishing (SMS Phishing)

1.3.5 เปรียบเทียบโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ดที่ใช้สำหรับตรวจจับภัยคุกคามทางไซเบอร์ในรูปแบบของ Smishing (SMS Phishing)

1.4 ประโยชน์ของการวิจัย

1.4.1 ประเมินประสิทธิภาพโมเดลการเรียนรู้ของเครื่องแบบไม่ต้องเขียนโค้ด เพื่อใช้สำหรับตรวจจับภัยคุกคามทางไซเบอร์ในรูปแบบของ Smishing (SMS Phishing)

1.4.2 แปลงข้อความเป็นเวกเตอร์เพื่อให้โมเดลการเรียนรู้ของเครื่องเข้าใจและประมวลผลข้อมูลที่เป็นข้อความได้ในรูปแบบที่คำนวณได้

1.4.2 รูปแบบโมเดลที่ช่วยในการตรวจจับการหลอกลวงผ่านข้อความสั้น

บทที่ 2
เอกสารและงานวิจัยที่เกี่ยวข้อง

งานวิจัยนี้มีวัตถุประสงค์เพื่อสร้างแบบจำลอง สำหรับการตรวจจับภัยคุกคามทางไซเบอร์ทางข้อความบนโทรศัพท์เคลื่อนที่ได้อย่างอัตโนมัติและเพื่อเป็นแนวทางในการป้องกัน Smishing (SMS Phishing) หรือ การตรวจจับภัยคุกคามทางไซเบอร์ที่เป็นการส่ง SMS มาเพื่อหลอกให้กดลิงก์หรือให้ดาวน์โหลดโปรแกรมมัลแวร์ที่จะแอบติดตั้งเข้าไปอุปกรณ์โทรศัพท์มือถือ โดยทำการศึกษาหลักการแนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้องในหัวข้อเรื่อง การตรวจจับการหลอกลวงผ่านข้อความสั้น (Smishing) แนวทางในการพัฒนาแอปพลิเคชันหรือซอฟต์แวร์โดยใช้เครื่องมือที่ลดหรือไม่ต้องเขียนโค้ด (LowCode/NoCode) โมเดลพื้นฐานที่เหมาะสมสำหรับใช้ในการตรวจจับ Smishing และการเรียนรู้ของเครื่องบนอาซัวร์ (Azure Machine Learning) ซึ่งมีรายละเอียดดังนี้

2.1 การตรวจจับการหลอกลวงผ่านข้อความสั้น (Smishing)

2.1.1 นิยามและความหมายของการหลอกลวงผ่านข้อความสั้น (Smishing)

การหลอกลวงผ่านข้อความสั้น [5] หรือที่เรียกว่า Smishing (ย่อมาจาก SMS Phishing) เป็นรูปแบบหนึ่งของการโจมตีทางไซเบอร์ที่ผู้ไม่ประสงค์ดีใช้ข้อความสั้น (SMS) เพื่อหลอกลวงผู้รับให้เปิดเผยข้อมูลส่วนบุคคลหรือดำเนินการที่อาจก่อให้เกิดความเสียหายทางการเงินหรือความปลอดภัย คำว่า Smishing มาจากการผสมคำระหว่าง SMS (Short Message Service) และ Phishing (การหลอกลวงทางอินเทอร์เน็ต)

2.1.2 ประเภทของการหลอกลวงผ่านข้อความสั้น

ประเภทและรูปแบบของการโจมตีทางไซเบอร์ที่ผู้ไม่ประสงค์ดีใช้ข้อความสั้น (SMS) เพื่อหลอกลวงผู้รับให้เปิดเผยข้อมูลส่วนบุคคลหรือดำเนินการที่อาจก่อให้เกิดความเสียหาย จากการรวบรวมข้อมูลจากเอกสารราชการและแหล่งข้อมูลที่เชื่อถือได้ สามารถจำแนกประเภทของการหลอกลวงผ่านข้อความสั้นได้ดังนี้

2.1.2.1 การหลอกลวงผ่านอีเมล (Email Scam) ผู้ไม่หวังดีส่งอีเมลที่มีเนื้อหาแสดงความยินดี เช่น แจ้งว่าผู้รับถูกรางวัลหรือได้รับสิทธิพิเศษ เพื่อชักชวนให้กดลิงก์หรือให้ข้อมูลส่วนบุคคล [6]

2.1.2.2 การหลอกลวงผ่านโทรศัพท์ (Voice Phishing หรือ Vishing) ผู้ไม่หวังดีโทรศัพท์หาเหยื่อ โดยแอบอ้างเป็นเจ้าหน้าที่จากหน่วยงานหรือองค์กรที่น่าเชื่อถือ เพื่อขอข้อมูลส่วนบุคคลหรือข้อมูลทางการเงิน [7]

2.1.2.3 การหลอกลวงผ่านข้อความสั้น (Smishing) ผู้ไม่หวังดีส่งข้อความสั้น (SMS) ที่มีเนื้อหาโน้มน้าวหรือชักชวนให้เหยื่อกดลิงก์ไปยังเว็บไซต์ปลอมหรือให้ข้อมูลส่วนบุคคล [8]

2.1.2.4 การหลอกลวงโดยแอบอ้างเป็นบุคคลอื่นเพื่อยืมเงิน ผู้ไม่หวังดีนำภาพหรือข้อมูลของบุคคลอื่นมาใช้สร้างบัญชีปลอมในสื่อสังคมออนไลน์ หรือเข้าถึงบัญชีของบุคคลอื่นเพื่อสวมรอยแล้วส่งข้อความขอยืมเงินหรือขอให้โอนเงิน [9]

2.1.2.5 การหลอกลวงด้วยการเสนอสินเชื่อหรือเงินกู้ (Loan Scam) ผู้ไม่หวังดีส่งข้อความเสนอสินเชื่อหรือเงินกู้ที่มีเงื่อนไขน่าสนใจ เพื่อหลอกลวงให้เหยื่อให้ข้อมูลส่วนบุคคลหรือชำระค่าธรรมเนียมล่วงหน้า [10]

2.1.3 การตรวจจับการหลอกลวงผ่านข้อความสั้น (Smishing Detection)

การตรวจจับการหลอกลวงผ่านข้อความสั้น (Smishing Detection) เป็นกระบวนการในการระบุและแยกแยะข้อความสั้น (SMS) ที่มีแนวโน้มว่าจะเป็นการหลอกลวง โดยใช้วิธีการหลากหลายทั้งเชิงเทคนิคและการสังเกตพฤติกรรม [11] [12] [13]

2.1.3.1 วิธีการตรวจจับการหลอกลวงผ่านข้อความสั้น

2.1.3.1.1 การวิเคราะห์เนื้อหา (Content Analysis)

ก) การตรวจสอบคำและวลีที่น่าสงสัย: เนื้อหาที่มีข้อความล่อใจ เช่น “ด่วน”, “ยืนยันตัวตน”, “คุณถูกรางวัล” มีลิงก์ที่ย่อ URL เช่น bit.ly, tinyurl

ข) การตรวจสอบภาษาที่ผิดปกติ: ข้อความที่มีไวยากรณ์ผิด หรือใช้ภาษาแปลก ๆ อาจเป็นการหลอกลวง

2.1.3.1.2 การวิเคราะห์แหล่งที่มา (Source Analysis)

ก) ตรวจสอบหมายเลขโทรศัพท์: หากเป็นหมายเลขที่ไม่คุ้นเคย หรือหมายเลขต่างประเทศ ควรระมัดระวัง

ข) การตรวจสอบผู้ส่ง: หากชื่อผู้ส่งเป็นตัวอักษรผสมตัวเลข เช่น “B@nk01” อาจเป็นผู้ส่งปลอม

2.1.3.1.3 การใช้เทคโนโลยีการเรียนรู้ของเครื่อง (Machine Learning)

ก) การใช้โมเดลการจำแนกข้อความ (Text Classification): ใช้อัลกอริธึม เช่น Naive Bayes, SVM หรือ Neural Networks เพื่อจำแนกข้อความที่เป็นภัย

ข) การวิเคราะห์ความถี่คำ (Keyword Frequency): วิเคราะห์ความถี่ของคำที่เกี่ยวข้องกับการหลอกลวง เช่น “คลิก”, “รางวัล”, “ยืนยัน”

2.1.3.1.4 การวิเคราะห์พฤติกรรม (Behavioral Analysis)

ก) การวิเคราะห์การคลิกลิงก์: ตรวจสอบว่าผู้ใช้งานคลิกลิงก์ในข้อความหรือไม่

ข) การวิเคราะห์รูปแบบการตอบกลับ: หากผู้ใช้ตอบกลับข้อความที่น่าสงสัย อาจมีความเสี่ยงสูง

2.1.3.1.5 การใช้ระบบกรองข้อความ (SMS Filtering Systems)

ก) การใช้แอปพลิเคชันป้องกันสแปม (Anti-Spam Apps): เช่น Truecaller, Hiya ที่สามารถตรวจจับหมายเลขต้องสงสัย

ข) การใช้ระบบแจ้งเตือนภัยไซเบอร์: ระบบที่รวมฐานข้อมูลเบอร์โทรศัพท์อันตราย

2.1.3.2 เทคโนโลยีและเครื่องมือที่ใช้ในการตรวจจับ

2.1.3.2.1 SpamAssassin: ใช้วิเคราะห์ข้อความโดยพิจารณาจากคำหลักและโครงสร้าง

2.1.3.2.2 TensorFlow และ Scikit-learn: สำหรับสร้างโมเดล Machine Learning ตรวจจับ Smishing

2.1.3.2.3 Natural Language Processing (NLP): วิเคราะห์ข้อความเพื่อระบุเนื้อหาที่เป็นอันตราย

2.1.4 ความสำคัญของการตรวจจับการหลอกลวงผ่านข้อความสั้น

การหลอกลวงผ่านข้อความสั้น หรือที่เรียกว่า Smishing (SMS Phishing) เป็นการหลอกลวงที่เกิดจากการส่งข้อความสั้น (SMS) ที่มีลักษณะพิเศษเพื่อหลอกลวงให้ผู้รับเปิดเผยข้อมูลส่วนตัวหรือทำการใด ๆ ที่อาจก่อให้เกิดความเสียหายทั้งในด้านการเงินและข้อมูลส่วนบุคคล การตรวจจับการหลอกลวงประเภทนี้จึงเป็นสิ่งสำคัญที่ช่วยป้องกันไม่ให้ประชาชนตกเป็นเหยื่อของการโจมตีทางไซเบอร์และปกป้องทรัพย์สินและข้อมูลส่วนบุคคลจากการถูกขโมย ความสำคัญของการตรวจจับการหลอกลวงผ่านข้อความสั้น โดยสรุปมีดังต่อไปนี้ [14] [15] [16] [17]

2.1.4.1 ป้องกันการสูญเสียทางการเงินและทรัพย์สิน การหลอกลวงผ่านข้อความสั้นมักเกี่ยวข้องกับการขอให้เหยื่อทำการโอนเงินหรือให้ข้อมูลทางการเงิน เช่น หมายเลขบัตรเครดิต หรือรหัสผ่านการเงิน ซึ่งอาจทำให้ผู้ตกเป็นเหยื่อสูญเสียเงินจำนวนมากได้ การตรวจจับการหลอกลวงได้อย่างรวดเร็วและมีประสิทธิภาพจึงช่วยลดความเสี่ยงจากการสูญเสียเงินและทรัพย์สินของประชาชน

2.1.4.2 ป้องกันการขโมยข้อมูลส่วนบุคคล ผู้ไม่หวังดีอาจใช้ข้อความสั้นเพื่อขอข้อมูลส่วนบุคคล เช่น ชื่อผู้ใช้งาน, รหัสผ่าน หรือหมายเลขบัตรเครดิต ซึ่งสามารถนำไปใช้ในการแอบอ้างตัวตน

เพื่อทำธุรกรรมที่ไม่ถูกต้อง การตรวจจับการหลอกลวงผ่านข้อความสั้นช่วยปกป้องข้อมูลส่วนบุคคลของประชาชนไม่ให้ตกไปอยู่ในมือของผู้ไม่ประสงค์ดี

2.1.4.3 สร้างความเชื่อมั่นในระบบดิจิทัล เมื่อประชาชนได้รับการคุ้มครองจากการหลอกลวงออนไลน์ ก็จะมีความเชื่อมั่นในการใช้เทคโนโลยีต่าง ๆ ได้มากขึ้น การตรวจจับการหลอกลวงจึงไม่เพียงแต่ช่วยป้องกันความเสียหายทางการเงิน แต่ยังเป็นการส่งเสริมการใช้งานระบบดิจิทัลในสังคมอย่างปลอดภัย

2.1.4.4 ลดการเกิดอาชญากรรมไซเบอร์ การหลอกลวงผ่านข้อความสั้นเป็นหนึ่งในประเภทของอาชญากรรมไซเบอร์ที่มีแนวโน้มเพิ่มขึ้นอย่างรวดเร็ว การตรวจจับและป้องกันการหลอกลวงดังกล่าวช่วยลดจำนวนของอาชญากรรมทางไซเบอร์ โดยทำให้ผู้ไม่ประสงค์ดีไม่สามารถใช้วิธีนี้ในการหลอกลวงเหยื่อได้

2.1.4.5 สนับสนุนการปฏิบัติตามกฎหมายและมาตรการรักษาความปลอดภัย หน่วยงานที่เกี่ยวข้องสามารถใช้การตรวจจับการหลอกลวงเป็นส่วนหนึ่งของการบังคับใช้กฎหมายที่เกี่ยวข้อง เช่น พ.ร.บ. คอมพิวเตอร์ หรือกฎหมายที่เกี่ยวข้องกับการคุ้มครองข้อมูลส่วนบุคคล การตรวจจับอย่างมีประสิทธิภาพช่วยให้การปฏิบัติตามกฎหมายเป็นไปได้อย่างมีประสิทธิภาพ และป้องกันไม่ให้เกิดการละเมิดสิทธิของบุคคล

2.2 แนวทางในการพัฒนาแอปพลิเคชันหรือซอฟต์แวร์โดยใช้เครื่องมือที่ลดหรือไม่ต้องเขียนโค้ด (LowCode/NoCode)

การพัฒนาแอปพลิเคชันหรือซอฟต์แวร์โดยไม่ต้องเขียนโค้ด หรือที่รู้จักกันในชื่อ Low-Code และ No-Code (ลดหรือไม่ต้องเขียนโค้ด) ได้รับความนิยมเพิ่มขึ้นอย่างรวดเร็วในปัจจุบัน เนื่องจากความสะดวกในการใช้งานและสามารถทำให้ผู้ที่ไม่มีพื้นฐานทางเทคนิคสามารถพัฒนาแอปพลิเคชันได้โดยไม่ต้องมีความรู้ในการเขียนโค้ด ซอฟต์แวร์ประเภทนี้เหมาะสมสำหรับการสร้างแอปพลิเคชันที่ใช้ในธุรกิจต่าง ๆ รวมถึงการพัฒนาระบบที่ช่วยปรับปรุงการทำงานภายในองค์กรได้อย่างมีประสิทธิภาพ [18] [19] [20]

2.2.1 เครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชัน Low-Code/No-Code

2.2.1.1 เครื่องมือพัฒนาแอปพลิเคชัน (Low-Code/No-Code Platforms)

เครื่องมือเหล่านี้ช่วยให้การพัฒนาแอปพลิเคชันเป็นไปได้โดยใช้ การลากและวาง หรือการใช้เทมเพลตที่มีอยู่ ซึ่งสามารถช่วยให้ผู้ใช้งานสร้างแอปพลิเคชันที่มีฟังก์ชันการทำงานที่ซับซ้อนได้โดยไม่ต้องเขียนโค้ด ตัวอย่างเครื่องมือที่นิยม ได้แก่ OutSystems, AppSheet, Bubble, Microsoft PowerApps, และ Zoho Creator ซึ่งมีฟังก์ชันการทำงานที่ครอบคลุมทั้งการพัฒนาแอปพลิเคชันบนมือถือและเว็บ

2.2.1.2 การใช้การลากและวาง (Drag-and-Drop Interface)

หลักการของเครื่องมือประเภทนี้คือการใช้การลากและวางส่วนประกอบต่าง ๆ เช่น ปุ่ม, ฟอร์ม, หรือหน้าต่างการแสดงผล ซึ่งผู้ใช้สามารถเลือกและจัดการได้ตามต้องการ วิธีนี้ช่วยให้สามารถพัฒนาแอปพลิเคชันได้ในเวลาอันสั้นและลดความซับซ้อนในการพัฒนาที่ต้องใช้การเขียนโค้ด

2.2.1.3 การใช้เทมเพลตที่มีให้เลือก (Pre-built Templates)

เครื่องมือเหล่านี้มักจะมีเทมเพลตแอปพลิเคชันสำเร็จรูปที่สามารถนำมาใช้หรือปรับแต่งตามความต้องการของผู้ใช้งาน ตัวอย่างเช่น เทมเพลตสำหรับการจัดการคำสั่งซื้อในร้านค้า, การจัดการโปรเจกต์, หรือการสร้างฟอร์มลงทะเบียนผู้ใช้งาน

2.2.1.4 การเชื่อมต่อกับฐานข้อมูลและบริการอื่น ๆ (Integration)

เครื่องมือ Low-Code/No-Code ส่วนใหญ่รองรับการเชื่อมต่อกับระบบฐานข้อมูล หรือบริการที่มีอยู่แล้ว เช่น Google Sheets, Microsoft Excel, Salesforce, หรือ CRM systems การทำงานร่วมกับ API (Application Programming Interface) เพื่อดึงข้อมูลหรือเชื่อมต่อกับบริการต่าง ๆ ทำให้สามารถสร้างแอปพลิเคชันที่ทำงานร่วมกับระบบภายนอกได้

2.2.1.5 การปรับแต่งและพัฒนาเพิ่มเติม (Customization)

แม้ว่าเครื่องมือ Low-Code/No-Code จะลดขั้นตอนการเขียนโค้ด แต่ยังสามารถเพิ่มฟังก์ชันที่ซับซ้อนขึ้นได้โดยการใช้ JavaScript หรือ SQL ในการเขียนโค้ดเพิ่มเติมในบางกรณี สิ่งนี้ช่วยให้แอปพลิเคชันที่พัฒนาขึ้นสามารถขยายขอบเขตการทำงานได้ตามความต้องการของผู้ใช้

2.2.2 ขั้นตอนการพัฒนาแอปพลิเคชัน Low-Code/No-Code

2.2.2.1 การวิเคราะห์ความต้องการ

ขั้นตอนแรกของการพัฒนาแอปพลิเคชันคือการระบุความต้องการของผู้ใช้งาน เช่น ฟังก์ชันการทำงานที่ต้องการ, การออกแบบ UI/UX และการใช้งานของแอปพลิเคชัน ผู้พัฒนาแอปพลิเคชันต้องทำความเข้าใจลักษณะของแอปพลิเคชันที่จะพัฒนาและปัญหาที่ต้องการแก้ไข

2.2.2.2 การออกแบบและพัฒนา

ในขั้นตอนนี้ผู้ใช้งานสามารถเริ่มต้นสร้างแอปพลิเคชันโดยใช้เครื่องมือที่มีอยู่ โดยการลากและวางส่วนประกอบต่าง ๆ ของแอปพลิเคชัน การเชื่อมต่อกับฐานข้อมูลและระบบต่าง ๆ จะช่วยให้แอปพลิเคชันสามารถทำงานได้ตามความต้องการ

2.2.2.3 การทดสอบและการปรับปรุง

หลังจากพัฒนาเสร็จแล้ว ผู้พัฒนาจะต้องทำการทดสอบแอปพลิเคชันเพื่อให้แน่ใจว่าแอปพลิเคชันทำงานได้ตามที่คาดหวัง หากพบปัญหาหรือข้อผิดพลาด จะต้องปรับปรุงแอปพลิเคชันตามคำแนะนำจากการทดสอบ

2.2.2.4 การเผยแพร่และดูแลรักษา

เมื่อแอปพลิเคชันพัฒนาเสร็จสมบูรณ์แล้ว ผู้พัฒนาสามารถเผยแพร่แอปพลิเคชันไปยังผู้ใช้งานหรือใช้งานภายในองค์กร การดูแลรักษาแอปพลิเคชันจะต้องมีการอัปเดตอย่างต่อเนื่องเพื่อให้สามารถใช้งานได้อย่างมีประสิทธิภาพ

2.3 โมเดลพื้นฐานที่เหมาะสมสำหรับใช้ในการตรวจจับ Smishing

SMS Phishing หรือที่เรียกกันว่า Smishing (SMS + Phishing) เป็นเทคนิคการโจมตีทางไซเบอร์ที่ใช้ข้อความ SMS เพื่อหลอกลวงให้เหยื่อเปิดเผยข้อมูลส่วนตัว เช่น รหัสผ่าน ข้อมูลบัตรเครดิต หรือคลิกลิงก์ที่เป็นอันตราย การตรวจจับ SMS Phishing เป็นปัญหาที่เกี่ยวข้องกับ การประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) ซึ่งเป็นแขนงหนึ่งของ ปัญญาประดิษฐ์ (Artificial Intelligence: AI) ที่ช่วยให้คอมพิวเตอร์สามารถเข้าใจ วิเคราะห์ และประมวลผลข้อความได้อย่างมีประสิทธิภาพ

2.3.1 การถดถอยโลจิสติก (Logistic Regression)

Logistic Regression เป็นเทคนิคทางสถิติและการเรียนรู้ของเครื่องที่ใช้ใน การจำแนกประเภทข้อมูลแบบไบนารีหรือพหุคลาส โมเดลนี้เป็นการขยายแนวคิดของการถดถอยเชิงเส้น (Linear Regression) โดยใช้ฟังก์ชันโลจิสติก (Logistic Function) หรือที่เรียกว่าฟังก์ชันซิกมอยด์ (Sigmoid Function) เพื่อแปลงค่าผลลัพธ์ให้อยู่ในช่วงระหว่าง 0 ถึง 1 บทความนี้อธิบายถึงหลักการทางคณิตศาสตร์ของโมเดล การประมาณค่าพารามิเตอร์โดยใช้วิธี Maximum Likelihood Estimation (MLE) และการนำไปประยุกต์ใช้ในสาขาต่าง ๆ เช่น การแพทย์ การเงิน และวิทยาการข้อมูลในปัญหาการจำแนกประเภทข้อมูล เช่น การพิจารณาว่าผู้ป่วยมีโรคหรือไม่ หรือการพิจารณาว่าลูกค้าจะซื้อสินค้าหรือไม่ การถดถอยโลจิสติกเป็นหนึ่งในเทคนิคที่ได้รับความนิยมสูงสุด เนื่องจากมีความสามารถในการจำแนกข้อมูลไบนารีและสามารถขยายไปสู่การจำแนกประเภทแบบพหุคลาส (Multinomial Logistic Regression) ได้

การถดถอยโลจิสติกถูกนำมาใช้ครั้งแรกในสถิติและต่อมาได้รับการพัฒนาให้เป็นอัลกอริธึมที่สำคัญใน Machine Learning (ML) โดยเฉพาะสำหรับปัญหา Supervised Learning ที่ต้องการทำนายตัวแปรเป้าหมายที่ไม่ต่อเนื่อง (Discrete Output)

2.3.1.1 ฟังก์ชันโลจิสติก (Logistic Function หรือ Sigmoid Function) การถดถอยโลจิสติกใช้ฟังก์ชันซิกมอยด์เพื่อแปลงค่าการคาดการณ์ให้อยู่ระหว่าง 0 และ 1

2.3.1.2 การประมาณค่าพารามิเตอร์โดยใช้ Maximum Likelihood Estimation (MLE) ในการหาค่าพารามิเตอร์ที่เหมาะสม โมเดลจะใช้ Maximum Likelihood Estimation (MLE) แทนการลดค่าความคลาดเคลื่อนแบบ Least Squares ที่ใช้ใน Linear Regression ฟังก์ชันความน่าจะ

เป็นของ Logistic Regression ค่าที่เหมาะสมของ คือค่าที่ทำให้ฟังก์ชันนี้มีค่าสูงสุด โดยใช้เทคนิค Gradient Descent หรือ Newton's Method

2.3.1.3 Multinomial Logistic Regression (Softmax Regression) เมื่อมีมากกว่าสองคลาส (Multiclass Classification) เราสามารถใช้ Softmax Function แทน Sigmoid Function เพื่อปรับปรุงการจำแนกประเภทให้รองรับหลายคลาส

2.3.1.4 Regularization ใน Logistic Regression การเพิ่ม Regularization Term ช่วยลดปัญหา Overfitting โดยมีสองรูปแบบหลัก คือ L1 Regularization (Lasso Regression): ลดจำนวนฟีเจอร์โดยเลือกค่าพารามิเตอร์บางค่าที่เป็นศูนย์ และ L2 Regularization (Ridge Regression): ป้องกันค่าพารามิเตอร์ที่มีขนาดใหญ่เกินไป

การถดถอยโลจิสติกเป็นอัลกอริธึมที่สำคัญในสถิติและ Machine Learning เนื่องจากมีโครงสร้างที่เรียบง่ายแต่สามารถนำไปใช้กับปัญหาการจำแนกประเภทที่หลากหลายได้ การใช้ Maximum Likelihood Estimation (MLE) ทำให้สามารถประมาณค่าพารามิเตอร์ได้อย่างแม่นยำ และการเพิ่ม Regularization ช่วยป้องกันปัญหา Overfitting การนำไปประยุกต์ใช้ในด้านต่าง ๆ แสดงให้เห็นถึงความยืดหยุ่นของโมเดลนี้

2.3.2 นาอีฟเบย (Naïve Bayes)

Naïve Bayes เป็นอัลกอริทึม (Algorithm) การจำแนกประเภท ที่อาศัยทฤษฎีเบย์ (Bayes Theorem) โดยตั้งสมมติฐานว่าคุณลักษณะทั้งหมดเป็นอิสระต่อกัน แม้ในความเป็นจริงอาจไม่เป็นเช่นนั้น แต่โมเดลนี้ยังคงทำงานได้ดีและมีประสิทธิภาพสูงในหลายๆงาน เช่น การจำแนกข้อความและการตรวจจับข้อความที่ไม่พึงประสงค์ (Spam) [21]

Naïve Bayes (NB) เป็นอัลกอริธึมการจำแนกประเภทที่อาศัยพื้นฐานของทฤษฎีบทของเบย์ (Bayes' Theorem) โดยใช้สมมติฐานว่าคุณลักษณะของข้อมูลเป็นอิสระต่อกัน (Conditional Independence Assumption) แม้ว่าในความเป็นจริงข้อมูลอาจมีความสัมพันธ์กันก็ตาม อย่างไรก็ตาม NB ยังคงเป็นอัลกอริธึมที่มีประสิทธิภาพและได้รับความนิยมอย่างมากในปัญหาต่าง ๆ เช่น การวิเคราะห์ข้อความ (Text Classification), การตรวจจับอีเมลสแปม (Spam Detection) และการพยากรณ์ทางการแพทย์ (Medical Diagnosis)

Naïve Bayes เป็นอัลกอริธึมการเรียนรู้ของเครื่องที่ใช้แนวคิดของความน่าจะเป็นในการจำแนกประเภท โดยมีพื้นฐานมาจากทฤษฎีบทของเบย์ ซึ่งช่วยให้สามารถคำนวณค่าความน่าจะเป็นของแต่ละประเภท (Class) ได้จากชุดข้อมูลที่กำหนด โมเดลนี้ถูกนำไปใช้ในหลายสาขา เช่น การวิเคราะห์ข้อความ (Text Classification): เช่น การจำแนกอีเมลเป็น "สแปม" หรือ "ไม่สแปม", การวิเคราะห์ความคิดเห็น (Sentiment Analysis): ใช้ในการพิจารณาว่าข้อความมีแนวโน้มเป็นบวกหรือลบ และการวิเคราะห์ทางชีวการแพทย์: ใช้ในระบบวินิจฉัยโรคโดยอาศัยค่าความน่าจะเป็นของอาการต่าง ๆ

ถึงแม้ว่า Naïve Bayes จะมีสมมติฐานที่ค่อนข้างเรียบง่าย แต่ก็สามารถให้ผลลัพธ์ที่มีประสิทธิภาพในหลาย ๆ งาน

2.3.2.1 ทฤษฎีพื้นฐานของ Naïve Bayes ใช้ ทฤษฎีบทของเบย์ (Bayes' Theorem) ซึ่งสามารถทำการคำนวณโดยถือว่าคุณลักษณะของข้อมูลทั้งหมดเป็นอิสระจากกัน ซึ่งนำไปสู่สมการที่ง่ายขึ้นดังนี้

2.3.2.2 ประเภทของ Naïve Bayes มีหลายประเภทที่ใช้สำหรับข้อมูลที่แตกต่างกัน ได้แก่ Gaussian Naïve Bayes (GNB) ใช้สำหรับข้อมูลที่มีการแจกแจงแบบปกติ (Normal Distribution) โดยความน่าจะเป็นของคุณลักษณะของการคำนวณจากฟังก์ชันความหนาแน่นของความน่าจะเป็น (PDF) ของการแจกแจงปกติ

Multinomial Naïve Bayes (MNB) ใช้สำหรับข้อมูลที่เป็นค่าจำนวนเต็ม เช่น จำนวนครั้งที่คำปรากฏในเอกสารหนึ่ง ๆ เหมาะสำหรับปัญหาการจำแนกข้อความ เช่น Spam Detection

Bernoulli Naïve Bayes (BNB) ใช้สำหรับข้อมูลที่เป็นค่าทางตรรกศาสตร์ (Binary) เช่น คำใดคำหนึ่งปรากฏหรือไม่ปรากฏในเอกสาร (0 หรือ 1) มักใช้ในงานตรวจจับอีเมลสแปมเช่นเดียวกับ MNB แต่เหมาะสำหรับข้อมูลที่เป็น Boolean Features

งานวิจัยจำนวนมากที่นำ Naïve Bayes ไปใช้ในงานด้านต่าง ๆ เช่น การตรวจจับอีเมลสแปม: Sahami et al. (1998) พบว่า Naïve Bayes สามารถจำแนกอีเมลสแปมได้อย่างมีประสิทธิภาพโดยใช้การแจกแจงของคำภายในข้อความ, การวิเคราะห์ความรู้สึก (Sentiment Analysis): Pang et al. (2002) ใช้ Naïve Bayes ในการจำแนกความคิดเห็นของผู้ใช้เป็นเชิงบวกหรือลบ และพบว่ามีประสิทธิภาพใกล้เคียงกับอัลกอริธึมอื่น ๆ และ การวิเคราะห์ข้อมูลทางการแพทย์: Kononenko (2001) ศึกษาการใช้ Naïve Bayes ในการวินิจฉัยโรค และพบว่าเป็นเครื่องมือที่มีประโยชน์สำหรับการช่วยตัดสินใจของแพทย์ เป็นต้น

Naïve Bayes เป็นอัลกอริธึมที่มีพื้นฐานจากความน่าจะเป็นและมีโครงสร้างที่เรียบง่าย แต่มีประสิทธิภาพสูงในงานจำแนกประเภทหลายประเภท โดยเฉพาะงานที่เกี่ยวข้องกับข้อความ แม้ว่าจะมีข้อจำกัดบางประการ เช่น สมมติฐานว่าคุณลักษณะของข้อมูลเป็นอิสระกัน แต่ก็สามารถแก้ไขได้โดยการใช้เทคนิคการปรับปรุงต่าง ๆ Naïve Bayes ยังคงเป็นหนึ่งในโมเดลที่ได้รับความนิยมและใช้งานได้ดีในสาขาต่าง ๆ

SMS Phishing Detection เป็นปัญหาด้านการประมวลผล ภาษาธรรมชาติ (Natural Language Processing : NLP) ที่วิเคราะห์ข้อความเพื่อตรวจหาความเสี่ยง เช่น ลิงก์ อันตรายหรือการขอข้อมูลส่วนตัว การเลือกโมเดล Machine Learning ที่เหมาะสมช่วยลดความเสี่ยงจากการโจมตี หากมีข้อจำกัดด้านทรัพยากร ควรใช้โมเดลที่มีความซับซ้อนต่ำ เช่น "นาอีฟเบย์" (Naïve

Bayes: NB) หรือ “การถดถอยโลจิสติกส์” (Logistic Regression: LR) ซึ่งมีความรวดเร็วและแม่นยำในการจำแนกข้อความต้องสงสัย [22] [23]

2.4 การเรียนรู้ของเครื่องบนอาชัวร์ (Azure Machine Learning) [24]

ใน Azure ML มีเครื่องมือที่รองรับ การพัฒนาแอปพลิเคชันแบบ Low Code / No Code ซึ่งช่วยให้การสร้างและใช้งานโมเดล Machine Learning เป็นเรื่องที่ง่ายและสะดวก โดยไม่จำเป็นต้องเขียนโค้ด หรือสามารถเขียนโค้ดเพียงเล็กน้อย ซึ่งเหมาะสำหรับผู้ที่ไม่มีความเชี่ยวชาญด้านการ เขียนโค้ด หรือผู้ที่ต้องการพัฒนาโมเดลอย่างรวดเร็วเครื่องมือเหล่านี้ยังสามารถนำไปใช้ในการวิจัยเพื่อพัฒนาและทดสอบโมเดลต่าง ๆ ได้อย่างมีประสิทธิภาพ

2.4.1 การสร้างแบบจำลอง Machine Learning แบบอัตโนมัติ (Automated Machine Learning : AutoML)

การเรียนรู้ของเครื่อง (Machine Learning: ML) เป็นเทคโนโลยีที่มีบทบาทสำคัญในยุคดิจิทัล แต่กระบวนการพัฒนาแบบจำลอง ML มักต้องใช้เวลาและทรัพยากรสูง รวมถึงต้องอาศัยผู้เชี่ยวชาญด้านข้อมูล (Data Scientists) ในการเลือกคุณลักษณะ (Feature Selection), การปรับค่าพารามิเตอร์ (Hyperparameter Tuning) และการเลือกโมเดลที่เหมาะสม เพื่อแก้ไขปัญหาดังกล่าว การเรียนรู้ของเครื่องแบบอัตโนมัติ (Automated Machine Learning: AutoML) ได้ถูกพัฒนาขึ้นเพื่อช่วยลดภาระของมนุษย์ในการสร้างแบบจำลอง โดย AutoML สามารถเลือกโมเดลที่ดีที่สุด ปรับค่าพารามิเตอร์ให้เหมาะสม และปรับปรุงประสิทธิภาพของโมเดลได้โดยอัตโนมัติ ในช่วงไม่กี่ปีที่ผ่านมา Machine Learning (ML) ได้ถูกนำไปใช้ในหลายอุตสาหกรรม เช่น การเงิน การแพทย์ และวิทยาศาสตร์ข้อมูล อย่างไรก็ตาม การพัฒนาแบบจำลอง ML ที่มีประสิทธิภาพต้องใช้เวลานานและต้องอาศัยผู้เชี่ยวชาญ ซึ่งเป็นอุปสรรคสำคัญสำหรับองค์กรที่ไม่มีทรัพยากรเพียงพอ
Automated Machine Learning (AutoML) เป็นแนวคิดที่ช่วยลดความซับซ้อนของกระบวนการ ML โดยอัตโนมัติ ซึ่งช่วยให้ผู้ใช้ที่ไม่มีความเชี่ยวชาญด้านวิทยาศาสตร์ข้อมูลสามารถสร้างแบบจำลองที่มีประสิทธิภาพสูงได้อย่างง่ายดาย AutoML ครอบคลุมกระบวนการตั้งแต่การเตรียมข้อมูล การเลือกโมเดล การปรับค่าพารามิเตอร์ จนถึงการประเมินผลแบบจำลอง

2.4.1.1 แนวคิดของ Automated Machine Learning (AutoML) AutoML เป็นกระบวนการที่ใช้ อัลกอริธึมอัตโนมัติ เพื่อเลือกโมเดลที่ดีที่สุดและปรับปรุงประสิทธิภาพของโมเดล ML โดยทั่วไป AutoML จะมีองค์ประกอบหลักดังต่อไปนี้

2.4.1.1.1 การเตรียมข้อมูลอัตโนมัติ (Automated Data Preprocessing) AutoML สามารถทำความสะอาดข้อมูล เติมค่าที่หายไป (Missing Values), เลือกคุณลักษณะที่สำคัญ (Feature Selection) และแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม

2.4.1.1.2 การเลือกโมเดลที่เหมาะสม (Model Selection) AutoML สามารถทดสอบโมเดลหลายประเภท เช่น Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Machines (GBM), Deep Learning และเลือกโมเดลที่ให้ผลลัพธ์ดีที่สุดโดยอัตโนมัติ

2.4.1.1.3 การปรับค่าพารามิเตอร์ (Hyperparameter Optimization) AutoML ใช้ เทคนิคการค้นหาพารามิเตอร์ที่ดีที่สุด เช่น Grid Search, Random Search, Bayesian Optimization และEvolutionary Algorithms เพื่อปรับค่าพารามิเตอร์ให้โมเดลมีประสิทธิภาพสูงสุด

2.4.1.1.4 การประเมินผลแบบจำลอง (Model Evaluation & Selection) AutoML ใช้เทคนิคต่าง ๆ เช่น Cross-Validation, AUC-ROC, F1 Score, RMSE เพื่อตรวจสอบและเลือกแบบจำลองที่เหมาะสมที่สุด

2.4.1.2 ข้อดีและข้อเสียของ AutoML

2.4.1.2.1 ข้อดีของ AutoML ลดเวลาและทรัพยากรที่ต้องใช้ในการพัฒนาแบบจำลอง, ลดความจำเป็นในการใช้ผู้เชี่ยวชาญด้าน Data Science, สามารถทดสอบโมเดลหลายตัวและเลือกตัวที่ดีที่สุดได้โดยอัตโนมัติ และ รองรับงานที่หลากหลาย เช่น การพยากรณ์ การวิเคราะห์ข้อความ และการจดจำภาพ เป็นต้น

2.4.1.2.2 ข้อเสียของ AutoML อาจไม่สามารถปรับแต่งแบบจำลองในเชิงลึกได้เท่ากับการพัฒนาแบบจำลองเอง, ใช้พลังงานในการคำนวณสูง เนื่องจากต้องทดลองโมเดลและพารามิเตอร์จำนวนมาก และอาจไม่เหมาะสำหรับงานที่ต้องการความเข้าใจเชิงลึกเกี่ยวกับข้อมูล เป็นต้น

AutoML เป็นเทคโนโลยีที่ช่วยลดความซับซ้อนของกระบวนการพัฒนาแบบจำลอง Machine Learning ทำให้ผู้ที่ไม่มีความเชี่ยวชาญสามารถใช้ ML ได้ง่ายขึ้น แม้ว่าจะมีข้อจำกัดบางอย่าง เช่น ข้อจำกัดในการปรับแต่งแบบจำลอง และต้องใช้ทรัพยากรในการคำนวณสูง แต่ AutoML ยังคงเป็นแนวทางที่สำคัญในการพัฒนา AI ในอนาคต

2.4.2 ออกแบบ Machine Learning บนแอชัวร์ (AzureML Designer)

AzureML Designer ช่วยให้ผู้ใช้สามารถออกแบบกระบวนการทำงาน (Workflow) ของโมเดล Machine Learning ได้ในลักษณะการลากและวาง (Drag-and-Drop) โดยไม่ต้องเขียนโค้ด ผู้ใช้สามารถเลือกโมดูลที่เตรียมไว้แล้ว เช่น การเตรียมข้อมูล การเลือกฟีเจอร์ การฝึกโมเดล และการประเมินผลจากชุดข้อมูลคุณสมบัติหลัก โมดูลที่สำเร็จรูปเพื่อการทำงานร่วมกัน

Machine Learning (ML) ได้กลายเป็นเครื่องมือสำคัญในการพัฒนานวัตกรรมในหลากหลายอุตสาหกรรม อย่างไรก็ตาม การพัฒนาแบบจำลอง ML มักต้องอาศัยทักษะเชิงลึกด้านการเขียนโค้ด

และการปรับแต่งแบบจำลอง Azure Machine Learning Designer (Azure ML Designer) เป็นแพลตฟอร์มที่ช่วยให้ผู้ใช้สามารถสร้าง ทดสอบ และปรับแต่งโมเดล ML ได้โดยไม่ต้องเขียนโค้ดมากมายผ่านอินเทอร์เฟซแบบลากและวาง (Drag-and-Drop)

ในยุคปัจจุบัน Machine Learning ได้ถูกนำไปประยุกต์ใช้ในหลากหลายด้าน เช่น การแพทย์ การเงิน การค้าปลีก และระบบอัตโนมัติ อย่างไรก็ตาม อุปสรรคที่สำคัญของการใช้ ML คือความซับซ้อนของกระบวนการพัฒนาแบบจำลองที่ต้องใช้ ความเชี่ยวชาญด้านการเขียนโค้ด และการปรับค่าพารามิเตอร์

Microsoft Azure Machine Learning Designer เป็นเครื่องมือที่ช่วยให้ผู้ใช้สามารถออกแบบเวิร์กโฟลว์การเรียนรู้ของเครื่องแบบภาพ (Visual Workflow) โดยไม่ต้องมีทักษะการเขียนโค้ดเชิงลึก ซึ่งเป็นโซลูชันที่เหมาะสำหรับ นักพัฒนา นักวิทยาศาสตร์ข้อมูล และองค์กรที่ต้องการนำ AI มาใช้ในธุรกิจของตน

2.4.2.1 แนวคิดและองค์ประกอบหลักของ Azure ML Designer Azure ML Designer เป็นส่วนหนึ่งของ Azure Machine Learning Studio โดยให้ผู้ใช้สามารถพัฒนา ML Model ผ่านอินเทอร์เฟซแบบลากและวางที่ใช้งานง่าย องค์ประกอบหลักของ Azure ML Designer มีดังนี้

2.4.2.1.1 อินเทอร์เฟซแบบลากและวาง (Drag-and-Drop Interface) Azure ML Designer มี เครื่องมือช่วยออกแบบแบบจำลอง (Model Builder) ที่ช่วยให้ผู้ใช้สามารถ สร้างเวิร์กโฟลว์ ML ได้โดยการลากและวางโมดูลต่างๆ เช่น การโหลดข้อมูล การเลือกฟีเจอร์ การฝึกสอนโมเดล และการประเมินผล

2.4.2.1.2 โมดูลที่พร้อมใช้งาน (Prebuilt Modules) Azure ML Designer มีโมดูลที่รองรับขั้นตอนสำคัญ เช่น Data Ingestion: โหลดข้อมูลจาก Azure Blob Storage, SQL Server หรือไฟล์ CSV, Data Transformation: การล้างข้อมูลและการเลือกคุณลักษณะ (Feature Engineering), Model Training: ใช้โมเดล เช่น Logistic Regression, Decision Tree, Random Forest, Deep Learning และ Model Evaluation: การวัดผลโมเดล เช่น Accuracy, AUC-ROC, RMSE เป็นต้น

2.4.2.1.3 การเชื่อมต่อกับ Azure Services Azure ML Designer รองรับการใช้งานร่วมกับ Azure Synapse Analytics, Azure Data Lake, Azure Databricks และบริการอื่นๆ ทำให้สามารถปรับแต่งและนำแบบจำลองไปใช้งานจริงได้ง่ายขึ้น

2.4.2.1.4 การทำ MLOps และ Deployment ผู้ใช้สามารถ นำแบบจำลองที่พัฒนาไป Deploy เป็น REST API และเชื่อมต่อกับ Azure Kubernetes Service (AKS), Azure Functions หรือ Power BI ได้อย่างง่ายดาย

2.4.2.2 กระบวนการทำงานของ Azure ML Designer กระบวนการพัฒนา Machine Learning บน Azure ML Designer สามารถแบ่งออกเป็น 5 ขั้นตอนหลัก ได้แก่

2.4.2.2.1 การนำเข้าข้อมูล (Data Ingestion) ผู้ใช้สามารถเลือกแหล่งข้อมูล เช่น Azure Blob Storage, SQL Database, Web Data Sources หรือ Upload ไฟล์ CSV และเชื่อมโยงกับโมดูลการประมวลผลข้อมูล

2.4.2.2.2 การเตรียมและแปลงข้อมูล (Data Preprocessing & Transformation) การทำ Feature Engineering, การจัดการข้อมูลที่ขาดหายไป (Handling Missing Data) และการลดมิติของข้อมูล (Dimensionality Reduction)

2.4.2.2.3 การเลือกและฝึกสอนโมเดล (Model Selection & Training) ผู้ใช้สามารถเลือกโมเดล ML ที่เหมาะสมจากโมดูลสำเร็จรูป เช่น Regression: Linear Regression, Decision Tree Regression, Classification: Logistic Regression, Support Vector Machines (SVM), Neural Networks, Clustering: K-Means Clustering

2.4.2.2.4 การประเมินผลและการเลือกแบบจำลอง (Model Evaluation & Selection) ระบบจะช่วยคำนวณค่า Accuracy, Precision, Recall, AUC-ROC และ RMSE เพื่อช่วยให้ผู้ใช้สามารถเลือกแบบจำลองที่ดีที่สุด

2.4.2.2.5 การ Deploy และ Monitoring เมื่อได้แบบจำลองที่เหมาะสมแล้ว สามารถ Deploy เป็น Web Service API บน Azure Kubernetes Service (AKS) หรือ Azure App Service เพื่อให้ระบบภายนอกสามารถเข้าถึงแบบจำลองได้

2.4.2.3 ข้อดีและข้อเสียของ Azure ML Designer

2.4.2.3.1 ข้อดีของ Azure ML Designer ใช้งานง่าย ไม่ต้องเขียนโค้ดมากมาย, รองรับการทำ End-to-End Machine Learning Pipeline, สามารถเชื่อมต่อกับแหล่งข้อมูลและเครื่องมือใน Azure ได้อย่างมีประสิทธิภาพ และรองรับ MLOps สำหรับการปรับใช้และตรวจสอบโมเดลในระยะยาว เป็นต้น

2.4.2.3.2 ข้อเสียของ Azure ML Designer มีข้อจำกัดในการปรับแต่งโมเดลเมื่อเทียบกับการใช้ Python หรือ R, ค่าใช้จ่ายสูงเมื่อใช้งานบน Cloud ในระดับองค์กร และไม่สามารถใช้ได้กับ Open-source frameworks บางประเภท เป็นต้น

Azure Machine Learning Designer เป็นแพลตฟอร์มที่ช่วยลดความซับซ้อนของกระบวนการพัฒนา ML โดยการให้ผู้ใช้สามารถออกแบบโมเดลผ่าน Drag-and-Drop Interface ทำให้ผู้ใช้ที่ไม่มีความเชี่ยวชาญด้านการเขียนโค้ดสามารถสร้างและปรับใช้โมเดล ML ได้ง่ายขึ้น อย่างไรก็ตาม แม้ว่าจะมีข้อจำกัดในเรื่องของการปรับแต่งแบบจำลอง Azure ML Designer ยังคงเป็นเครื่องมือที่เหมาะสำหรับธุรกิจที่ต้องการนำ AI มาใช้งานได้อย่างรวดเร็ว

2.5 งานวิจัยที่เกี่ยวข้อง

2.5.1 บทความนี้ [25] วิจารณ์ระบบตรวจจับสแปมที่ใช้ปัญญาประดิษฐ์อย่างมีวิจารณญาณ การทบทวนนี้แสดงรายการระบบที่มีอยู่ซึ่งใช้เครื่องจักรและเทคนิคการเรียนรู้เชิงลึกพร้อมข้อจำกัด ข้อดี และข้อเสีย นอกจากนี้ เอกสารนี้ยังครอบคลุมถึงขอบเขตสำหรับการปรับปรุงในอนาคตในการประมวลผลภาษาธรรมชาติ เพื่อป้องกันข้อความสแปมอย่างมีประสิทธิภาพ แทนที่จะตรวจจับข้อความสแปม

2.5.2 บทความนี้ [26] ใช้การเรียนรู้เชิงลึกเพื่อแยกประเภทข้อความตัวอักษรที่เป็นสแปมและไม่ใช่สแปม เพื่อขยายขอบเขตการตีพิมพ์ในปัจจุบัน โดยเฉพาะมีการใช้โมเดล Convolutional Neural Network และ Long Short-Term Memory แบบจำลองที่นำเสนอนั้นขึ้นอยู่กับข้อมูลข้อความเท่านั้น และได้แยกชุดคุณลักษณะออกด้วยตนเอง ในชุดข้อมูลการวัดประสิทธิภาพประกอบด้วยข้อความสแปม 747 ข้อความและข้อความที่ไม่ใช่สแปม 4,827 ข้อความ มีความแม่นยำที่น่าทึ่งถึงร้อยละ 99.44

2.5.3 บทความนี้ [27] การเติบโตของผู้ใช้โทรศัพท์มือถือส่งผลให้ข้อความสแปม SMS เพิ่มขึ้นอย่างมาก ในทางปฏิบัติ การต่อสู้กับสแปมโทรศัพท์มือถือเป็นเรื่องยากด้วยปัจจัยหลายประการ รวมถึงอัตรา SMS ที่ต่ำลงซึ่งทำให้ผู้ใช้และผู้ให้บริการจำนวนมากเพิกเฉยต่อปัญหานี้ และซอฟต์แวร์กรองสแปมโทรศัพท์มือถือที่มีจำกัด ในทางกลับกัน ในด้านวิชาการ อุปสรรคสำคัญคือการขาดแคลนชุดข้อมูลสแปม SMS สาธารณะ ซึ่งจำเป็นอย่างยิ่งสำหรับการตรวจสอบและเปรียบเทียบตัวแยกประเภทต่างๆ นอกจากนี้ เนื่องจากข้อความ SMS ค่อนข้างสั้น ตัวกรองสแปมตามเนื้อหาจึงอาจมีประสิทธิภาพลดลง ในบทความนี้ เรานำเสนอคอลเลกชัน สแปม SMS ที่แท้จริง สาธารณะ และไม่ได้เข้ารหัส ซึ่งเป็นคอลเลกชันที่ใหญ่ที่สุดเท่าที่เราทราบ นอกจากนี้ เรายังเปรียบเทียบประสิทธิภาพที่ได้รับจากวิธีการเรียนรู้ของเครื่องจักรหลายวิธี ผลลัพธ์ระบุว่า Support Vector Machine มีประสิทธิภาพเหนือกว่าตัวแยกประเภทที่ได้รับการประเมินอื่นๆ และด้วยเหตุนี้ จึงสามารถใช้เป็นพื้นฐานที่ดีสำหรับการเปรียบเทียบเพิ่มเติม

2.5.4 การศึกษานี้ [28] เกี่ยวกับปัญหาดังกล่าวในอดีตอยู่ในระดับไบนารี่ว่าเป็นแฮมหรือสแปม ซึ่งทำให้เกิดการปกปิดข้อความเฉพาะที่เป็นประโยชน์ต่อผู้ใช้ปลายทาง แต่กลับถูกมองว่าเป็นสแปม นอกจากนี้ยังขยายออกไปด้วยป้ายกำกับหลายป้าย เช่น แฮม สแปม และอื่นๆ ซึ่งไม่เพียงพอที่จะตอบสนองความจำเป็นทั้งหมดของผู้ใช้ปลายทาง ดังนั้นจึงจำเป็นต้องมีการจัดหมวดหมู่ SMS แบบหลายคลาสตามความหมาย (ข้อมูล) ที่ฝังอยู่ในนั้น บทความนี้แนะนำโมเดลตอบกลับอัตโนมัติอัจฉริยะโดยใช้ดัชนีความหมายสำหรับจัดหมวดหมู่ข้อความ SMS ออกเป็น 5 หมวดหมู่ ได้แก่ แฮม สแปม ข้อมูล ธุรกรรม และรหัสผ่านแบบใช้ครั้งเดียว โดยใช้อัลกอริธึม Perceptron หลายชั้น (MLP) ในแนวทางนี้ SMS แต่ละรายการจะถูกจัดประเภทเป็นหนึ่งในหมวดหมู่ที่กำหนดไว้ล่วงหน้า การ

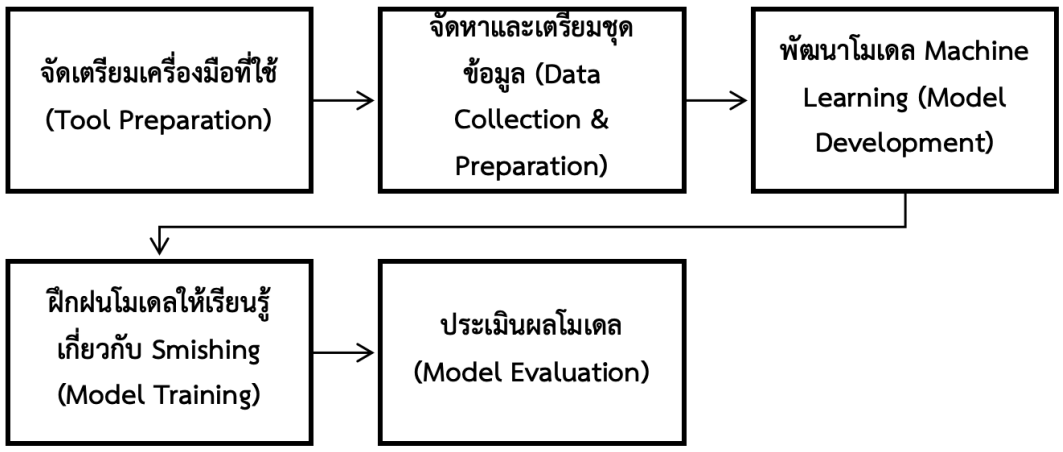
ทดลองนี้ดำเนินการบน "ชุดข้อมูล SMS หลายคลาส" ซึ่งมีข้อความ 7,398 ข้อความ ซึ่งแบ่งออกเป็น 5 คลาส ความแม่นยำที่ได้จากการทดลองคือร้อยละ 97

2.5.5 งานวิจัยนี้ [29] อธิบายการเติบโตของแพลตฟอร์ม low-code และ no-code ที่ช่วยให้การพัฒนาแอปพลิเคชันที่ใช้ machine learning ง่ายขึ้นสำหรับผู้ใช้งานที่ไม่มีพื้นฐานในการเขียนโค้ด การใช้เครื่องมือเหล่านี้ช่วยให้สามารถสร้างโมเดล machine learning ที่มีความซับซ้อนได้โดยไม่ต้องมีความรู้ในเชิงลึกเกี่ยวกับเทคโนโลยี

2.5.6 งานวิจัยนี้ [30] สำรวจแพลตฟอร์ม no-code และ low-code ที่ใช้ในการพัฒนา AI และ machine learning โดยเฉพาะ มันเน้นที่เครื่องมือที่ช่วยให้ผู้ใช้สามารถสร้างแอปพลิเคชันที่ใช้ AI โดยไม่จำเป็นต้องเขียนโค้ด สิ่งนี้ช่วยเปิดโอกาสให้กับบุคคลที่ไม่มีพื้นฐานการเขียนโปรแกรม แต่ต้องการใช้เทคโนโลยี AI ในการทำงาน

บทที่ 3
วิธีการดำเนินการวิจัย

กระบวนการสร้างโมเดลตรวจจับ Smishing ประกอบด้วย การจัดเตรียมเครื่องมือ, การเตรียมชุดข้อมูล, การพัฒนาโมเดล, การฝึกฝนโมเดล และการประเมินผล โมเดล Machine Learning สามารถช่วยเพิ่มความปลอดภัยในการใช้ SMS และลดโอกาสที่ผู้ใช้งานจะถูกหลอกลวงจากข้อความ Phishing ได้ หากโมเดลมีการปรับปรุงอย่างต่อเนื่อง ก็สามารถช่วยให้สามารถตรวจจับภัยคุกคามได้อย่างมีประสิทธิภาพมากขึ้นโดยกระบวนการพัฒนาโมเดล Machine Learning เพื่อตรวจจับ SMS Phishing หรือ Smishing ในงานวิจัยนี้สามารถแบ่งออกเป็น 5 ขั้นตอนหลัก ดังนี้



ภาพที่ 3-1 กระบวนการทำงานในการดำเนินการวิจัย

3.1 จัดเตรียมเครื่องมือที่ใช้ (Tool Preparation) โดยเลือกใช้ เครื่องมือแบบโอเพนซอร์ส (Open-source Tools) ที่สามารถใช้งานได้ฟรี เช่น Python, Scikit-learn, TensorFlow, Pandas และ Jupyter Notebook เพื่อช่วยในการพัฒนาและวิเคราะห์โมเดล Machine Learning ศึกษาวิธีการใช้งานเครื่องมือที่เลือก เช่น การติดตั้งไลบรารีที่จำเป็น การเรียกใช้งานฟังก์ชัน และการตั้งค่าการทำงานของโมเดล ตรวจสอบ ความเข้ากันได้ของเครื่องมือ กับอุปกรณ์และระบบปฏิบัติการ เพื่อให้แน่ใจว่าสามารถทำงานได้อย่างราบรื่น

3.2 จัดหาและเตรียมชุดข้อมูล (Data Collection & Preparation) ค้นหาและเลือก Dataset ที่เหมาะสม เช่น ชุดข้อมูลจาก Kaggle, UCI Machine Learning Repository หรือแหล่งข้อมูลที่เกี่ยวข้อง ซึ่งมีตัวอย่างข้อความ SMS ที่เป็น Spam และ Ham (ข้อความปกติ) และนำมาวิเคราะห์ ทำความสะอาดข้อมูล (Data Cleaning) เช่น ลบข้อความที่ซ้ำซ้อน ลบอักขระพิเศษ เช่น เครื่องหมายวรรคตอน หรืออีโมจิ แปลงข้อความเป็น ตัวพิมพ์เล็กทั้งหมด (Lowercase) ลบคำที่ไม่จำเป็น (Stopwords) เช่น “คือ”, “ที่”, “และ” เป็นต้น, ใช้ Tokenization เพื่อแบ่งคำในประโยค แปลงข้อมูลเป็นรูปแบบที่ใช้กับ Machine Learning Model ได้ เช่น การใช้ TF-IDF (Term Frequency - Inverse Document Frequency) หรือ Word Embeddings เพื่อแปลงข้อความเป็นตัวเลข

3.3 พัฒนาโมเดล Machine Learning (Model Development) เลือกโมเดล Machine Learning ที่เหมาะสม เช่น Naïve Bayes (NB): ใช้งานง่ายและมีประสิทธิภาพสูงในการจำแนกข้อความ, Logistic Regression (LR): มีความเร็วสูงและสามารถนำไปใช้งานได้ง่าย, Random Forest (RF): มีความแม่นยำสูงและจัดการกับฟีเจอร์จำนวนมากได้ดี, Deep Learning (LSTM, CNN): เหมาะสำหรับงานที่มีข้อมูลจำนวนมากและต้องการเรียนรู้บริบทของข้อความ พัฒนาโมเดลโดยใช้เครื่องมือ No Code/Low Code เช่น Azure ML Designer หรือ Google AutoML ซึ่งช่วยให้สามารถสร้างโมเดลได้โดยไม่ต้องเขียนโค้ดมากนัก ปรับแต่งพารามิเตอร์ของโมเดล (Hyperparameter Tuning) เช่น ปรับค่าการเรียนรู้ (Learning Rate) ปรับขนาดชุดข้อมูลฝึก (Batch Size) เลือกจำนวนรอบการฝึก (Epochs),

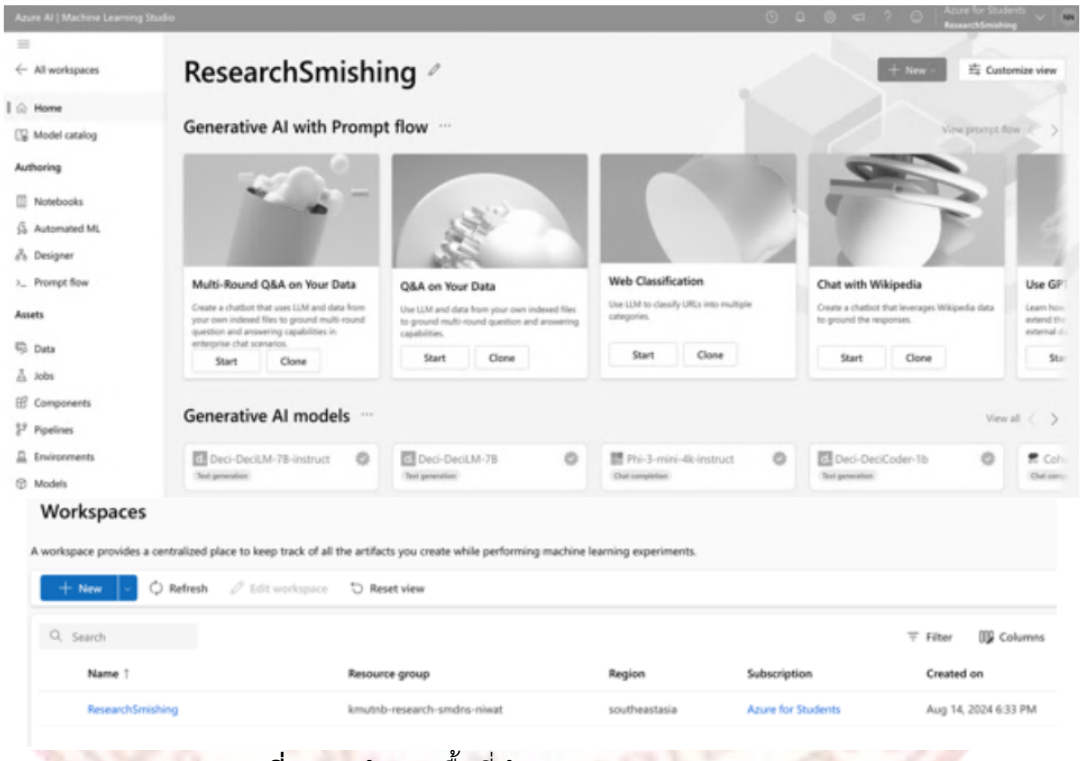
3.4 ฝึกฝนโมเดลให้เรียนรู้เกี่ยวกับ Smishing (Model Training) แบ่งข้อมูลออกเป็น ชุดฝึกสอน (Training Set) และชุดทดสอบ (Test Set) เช่น 80:20 หรือ 70:30 เพื่อให้โมเดลเรียนรู้จากข้อมูลและสามารถทดสอบผลลัพธ์ได้ ใช้อัลกอริธึม Gradient Descent หรือ Backpropagation (สำหรับ Neural Networks) เพื่อปรับปรุงประสิทธิภาพของโมเดล ตรวจสอบ ค่าความสูญเสีย (Loss Function) และความแม่นยำ (Accuracy) ของโมเดลเพื่อให้แน่ใจว่าโมเดลไม่ได้เรียนรู้แบบ Overfitting หรือ Underfitting ทดลองปรับแต่งฟีเจอร์เพิ่มเติม เช่น การเพิ่ม Stopwords, การใช้ Stemming หรือ Lemmatization, การขยายข้อมูล (Data Augmentation)

3.5 ประเมินผลโมเดล (Model Evaluation) ใช้ Metrics วัดประสิทธิภาพของโมเดล เช่น Accuracy: วัดความถูกต้องของโมเดลโดยรวม, Precision & Recall: ตรวจสอบอัตราการตรวจจับข้อความ Phishing ได้อย่างถูกต้อง, F1-Score: ใช้ในการเปรียบเทียบโมเดลที่มี Precision และ Recall ต่างกัน, Confusion Matrix: ตรวจสอบการทำนายที่ผิดพลาดของโมเดล ทดสอบโมเดลกับข้อมูลจริง (Real-world Testing) โดยใช้ข้อความ Smishing ที่ไม่ได้อยู่ในชุดข้อมูลฝึกเพื่อดูว่าโมเดลสามารถตรวจจับได้แม่นยำหรือไม่ วิเคราะห์ข้อผิดพลาดของโมเดล และนำไปปรับปรุงในรอบถัดไป

3.1 จัดเตรียมเครื่องมือที่ใช้ (Tool Preparation)

3.1.1 กำหนดพื้นที่ทำงาน Azure Machine Learning

การกำหนดพื้นที่ทำงาน (Workspace) ใน Azure Machine Learning เป็นขั้นตอนสำคัญในการเริ่มต้นพัฒนาและจัดการโครงการ Machine Learning บนแพลตฟอร์ม Azure โดยพื้นที่ทำงานนี้ทำหน้าที่เป็นศูนย์กลางสำหรับการจัดการทรัพยากรที่เกี่ยวข้องกับ Machine Learning เช่น ชุดข้อมูล (Datasets), โมเดล (Models), และการทดลอง (Experiments) นอกจากนี้ ยังช่วยในการติดตามและควบคุมกระบวนการทำงานของทีมได้อย่างมีประสิทธิภาพ



ภาพที่ 3-2 กำหนดพื้นที่ทำงาน Azure Machine Learning

3.1.2 ขั้นตอนการสร้างพื้นที่ทำงาน Azure Machine Learning

1) เข้าสู่ Azure Portal

- ไปที่ Azure Portal และลงชื่อเข้าใช้ด้วยบัญชี Microsoft

2) สร้างทรัพยากรใหม่

- คลิกที่ “สร้างทรัพยากร” (Create a resource)
- ค้นหา “Machine Learning” ในแถบค้นหา
- เลือก “Machine Learning” จากผลการค้นหา และคลิก “สร้าง” (Create)

3) สร้างทรัพยากรใหม่

- Subscription: เลือก Subscription ที่ต้องการใช้
- Resource Group: สร้าง Resource Group ใหม่
- Workspace Name: ตั้งชื่อพื้นที่ทำงาน
- Region: เลือกภูมิภาคที่ต้องการให้พื้นที่ทำงานตั้งอยู่
- Storage Account, Key Vault, Application Insights, Container Registry: สามารถเลือกให้ระบบสร้างใหม่โดยอัตโนมัติ

4) สร้างทรัพยากรใหม่

- ตรวจสอบข้อมูลที่กรอกทั้งหมด
- คลิก “ตรวจสอบและสร้าง” (Review + Create)
- หลังจากการตรวจสอบเสร็จสิ้น คลิก “สร้าง” (Create) เพื่อเริ่มกระบวนการสร้างพื้นที่ทำงาน

3.1.3 การสร้างงานการเรียนรู้ของเครื่องอัตโนมัติ (Create an Automated Machine Learning job)

การสร้างงานการเรียนรู้ของเครื่องอัตโนมัติ (Automated Machine Learning หรือ AutoML) หมายถึงการใช้เครื่องมือของ Azure เพื่อให้ระบบทำการเลือก, สร้าง, และฝึกฝนโมเดล machine learning โดยอัตโนมัติ โดยที่ผู้ใช้ไม่จำเป็นต้องมีความรู้ลึกในด้านการเขียนโค้ดหรือการปรับแต่งโมเดล ซึ่งเหมาะสำหรับการค้นหาวิธีที่ดีที่สุดในการสร้างโมเดลสำหรับข้อมูลที่กำหนด โดยกระบวนการนี้จะดำเนินการแบบอัตโนมัติ ตั้งแต่การเตรียมข้อมูลไปจนถึงการเลือกและฝึกฝนโมเดลที่เหมาะสม

3.1.3.1 ขั้นตอนการสร้างพื้นที่ทำงานของ AutoML

1) เปิด Azure Machine Learning Studio

2) ไปที่ Automated ML

3) คลิก New automated ML run

4) เลือก Dataset ที่ต้องการใช้ในการฝึกฝนโมเดล

5) กำหนด Task type เช่น Classification, Regression หรือ Forecasting

6) กำหนด Compute target ซึ่งเป็นทรัพยากรที่ Azure จะใช้ในการฝึกโมเดล

7) ตั้งค่าพารามิเตอร์อื่น ๆ ตามที่ต้องการ เช่น เวลาในการฝึกฝน, จำนวนของโมเดลที่ต้องการทดลอง, และวิธีการประเมินผล

8) คลิก Start เพื่อเริ่มการฝึกฝนโมเดล

3.1.4 การสร้าง Azure Machine Learning Designer

Azure Machine Learning Designer เป็นเครื่องมือที่ใช้สำหรับสร้างและฝึกฝนโมเดล Machine Learning แบบกราฟิก โดยไม่ต้องเขียนโค้ด ช่วยให้ผู้ใช้สามารถลากและวางคอมโพเนนต์ต่างๆ เพื่อสร้าง pipeline สำหรับการประมวลผลข้อมูลและการฝึกฝนโมเดลได้ง่ายขึ้น

3.1.4.1 ขั้นตอนการใช้งาน Azure ML Designer

1) เปิด Azure Machine Learning Studio

2) เลือก Designer ใน Azure Machine Learning Studio, ไปที่ Designer ซึ่งจะช่วยให้สามารถสร้าง pipeline โดยใช้การลากและวาง

3) คลิก + New pipeline เพื่อสร้าง pipeline ใหม่ ขั้นตอนแรกของการสร้าง pipeline คือการเพิ่ม Data Input เพื่อเชื่อมต่อกับแหล่งข้อมูล (Dataset) ที่จะใช้ในการฝึกฝนโมเดล

4) เลือก Dataset ที่ต้องการใช้จาก Azure ML Datasets หรือ Data from file

5) ลาก Dataset จาก panel ด้านซ้ายไปยัง canvas

6) ใช้คอมโพเนนต์ต่าง ๆ สำหรับการเตรียมข้อมูล เช่น การทำความสะอาดข้อมูล, การแปลงข้อมูล, การจัดการค่าที่ขาดหาย, และการแปลงฟีเจอร์

7) เลือกอัลกอริทึมที่คุณต้องการใช้ในการฝึกฝนโมเดล

3.2 จัดหาและเตรียมชุดข้อมูล (Data Collection & Preparation)

การจัดประเภทและการวิเคราะห์ข้อมูลเหล่านี้มีความสำคัญในการพัฒนาโมเดล Machine Learning สำหรับตรวจจับและป้องกัน Smishing ซึ่งเป็นภัยคุกคามที่พบบ่อยในปัจจุบัน เนื่องจาก Smishing เป็นการโจมตีโดยใช้ข้อความสั้น (SMS) ที่แอบอ้างหรือปลอมแปลงข้อมูล เพื่อหลอกลวงให้ผู้ใช้งานเปิดลิงก์ที่เป็นอันตรายหรือให้ข้อมูลส่วนบุคคล เช่น รหัสผ่านหรือข้อมูลบัตรเครดิต การนำชุดข้อมูลดังกล่าวมาวิเคราะห์ช่วยให้สามารถสร้างโมเดลในการจำแนกข้อความระหว่าง ham, spam และ smishing ได้อย่างแม่นยำ การใช้เทคนิค Machine Learning เช่น Naïve Bayes, Logistic Regression หรือ Deep Learning จะช่วยเพิ่มประสิทธิภาพในการตรวจจับข้อความฟิชชิ่งได้อย่างทันท่วงที ส่งผลให้สามารถป้องกันผู้ใช้งานจากการถูกหลอกลวงและเพิ่มความปลอดภัยในระบบการสื่อสารผ่านข้อความได้มากขึ้น

3.2.1 ชุดข้อมูลที่ใช้ในการทดลอง

โดยการเลือกข้อมูลจากชุดข้อมูลสาธารณะ (Public Dataset) ที่เผยแพร่เมื่อวันที่ 20 มิถุนายน 2022 โดยชุดข้อมูลนี้เป็นชุดข้อความที่ได้รับการติดป้าย (Label) ที่ถูกรวบรวมขึ้นเพื่อการวิจัยเกี่ยวกับ Smishing ซึ่งเป็นการโจมตีทางไซเบอร์ผ่านข้อความสั้น (SMS) ที่มีลักษณะเป็นฟิชชิ่ง ชุด

ข้อมูลนี้ประกอบด้วย 5,971 ข้อความ โดยแบ่งประเภทข้อความออกเป็น 3 ประเภท ได้แก่ ข้อความที่ไม่เป็นอันตราย (Ham), Spam, และ Smishing โดยมีจำนวนข้อความในแต่ละประเภทดังนี้

ข้อความ Ham จำนวน 4,844 ข้อความ

ข้อความ Spam จำนวน 489 ข้อความ

ข้อความ Smishing จำนวน 638 ข้อความ

LABEL	TEXT	URL	EMAIL	PHONE
ham	Your opinion about me? 1. Over 2. Jada 3. Kusruthi 4. Lovable 5. Silent 6. Spl character 7. Not matured 8. Stylish 9. Simple Pls reply..	No	No	No
ham	What's up? Do you want me to come online? If you are free we can talk sometime	No	No	No
ham	So u workin overtime nigpun?	No	No	No
ham	Also sir, i sent you an email about how to log into the usc payment portal. I.ll send you another message that should explain how things are back home. Have a great weekend.	No	No	No
Smishing	Please Stay At Home. To encourage the notion of staying at home. All tax-paying citizens are entitled to 305.96 or more emergency refund. smsg.io/fCVbD	No	No	No
Smishing	BankOfAmerica Alert 137943. Please follow http://bit.do/cgjK-and re-activate	yes	No	yes
ham	Sorry dude. Dont know how i forgot. Even after Dan reminded me. Sorry. Hope you guys had fun.	No	No	No
ham	I don't quite know what to do. I still can't get hold of anyone. I cud pick you up bout 7.30pm and we can see if they're in the pub?	No	No	No
ham	Ok lor. Anyway i thk we cant get tickets now cos like quite late already. U wan 2 go look 4 ur frens a not? Darren is wif them now...	No	No	No
ham	Wat r u doing now?	No	No	No
ham	Buy one egg for me da..please:)	No	No	No
ham	Is there a reason we've not spoken this year? Anyways have a great week and all the best in your exam	No	No	No
ham	Stop the story. I've told him i've returned it and he's saying i should not re order it.	No	No	No
ham	No. I dont want to hear anything	No	No	No
ham	Waqt se pehle or naseeb se zyada kisi ko kuch nahi milta,Zindgi wo nahi he jo hum sochte hai Zindgi wo hai jo ham jeetey hai.....	No	No	No
ham	I'm e person who's doing e sms survey...	No	No	No
ham	Mm i had my food da from out	No	No	No
ham	4 tacos + 1 rajas burrito, right?	No	No	No
ham	House-Maid is the murderer, coz the man was murdered on 26th January..	No	No	No
ham	Yes! How is a pretty lady like you single?	No	No	No
ham	(That said can you text him one more time?)	No	No	No
ham	I know she called me	No	No	No
ham	In sch but neva mind u eat 1st lor..	No	No	No
spam	lyricalladie(21/F) is inviting you to be her friend. Reply YES-910 or NO-910. See her: www.SMS.ac/u/hmmross STOP? Send STOP FRND to 62468	No	No	yes
ham	Jay says that you're a double-faggot	No	No	No
ham	Ok i thk i got it. Then u wan me 2 come now or wat?	No	No	No
Smishing	UR awarded a City Break and could WIN a £200 Summer Shopping spree every WK. Txt STORE to 88039 . SkilGme. TsCs087147403233	No	No	yes
spam	Hello from Orange. For 1 month's free access to games, news and sport, plus 10 free texts and 20 photo messages, reply YES. Terms apply: www.orange.co.uk/ow	yes	No	No
ham	Oh... Lk tt den we take e one tt ends at cine lor... Dun wan yogasana oso can...	No	No	No
ham	Whens your radio show?	No	No	No
ham	Jane babes not goin 2 wrk, feel ill after lst nite. Foned in already cover 4 me chuck.:-)	No	No	No

ภาพที่ 3-3 เตรียมชุดข้อมูล Smishing

จากภาพที่ 3-3 เป็นข้อมูลตัวอย่างของข้อความ (SMS) ในรูปแบบของภาษาอังกฤษ ที่ถูกจัดประเภทเพื่อวิเคราะห์ว่าเป็น “ham” (ข้อความปกติ) หรือ “spam/smishing” (ข้อความสแปมหรือฟิชชิ่ง) โดยมีคอลัมน์หลักดังนี้:

LABEL - ประเภทของข้อความ แบ่งเป็น ham, spam, และ smishing

TEXT – เนื้อหาของข้อความที่ได้รับ

URL - ระบุว่ามีลิงก์ URL อยู่ในข้อความหรือไม่ (Yes/No)

EMAIL - ระบุว่ามีอีเมลอยู่ในข้อความหรือไม่ (Yes/No)

PHONE - ระบุว่ามีหมายเลขโทรศัพท์อยู่ในข้อความหรือไม่ (Yes/No)

3.2.2 การจัดเตรียมข้อมูลก่อนการประมวลผล (Data pre-processing)

กระบวนการเตรียมและแปลงข้อมูลดิบ (Raw Data) ให้อยู่ในรูปแบบที่เหมาะสมสำหรับการวิเคราะห์หรือใช้งานในโมเดล Machine Learning (ML) โดยข้อมูลดิบมักจะมีปัญหา เช่น ข้อมูลสูญหาย (Missing Data), ข้อมูลผิดรูปแบบ (Inconsistent Data), หรือข้อมูลซ้ำซ้อน (Duplicate Data)

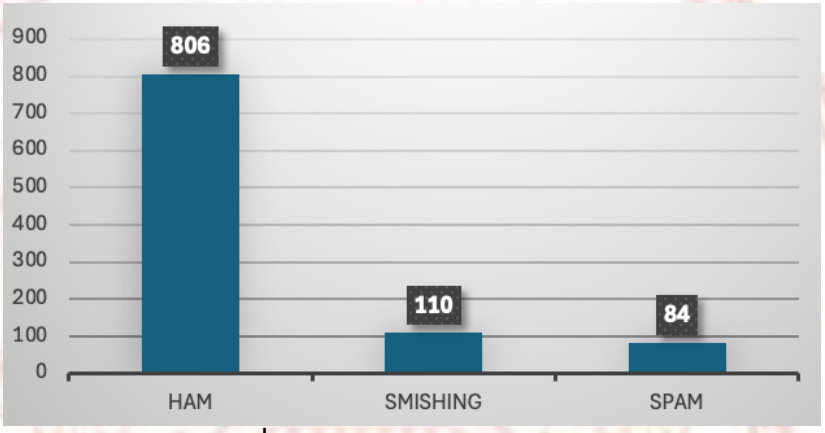
3.2.2.1 Data Reduction (การลดขนาดข้อมูล) ลดปริมาณข้อมูลดิบลง โดยยังคงรักษาข้อมูลสำคัญและคุณลักษณะหลักไว้ ซึ่งช่วยลดภาระในการประมวลผลและเพิ่มประสิทธิภาพในการวิเคราะห์หรือสร้างโมเดล Machine Learning โดยลดแถวที่ไม่ต้องการ และให้คงเหลือเฉพาะคอลัมน์ที่สนใจคือ LABLE และ TEXT

LABEL	TEXT
ham	Your opinion about me? 1. Over 2. Jada 3. Kusruthi 4. Lovable 5. Silent 6. Spl character 7. Not matured 8. Stylish 9. Simple Pls reply..
ham	What's up? Do you want me to come online? If you are free we can talk sometime
ham	So u workin overtime nigpun?
ham	Also sir, i sent you an email about how to log into the usc payment portal. I.ll send you another message that should explain how things are back home. Have a great weekend.
Smishing	Please Stay At Home. To encourage the notion of staying at home. All tax-paying citizens are entitled to 305.96 or more emergency refund. smsg.io/fCVbD
Smishing	BankOfAmerica Alert 137943. Please follow http://bit.do/cgjK-and re-activate
ham	Sorry dude. Dont know how i forgot. Even after Dan reminded me. Sorry. Hope you guys had fun.
ham	I don't quite know what to do. I still can't get hold of anyone. I cud pick you up bout 7.30pm and we can see if they're in the pub?
ham	Ok lor. Anyway i thk we cant get tickets now cos like quite late already. U wan 2 go look 4 ur frens a not? Darren is wif them now...
ham	Wat r u doing now?
ham	Buy one egg for me da..please:)
ham	Is there a reason we've not spoken this year? Anyways have a great week and all the best in your exam
ham	Stop the story. I've told him i've returned it and he's saying i should not re order it.
ham	No. I dont want to hear anything
ham	Waqt se pehle or naseeb se zyada kisi ko kuch nahi milta,Zindgi wo nahi he jo hum sochte hai Zindgi wo hai jo ham jeetey hai.....
ham	I'm e person who's doing e sms survey...
ham	Mm i had my food da from out
ham	4 tacos + 1 rajas burrito, right?
ham	House-Maid is the murderer, coz the man was murdered on 26th January..
ham	Yes! How is a pretty lady like you single?
ham	(That said can you text him one more time?)
ham	I know she called me
ham	In sch but neva mind u eat 1st lor..
spam	lyricalladie(21/F) is inviting you to be her friend. Reply YES-910 or NO-910. See her: www.SMS.ac/u/hmmross STOP? Send STOP FRND to 62468
ham	Jay says that you're a double-faggot
ham	Ok i thk i got it. Then u wan me 2 come now or wat?
Smishing	UR awarded a City Break and could WIN a £200 Summer Shopping spree every WK. Txt STORE to 88039 . SkilGme. TsCs087147403233
spam	Hello from Orange. For 1 month's free access to games, news and sport, plus 10 free texts and 20 photo messages, reply YES. Terms apply: www.orange.co.uk/ow
ham	Oh... Lk tt den we take e one tt ends at cine lor... Dun wan yogasana oso can...
ham	Whens your radio show?

ภาพที่ 3-4 ลดขนาดข้อมูล

จากภาพที่ 3-4 เป็นผลลัพธ์หลังจากทำการลดขนาดข้อมูล โดยตัดในส่วนของ URL - ระบุว่ามีลิงก์ URL อยู่ในข้อความหรือไม่ (Yes/No) EMAIL - ระบุว่ามีอีเมลอยู่ในข้อความหรือไม่ (Yes/No) และ PHONE - ระบุว่ามีหมายเลขโทรศัพท์อยู่ในข้อความหรือไม่ (Yes/No) และให้คงเหลือเฉพาะคอลัมน์ที่สนใจคือ LABLE และ TEXT

3.2.2.2 Data Aggregation (การรวมกลุ่มข้อมูล) เป็นกระบวนการของข้อมูลหลายรายการให้เป็นข้อมูลที่ย่อและให้มีความกระชับขึ้น ซึ่งช่วยในการวิเคราะห์และทำความเข้าใจข้อมูลได้ง่ายขึ้น ทำให้ข้อมูลขนาดใหญ่กลายเป็นข้อมูลที่เรียบง่ายและเข้าใจง่ายขึ้น เพื่อช่วยลดเวลาในการประมวลผล หลังจาก ชุดข้อมูลนี้ Data Aggregation แล้ว จะประกอบด้วย 1,000 ข้อความ ประกอบด้วย Spam จำนวน 84 ข้อความ Smishing จำนวน 110 ข้อความ และข้อความที่เป็น Ham จำนวน 806 ข้อความ



ภาพที่ 3-5 ข้อมูล Data Aggregation

3.2.2.3 การแปลง Word เป็น Vector (Word2vec) โดยใช้เทคนิคในการแปลงคำ (Word) ให้อยู่ในรูปแบบเวกเตอร์ของตัวเลข (Vector) เพื่อให้คอมพิวเตอร์สามารถประมวลผลและวิเคราะห์ข้อความได้ โดยเฉพาะในงานประมวลผลภาษาธรรมชาติ (NLP) และ Machine Learning ซึ่งคำหรือข้อความเป็นข้อมูลประเภทไม่ใช่เชิงตัวเลข (Textual Data) ซึ่งคอมพิวเตอร์ไม่สามารถนำไปคำนวณหรือวิเคราะห์ได้โดยตรง การแปลงเป็นเวกเตอร์ช่วยให้โมเดล Machine Learning สามารถวิเคราะห์ความสัมพันธ์ระหว่างคำและความคล้ายคลึงกันได้ โดยแปลง Word ในคอลัมน์ TEXT

การใช้ Text Vectorization ในการตรวจจับ Smishing ผ่าน Machine Learning เป็นกระบวนการที่มีความสำคัญ เนื่องจาก Smishing (SMS Phishing) มักมีลักษณะเฉพาะที่แตกต่างจากข้อความปกติ (Ham) หรือข้อความสแปม (Spam) เช่น การมีลิงก์ URL, เบอร์โทรศัพท์, คำเชิญชวน, หรือข้อความที่ให้ดำเนินการบางอย่าง (Actionable Text)

3.2.2.3.1 Text Vectorization ในบริบทของ AutoML (Automated Machine Learning) คือกระบวนการแปลงข้อความดิบให้เป็นรูปแบบเชิงตัวเลข (เวกเตอร์) เพื่อให้โมเดล Machine Learning สามารถประมวลผลและวิเคราะห์ได้ เนื่องจากโมเดลเรียนรู้จากตัวเลข ไม่ใช่ข้อความโดยตรง

การนำ Text Vectorization มาใช้ใน AutoML เพื่อจำแนก Smishing นั้นมีประสิทธิภาพในการตรวจจับภัยคุกคามที่แฝงมากับข้อความสั้น (SMS) โดยการใช้เทคนิคที่เหมาะสม เช่น TF-IDF และ Word Embeddings ร่วมกับโมเดลอัตโนมัติ สามารถช่วยเพิ่มความแม่นยำในการตรวจจับได้อย่างมีนัยสำคัญ

3.2.2.3.2 ในบริบทของ Azure Machine Learning Designer (Azure ML Designer) การใช้ Text Vectorization ถือเป็นขั้นตอนสำคัญในการแปลงข้อความดิบ (Raw Text) ให้เป็นเวกเตอร์เชิงตัวเลขเพื่อให้โมเดลสามารถนำไปใช้งานได้ เนื่องจากโมเดลไม่สามารถประมวลผลข้อมูลที่เป็นข้อความโดยตรงได้

การใช้ Text Vectorization ใน Azure ML Designer ช่วยให้สามารถแปลงข้อความดิบเป็นเวกเตอร์ได้อย่างรวดเร็วและแม่นยำ โดยใช้โมดูลที่หลากหลาย เช่น TF-IDF, N-Gram, และ Word Embeddings ช่วยในการวิเคราะห์และจำแนกประเภท Smishing ได้อย่างมีประสิทธิภาพ

3.3 พัฒนาโมเดล Machine Learning (Model Development)

3.3.1 กำหนดหน่วยประมวลผลข้อมูลที่ใช้ในการพัฒนาโมเดล

การกำหนด Compute ใน Azure Machine Learning (Azure ML) คือการเลือกและจัดการทรัพยากรคอมพิวเตอร์ (เช่น เครื่องคอมพิวเตอร์หรือคลัสเตอร์ของเครื่อง) ที่จะใช้ในการฝึกอบรมโมเดล Machine Learning (ML) หรือในการประมวลผลข้อมูลต่าง ๆ ที่เกี่ยวข้องในกระบวนการพัฒนา AI และ ML

Azure ML ช่วยให้สามารถสร้างและจัดการ Compute Resources ที่เหมาะสมสำหรับงานต่าง ๆ ได้อย่างง่ายดาย ไม่ว่าจะเป็นการฝึกโมเดล, การทดสอบ, หรือการประมวลผลข้อมูลจำนวนมากที่จำเป็นต้องใช้พลังคำนวณสูง โดยสามารถเลือกใช้ Compute ที่เหมาะสมกับงานแต่ละประเภททั้งในแง่ของ ขนาด, ประเภท และ ความสามารถ ของเครื่องคอมพิวเตอร์ที่ใช้ในกระบวนการต่างๆ

	Name	Category	Available quota ⓘ	Cost ↑ ⓘ
●	Standard_DS1_v2 1 cores, 3.5GB RAM, 7GB storage	General purpose	6 cores	$0.08/hr
○	Standard_F2s_v2 2 cores, 4GB RAM, 16GB storage	Compute optimiz...	16 cores	$0.10/hr
○	Standard_D2as_v4 2 cores, 8GB RAM, 16GB storage	General purpose	4 cores	$0.12/hr
○	Standard_D2s_v3 2 cores, 8GB RAM, 16GB storage	General purpose	6 cores	$0.12/hr
○	Standard_D2_v3 2 cores, 8GB RAM, 50GB storage	General purpose	6 cores	$0.12/hr
○	Standard_D2_v2 2 cores, 7GB RAM, 100GB storage	General purpose	6 cores	$0.16/hr

ภาพที่ 3-6 กำหนดหน่วยประมวลผลข้อมูลที่ใช้ในการพัฒนาโมเดล

จากภาพที่ 3-6 เป็นการระบุทรัพยากรที่ใช้ในการประมวลผลโดยใช้เครื่องเสมือน (Virtual Machine) ขนาด 1 cores, 3.5 GB RAM, 7 GB disk

3.3.2 การพัฒนาโมเดลโดยใช้การ AutoML

3.3.2.1 การใช้ Logistic Regression Algorithm ใช้ในการคาดการณ์ผลลัพธ์ที่เป็นหมวดหมู่ (Categorical Outcomes) โดยเฉพาะในกรณีที่ข้อมูลถูกแบ่งออกเป็นสองหมวดหมู่เช่น การจำแนกประเภท ซึ่งตัวอย่าง เช่น การคาดการณ์ว่า Email นั้นเป็น Spam หรือ Ham

3.3.2.2 การใช้ MultinomialNaiveBayes Algorithm เป็น Algorithm แบบ Classification ที่ใช้หลักการของทฤษฎีความน่าจะเป็น โดยเฉพาะ ทฤษฎีเบย์ (Bayes Theorem) ซึ่งอธิบายความน่าจะเป็นของเหตุการณ์หนึ่งที่เกิดขึ้นภายใต้ข้อมูลที่รู้ล่วงหน้า (Prior Knowledge) เหมาะสมสำหรับการจำแนกประเภทข้อความ เช่นการแยกข้อความที/มีลักษณะการหลอกลวงจากข้อความทั่วไป โดยใช้คำสำคัญในข้อความสั้น เช่น คำที่มีลักษณะการหลอกลวง, คำที่เกี่ยวข้องกับการขอข้อมูลส่วนตัว, และคำที่มักใช้ใน ตัวระบุแหล่งข้อมูลสากล (Uniform Resource Locator -URL) ที่อันตราย

3.3.3 การพัฒนาโมเดลโดยใช้ Azure ML Designer

ในการทดลองของผู้วิจัยเกี่ยวกับการออกแบบในการออกแบบโดยไม่ต้องเขียนโค้ด (Authoring NoCode) โดยใช้ Azure ML Designer ซึ่งเป็นเครื่องมือแบบ NoCode ที่ใช้การ Drag-and-Drop

3.3.3.1 Algorithm เครือข่ายประสาทเทียมหลายคลาส (Multiclass Neural Network Algorithm) ใช้โมเดลปัญญาประดิษฐ์ที่อาศัยเครือข่ายประสาทเทียม (Neural Network) สำหรับการจำแนกประเภทข้อมูลที่มีมากกว่า 2 คลาส (Multiclass Classification)

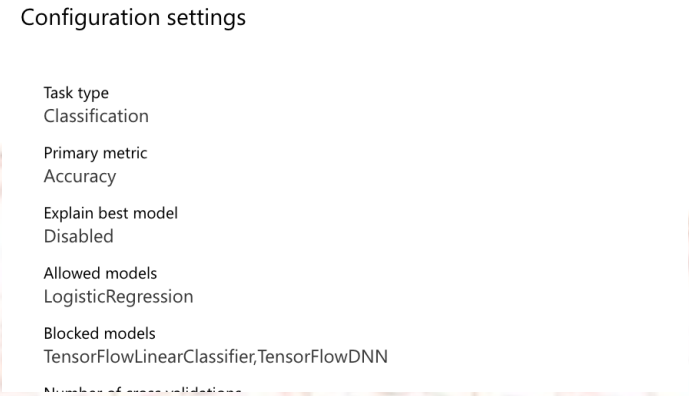
3.3.3.2 Multiclass Logistic Regression Algorithm เป็น Algorithm ที่ใช้สำหรับการจำแนกประเภทข้อมูลที่มากกว่า 2 คลาส (Multiclass Classification) โดยเหมาะสมสำหรับการจำแนกข้อมูลหลายคลาส เช่น การตรวจจับภัยคุกคามทางไซเบอร์ เช่น ข้อความสั้น, การหลอกลวงทางอิเล็กทรอนิกส์ (Phishing) และ ซอฟต์แวร์ที่เป็นอันตราย (Malware)

3.4 ฝึกฝนโมเดลให้เรียนรู้เกี่ยวกับ Smishing (Model Training)

3.4.1 การฝึกฝนโมเดลโดยใช้การ AutoML

3.4.1.1 การฝึกฝนโมเดล AutoML โดยการใช้ Logistic Regression Algorithm ใช้ในการคาดการณ์ผลลัพธ์ที่เป็น ในการตั้งค่าของ Machine Learning สำหรับ การจำแนก ประเภทนี้ การเลือกงานเป็นการจำแนกประเภท (Classification) และใช้ความแม่นยำ (Accuracy) เป็นตัวชี้วัดหลักในการประเมินประสิทธิภาพของโมเดลโดยเลือกโมเดลที่เหมาะสมสำหรับงานนี้ เช่น Logistic

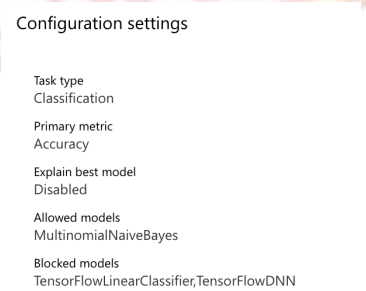
Regression ผ่าน AutoML และการแบ่งข้อมูลแบบสุ่ม ระหว่างชุดฝึกและชุดทดสอบ (Train-Validation Split) เพื่อประเมินประสิทธิภาพในการจำแนกประเภท



ภาพที่ 3-7 ออกแบบโมเดลโดยใช้การ AutoML Logistic Regression Algorithm

3.4.1.2 การฝึกฝนโมเดล AutoML โดยการใช้ MultinomialNaiveBayes Algorithm ในการกำหนด Task ในการฝึกโมเดล Machine Learning นั้นจะเลือกประเภทเป็น Classification ซึ่งเป็นการจำแนกประเภทของข้อมูล เช่น การจำแนกข้อความหรือข้อมูลอื่น ๆ ให้เข้ากลุ่มที่กำหนด โดยใช้ ตัวชี้วัดหลัก (Primary Metric) ในการประเมินประสิทธิภาพของโมเดล โดยเลือกใช้ Accuracy ซึ่งเหมาะสมในการประเมินความแม่นยำของโมเดลที่ใช้ในการจำแนกประเภท สำหรับการกำหนดโมเดลที่ได้รับอนุญาต (Allowed Models) ในการฝึก โดย AutoML จะเลือก MultinomialNaiveBayes Algorithm ซึ่งเป็นโมเดลที่ใช้สำหรับการจำแนกประเภทข้อมูลที่มี ลักษณะเป็นพหุคูณ

ส่วนในกระบวนการแบ่งข้อมูล (Validation Type) เพื่อตรวจสอบประสิทธิภาพของโมเดลจะเลือกใช้การแบ่งข้อมูลสำหรับการฝึกและการตรวจสอบ (Train-ValidationSplit) เพื่อแบ่งข้อมูลออกเป็นชุดฝึก (Training set) สำหรับการฝึกชุดทดสอบ (Validation set) สำหรับการทดสอบความสามารถของโมเดลในการจำแนกประเภท โดยการแบ่งข้อมูลนี้จะทำแบบสุ่มเพื่อให้การประเมินผลเป็นไปอย่างแม่นยำและเป็นธรรม



ภาพที่ 3-8 ออกแบบโมเดลโดยใช้การ AutoML MultinomialNaiveBayes Algorithm

3.4.2 การฝึกฝนโมเดลโดยใช้การ Azure ML Designer

ผู้วิจัยได้เลือกใช้องค์ประกอบ (Componence) ต่าง ๆ อย่างเหมือนกันทั้ง 2 Algorithmก่อนที่จะใช้ Algorithm เป้าหมาย ในกระบวนการจัดการข้อมูล ผู้วิจัยได้ใช้องค์ประกอบการเลือกคอลัมน์ (Componence Select Column) เพื่อกำหนดเป้าหมายให้ชัดเจน และใช้ องค์ประกอบการแปลข้อความเป็นเวกเตอร์ (Componence Text to Vector) เพื่อแปลงข้อความ (Text) ให้อยู่ในรูปแบบของเวกเตอร์ตัวเลข (Numerical Vector Representation) ซึ่งทำให้สามารถใช้งานได้กับโมเดล Machine Learning ต่อไปนอกจากนี้ ผู้วิจัยยังใช้องค์ประกอบการแบ่งข้อมูล (Componence Split Data) เพื่อแบ่งข้อมูลเป็นข้อมูลสำหรับการฝึกโมเดล (Training Data) ร้อยละ 80 และข้อมูลทดสอบ (Test Data) ร้อยละ 20 สำหรับการฝึกรวมทั้งทดสอบโมเดลตามลำดับ

3.4.2.1 ฝึกฝนโมเดลโดยใช้ Multiclass Neural Network Algorithm

3.4.2.2 ฝึกฝนโมเดลโดยใช้ Multiclass Logistic Regression Algorithm

หลังจากนั้นใช้องค์ประกอบการฝึกโมเดล (Componence Train Model) สำหรับฝึกโมเดล และองค์ประกอบการประเมินคะแนนโมเดล (Componence Score Model) เพื่อประเมินประสิทธิภาพของโมเดล

3.5 ประเมินผลโมเดล (Model Evaluation)

การประเมินโมเดลใน Azure AutoML และ Azure Designer มีความยืดหยุ่นและสะดวก โดย AutoML เหมาะสำหรับผู้ที่ต้องการความรวดเร็วในการสร้างโมเดล ส่วน Azure Designer เหมาะสำหรับผู้ที่ต้องการปรับแต่งโมเดลด้วยตนเองการใช้ Accuracy เป็นตัวชี้วัดใน Azure AutoML และ Azure Designer เหมาะสำหรับการจำแนกประเภท (Classification) ค่า Accuracy แสดงถึงความแม่นยำของโมเดลในการทำนายค่าถูกต้อง คำนวณจากอัตราส่วนระหว่างจำนวนการทำนายที่ถูกต้องกับจำนวนทั้งหมด ในการวิจัยนี้ผู้วิจัยมุ่งหวังที่จะเปรียบเทียบค่าความถูกต้องเป็นหลัก จึงให้ความสำคัญกับค่า Accuracy และใช้เป็นตัวชี้วัดของการวิจัยนี้ โดยการตีความ Accuracy ถ้า สูงใกล้ร้อยละ 100 สามารถแปลความหมายได้ว่าโมเดลทำนายได้แม่นยำ หากค่า Accuracy ปานกลาง อยู่ในห่วงร้อยละ 50-70 โมเดลทำนายได้ค่อนข้างดี แต่ยังมีข้อผิดพลาด แต่หากค่าที่ได้ต่ำกว่าร้อยละ 50 นั้นหมายถึงโมเดลมีประสิทธิภาพต่ำ

3.5.1 การประเมินโมเดลใน Azure AutoML

การประเมินโมเดลใน Azure AutoML เป็นขั้นตอนสำคัญที่ช่วยวิเคราะห์ประสิทธิภาพและความแม่นยำของโมเดลที่สร้างขึ้นโดยอัตโนมัติ การประเมินนี้มีบทบาทสำคัญในการเลือกโมเดลที่ดีที่สุดและทำให้มั่นใจว่าโมเดลสามารถตอบโจทย์ธุรกิจหรือวิจัยได้จริง

3.5.1.1 การกำหนดเป้าหมายและตัวชี้วัด (Metrics)

ก่อนการฝึกโมเดล คุณต้องกำหนดเป้าหมาย (Target) และตัวชี้วัด (Metric) ที่เหมาะสมกับประเภทของงาน

3.5.1.1.1 Classification (การจำแนกประเภท)

ก) Accuracy: เปอร์เซ็นต์ของการทำนายที่ถูกต้อง

ข) AUC (Area Under Curve): ความสามารถในการแยกแยะคลาส

ค) F1 Score: ความสมดุลระหว่าง Precision และ Recall

ง) Precision: ความแม่นยำของการทำนายคลาสบวก

จ) Recall: ความครอบคลุมของการทำนายคลาสบวก

3.5.1.2 การตรวจสอบประสิทธิภาพของโมเดล

3.5.1.2.1 คลิกที่ชื่อโมเดลเพื่อดูรายละเอียด

3.5.1.2.2 เลือก Metrics ในหน้าโมเดล

ก) ค่าต่างๆ เช่น Accuracy, AUC, MAE, RMSE จะแสดงในรูปแบบกราฟและตาราง

3.5.1.3 เปรียบเทียบโมเดลหลายตัว

3.5.1.4 ดูค่าสถิติที่ได้จากโมเดลแต่ละตัว คลิกเพื่อดูรายละเอียด และเปรียบเทียบค่าที่สนใจ

3.5.2 การประเมินโมเดลใน Azure Designer

การประเมินโมเดลใน Azure Machine Learning Designer เป็นกระบวนการตรวจสอบประสิทธิภาพของโมเดลโดยใช้โมดูลต่างๆ ที่มีอยู่ใน Azure ML Studio การประเมินนี้ช่วยให้คุณสามารถเปรียบเทียบโมเดลและเลือกโมเดลที่ดีที่สุดสำหรับนำไปใช้งาน

3.5.2.1 โมดูลสำหรับการประเมินโมเดล

3.5.2.1.1 Score Model ใน Azure Machine Learning Designer เป็นโมดูลที่ใช้สำหรับ ทำนายผลลัพธ์ หลังจากที่โมเดลได้รับการฝึก (Training) แล้ว โดยจะทำการทำนายจาก ชุดข้อมูลทดสอบ และแสดง คะแนน (Scores) หรือ ค่าทำนาย ซึ่งสามารถนำไปใช้ประเมินความแม่นยำของโมเดลต่อไป ซึ่งโมดูล Score Model ใช้ทำนายค่าหลังการฝึกโมเดลใน Azure ML Designer ผลลัพธ์ประกอบด้วย Scored Labels และ Scored Probabilities นำผลลัพธ์ไปวิเคราะห์เพิ่มเติมและประเมินโมเดลผ่าน Evaluate Model

3.5.2.1.2 Evaluate Model ใน Azure Machine Learning Designer เป็นโมดูลที่ใช้สำหรับ ประเมินประสิทธิภาพของโมเดล หลังจากที่ได้ทำการทำนายค่า ผ่านโมดูล Score Model แล้ว

3.5.2.2 ขั้นตอนการประเมินโมเดลใน Azure ML Designer

การประเมินโมเดลใน Azure ML Designer เป็นขั้นตอนสำคัญที่ช่วยให้ทราบถึง ประสิทธิภาพ และ ความแม่นยำ ของโมเดลก่อนนำไปใช้งานจริง โดยการประเมินจะใช้โมดูล Score Model และ Evaluate Model เพื่อวิเคราะห์ผลลัพธ์จากการทำนาย

3.5.2.2.1 ขั้นตอนการประเมินโมเดลใน Azure ML Designer

ก) การเตรียมข้อมูล ลากโมดูล Import Data เพื่อดึงข้อมูลจากแหล่งต่างๆ เช่น CSV, Datastore แบ่งข้อมูลเป็นชุดฝึกและชุดทดสอบโดยใช้ Split Data โดย แบ่งเป็นร้อยละ 80 สำหรับการฝึกโมเดล และร้อยละ 20 สำหรับข้อมูลทดสอบ

ข) การฝึกโมเดล (Training Model) ลากโมดูล Train Model มาเชื่อมกับโมดูล Split Data เลือกโมเดลที่ต้องการ ลากโมดูล Algorithm มาเชื่อมกับ Train Model และเชื่อมต่อ Train Model กับข้อมูลฝึก

ค) การทำนายผลลัพธ์ (Scoring) ลากโมดูล Score Model มาเชื่อมกับโมดูล Train Model เชื่อมต่อข้อมูลทดสอบ กับโมดูล Score Model รัน Pipeline เพื่อตรวจสอบผลลัพธ์

3.5.2.3 วิเคราะห์ผลลัพธ์ใน Azure ML Designer

การวิเคราะห์ผลลัพธ์ใน Azure ML Designer เป็นขั้นตอนสำคัญในการตรวจสอบและประเมินประสิทธิภาพของโมเดลหลังจากที่ได้ทำการทำนาย โดยใช้โมดูล Score Model และ Evaluate Model เพื่อเปรียบเทียบผลการทำนายกับค่าจริง

ตารางที่ 3-1 Metrics ที่วิเคราะห์ผลลัพธ์ใน Azure ML Designer

Metric	คำอธิบาย
Accuracy	อัตราการทำนายถูกต้องทั้งหมด
Precision	ความแม่นยำในการทำนายว่าคลาสเป็นบวก
Recall	ความสามารถในการตรวจจับคลาสบวก
F1 Score	ค่าเฉลี่ยระหว่าง Precision และ Recall
AUC (Area Under Curve)	วัดประสิทธิภาพการแยกคลาส

3.5.2.4 เปรียบเทียบโมเดลหลายตัว

การเปรียบเทียบโมเดลใน Azure ML Designer เป็นกระบวนการสำคัญในการเลือกโมเดลที่เหมาะสมที่สุดโดยอิงจากประสิทธิภาพและความแม่นยำ โดย Azure ML Designer รองรับการเปรียบเทียบโมเดลหลายตัวผ่านโมดูล Evaluate Model

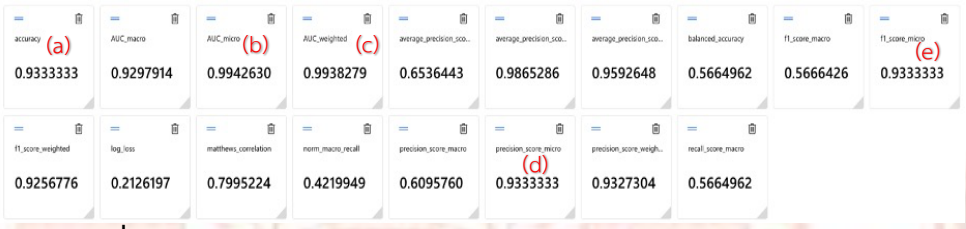
สำหรับงานวิจัยนี้ ดูค่าสถิติที่ได้จากโมเดลแต่ละตัว คลิกเพื่อดูรายละเอียด และเปรียบเทียบค่าที่สนใจ

บทที่ 4
ผลการวิจัย

4.1 ผลการวิจัย

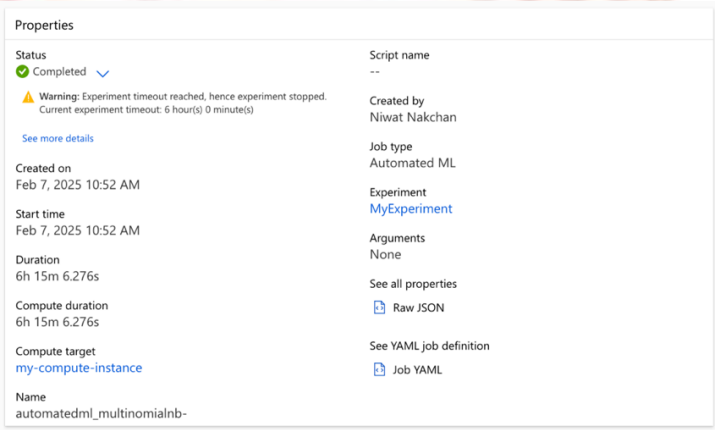
4.1.1 ผลการวิจัย Automated Machine Learning (AutoML)

4.1.1.1 LogisticRegression Algorithm



ภาพที่ 4-1 ผลลัพธ์การวิจัยโดยใช้ AutoML LogisticRegression Algorithm

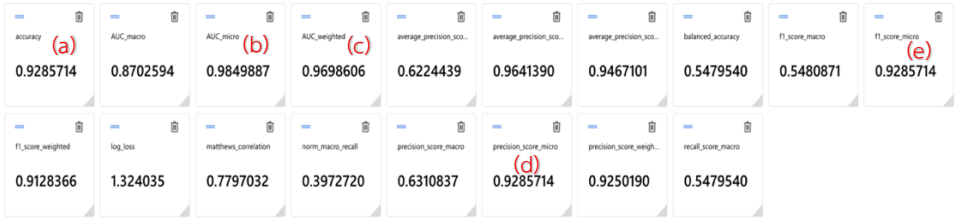
จากรูปภาพที่ 4-1 เป็นผลการประเมินโมเดลจาก Azure AutoML โดยใช้ LogisticRegression Algorithm ซึ่งแสดงค่าตัวชี้วัด (Metrics) หลายตัวที่ใช้ประเมินโมเดล โดยเฉพาะงานประเภท Classification ซึ่งสามารถสรุปได้ว่า โมเดลมี Accuracy (a) สูง ร้อยละ 93.33 แสดงว่าทำนายได้แม่นยำโดยรวม AUC_micro (b) และ AUC_weighted (c) สูง แสดงว่าโมเดลมีประสิทธิภาพดีในการแยกคลาส Precision (d) สูง (โดยเฉพาะ Micro) บ่งบอกว่าโมเดลทำนาย Positive ได้แม่นยำ F1 Score (e) สูง บ่งบอกว่า Precision และ Recall มีความสมดุลในภาพรวม



ภาพที่ 4-2 คุณสมบัติของการวิจัยโดยใช้ AutoML LogisticRegression Algorithm

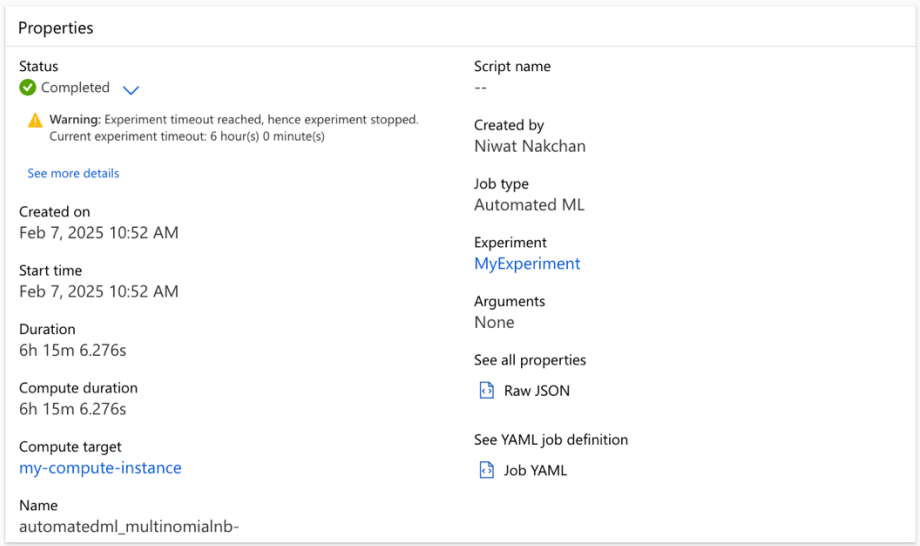
จากรูปภาพที่ 4-2 เป็นคุณสมบัติของการวิจัยโดยใช้ AutoML Logistic Regression Algorithm ซึ่งจะพบว่าระยะเวลาในการประมวลผลสูงถึง 6 ชั่วโมง 15 นาที โดยประมาณ

4.1.1.2 Multinomial Naïve Bayes Algorithm



ภาพที่ 4-3 ผลลัพธ์การวิจัยโดยใช้ AutoML MultinomialNaiveBayes Algorithm

จากรูปภาพที่ 4-3 เป็นผลการประเมินโมเดลจาก Azure AutoML โดยใช้ MultinomialNaiveBayes Algorithm ซึ่งแสดงค่าตัวชี้วัด (Metrics) หลายตัวที่ใช้ประเมินโมเดลโดยเฉพาะงานประเภท Classification ซึ่งสามารถสรุปได้ว่า โมเดลมี Accuracy (a) สูง ร้อยละ 92.28 แสดงว่าทำนายได้แม่นยำโดยรวม AUC_micro (b) และ AUC_weighted (c) สูง แสดงว่าโมเดลมีประสิทธิภาพดีในการแยกคลาส Precision (d) สูง (โดยเฉพาะ Micro) บ่งบอกว่าโมเดลทำนาย Positive ได้แม่นยำ F1 Score (e) สูง บ่งบอกว่า Precision และ Recall มีความสมดุลในภาพรวม



ภาพที่ 4-4 คุณสมบัติของการวิจัยโดยใช้ AutoML MultinomialNaiveBayes Algorithm

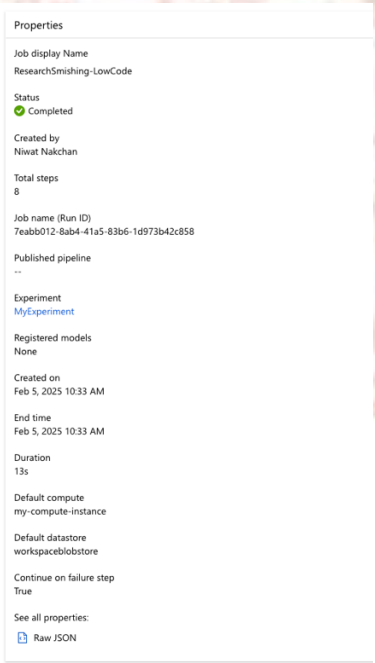
จากรูปภาพที่ 4-4 เป็นคุณสมบัติของการวิจัยโดยใช้ AutoML Multinomial NaiveBayes Algorithm ซึ่งจะพบว่าระยะเวลาในการประมวลผมสูงถึง 6 ชั่วโมง 15 นาที โดยประมาณ

4.1.2 ผลการวิจัย Azure ML Designer

4.1.2.1 Multiclass Neural Network Algorithm

ภาพที่ 4-5 ผลลัพธ์การวิจัยโดยใช้ Azure ML Designer Multiclass Neural Network Algorithm

จากรูปภาพที่ 4-5 เป็นผลการประเมินโมเดลจาก Azure ML Designer โดยใช้ Multiclass Neural Network Algorithm ค่าเหล่านี้เป็นตัวชี้วัดประสิทธิภาพของโมเดลการเรียนรู้ของเครื่อง (Machine Learning) โดยเฉพาะอย่างยิ่งในงานจำแนกประเภท (Classification) ซึ่งสามารถสรุปได้ว่า โมเดลมี ค่า Micro Precision, Micro Recall และ Overall Accuracy ที่สูง ร้อยละ 82 แสดงว่าโมเดลทำงานได้ดีเมื่อพิจารณาภาพรวม



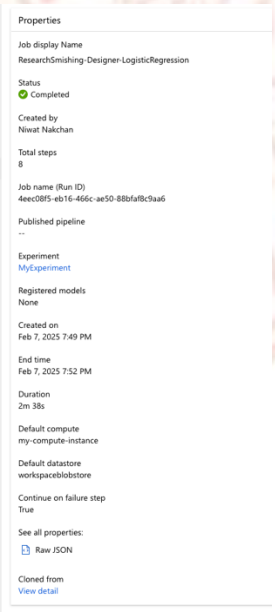
ภาพที่ 4-6 คุณสมบัติการวิจัยโดยใช้ Azure ML Designer Multiclass Neural Network Algorithm

จากรูปภาพที่ 4-6 เป็นคุณสมบัติของการวิจัยโดยใช้ Azure ML Designer โดยใช้ Multiclass Neural Network Algorithm ซึ่งจะพบว่าระยะเวลาในการประมวลผล 2 นาที 38 วินาที โดยประมาณ

4.1.2.2 Multiclass Logistic Regression Algorithm

ภาพที่ 4-7 ผลลัพธ์การวิจัยโดยใช้ Azure ML Designer Multiclass Logistic Regression Algorithm

จากรูปภาพที่ 4-7 เป็นผลการประเมินโมเดลจาก Azure ML Designer โดยใช้ Multiclass Logistic Regression Algorithm ค่าเหล่านี้เป็นตัวชี้วัดประสิทธิภาพของโมเดลการเรียนรู้ของเครื่อง (Machine Learning) โดยเฉพาะอย่างยิ่งในงานจำแนกประเภท (Classification) ซึ่งสามารถสรุปได้ว่า โมเดลมี ค่า Micro Precision, Micro Recall และ Overall Accuracy ที่สูง ร้อยละ 82 แสดงว่าโมเดลทำงานได้ดีเมื่อพิจารณาภาพรวม



ภาพที่ 4-8 คุณสมบัติการวิจัยโดยใช้ Azure ML Designer Multiclass Logistic Regression Algorithm

จากรูปภาพที่ 4-8 เป็นคุณสมบัติของการวิจัยโดยใช้ Azure ML Designer โดยใช้ Multiclass Neural Network Algorithm ซึ่งจะพบว่าระยะเวลาในการประมวลผล 2 นาที 38 วินาที โดยประมาณ

4.2 การเปรียบเทียบโมเดล Machine Learning สำหรับการตรวจจับ Smishing

การตรวจจับ Smishing (SMS Phishing) เป็นงานที่ท้าทายในการประมวลผลข้อความที่มีการหลอกลวงผ่านทางข้อความ SMS โดยการใช้เทคนิค Machine Learning (ML) เพื่อจำแนกข้อความที่มีความเสี่ยงและไม่เสี่ยงเป็นวิธีที่มีประสิทธิภาพในการลดความเสี่ยงจากการโจมตีดังกล่าว การศึกษานี้จะทำการเปรียบเทียบโมเดลต่างๆ ที่พัฒนาโดยการใช้เครื่องมือ Automated Machine Learning (AutoML) และ Azure ML Designer ในการประมวลผลข้อความและคำนวณผลลัพธ์การทำนายโดยใช้ Accuracy, Micro Precision, และ Compute Duration เป็นตัวชี้วัดหลักในการประเมิน ซึ่งเป็นปัจจัยสำคัญในการประเมินประสิทธิภาพของแต่ละแนวทาง

4.2.1 ภาพรวมของโมเดลที่ใช้ในการตรวจจับ Smishing

โมเดลที่ถูกเปรียบเทียบในงานนี้ถูกสร้างขึ้นโดยใช้สองเครื่องมือหลัก ได้แก่ Automated Machine Learning (AutoML) และ Azure ML Designer โดยมีการใช้ Logistic Regression, Multinomial Naive Bayes, Multiclass Neural Network, และ Multiclass Logistic Regression ในการสร้างโมเดลสำหรับการจำแนกข้อความที่เป็น Smishing หรือไม่

4.2.2 การเปรียบเทียบผลลัพธ์

ผลลัพธ์ที่ได้รับจากการเปรียบเทียบโมเดลมีการคำนวณโดยใช้สามค่าหลักที่สำคัญ ได้แก่ Accuracy, Micro Precision, และ Compute Duration ซึ่งมีรายละเอียดดังนี้

ตารางที่ 4-1 การเปรียบเทียบประสิทธิภาพของโมเดล Machine Learning

Authoring	Algorithm Type	Accuracy	Micro Precision	ComputeDuration
Automated	LogisticRegression	93%	93%	6h 15m 6.276s
Machine	MultinomialNaiveBayes	92%	92%	6h 15m 6.276s
Azure ML	Multiclass Neural Network	82%	82%	3m 15s
Designer	Multiclass Logistic Regression	82%	82%	2m 38s

4.2.3 การประเมินผลของโมเดล

4.2.3.1 ค่าความความแม่นยำ (Accuracy)

Accuracy เป็นตัวชี้วัดที่สำคัญในการประเมินประสิทธิภาพของโมเดล เนื่องจากเป็นตัววัดที่สะท้อนถึงสัดส่วนของการทำนายที่ถูกต้องเมื่อเทียบกับการทำนายทั้งหมด โดยค่า Accuracy ของโมเดลที่สร้างขึ้นจาก AutoML มีค่าที่สูงที่สุดเมื่อใช้ Logistic Regression ที่ร้อยละ 93 ตามมาด้วย

Multinomial Naive Bayes ที่ร้อยละ 92 ซึ่งแสดงให้เห็นว่าโมเดลเหล่านี้มีความสามารถในการทำนายข้อมูลได้อย่างแม่นยำและน่าเชื่อถือ

ในขณะที่โมเดลที่พัฒนาโดยใช้ Azure ML Designer ไม่สามารถทำคะแนนใน Accuracy ได้สูงเท่ากับ AutoML โดยโมเดล Multiclass Neural Network และ Multiclass Logistic Regression ทำคะแนน Accuracy ได้เพียงร้อยละ 82 ซึ่งแสดงให้เห็นถึงความแตกต่างในประสิทธิภาพของโมเดลที่ใช้ AutoML และ Azure ML Designer ในการทำงานที่ซับซ้อนอย่างการตรวจจับ Smishing

4.2.3.2 ความแม่นยำโดยรวม (Micro Precision)

Micro Precision เป็นการวัดความแม่นยำในการทำนายข้อมูลบวกในกรณีที่โมเดลมีหลายคลาส โดยคำนวณจากการรวม True Positives และ False Positives ทั้งหมดในทุกคลาส จากตารางเปรียบเทียบพบว่า AutoML มี Micro Precision ที่สูงที่สุดเมื่อใช้ Logistic Regression ที่ร้อยละ 93 และ Multinomial Naive Bayes ที่ร้อยละ 92 ซึ่งแสดงให้เห็นว่าโมเดลเหล่านี้มีความสามารถในการทำนายข้อความ Smishing และข้อความที่ไม่เป็น Smishing ได้อย่างแม่นยำ

ในทางกลับกัน โมเดลที่พัฒนาโดยใช้ Azure ML Designer มีค่า Micro Precision เท่ากับร้อยละ 82 สำหรับทั้ง Multiclass Neural Network และ Multiclass Logistic Regression ซึ่งแสดงให้เห็นว่าโมเดลเหล่านี้มีผลลัพธ์ที่ต่ำกว่าในแง่ของความแม่นยำในการทำนายข้อความที่เป็น Smishing

4.2.3.2 ระยะเวลาในการฝึกโมเดล (Compute Duration)

การวัดระยะเวลาในการฝึกโมเดล (Compute Duration) เป็นตัวชี้วัดที่สำคัญในแง่ของประสิทธิภาพในการใช้ทรัพยากรของระบบในการฝึกโมเดล โดย AutoML ใช้เวลามากที่สุดในการฝึกโมเดลที่ 6 ชั่วโมง 15 นาที 6.276 วินาที สำหรับทั้ง Logistic Regression และ Multinomial Naive Bayes ซึ่งสะท้อนให้เห็นถึงการประมวลผลที่ต้องใช้เวลานานกว่าโมเดลที่พัฒนาด้วย Azure ML Designer

ในขณะที่ Azure ML Designer ใช้เวลาฝึกโมเดลที่รวดเร็วกว่าอย่างเห็นได้ชัด โดย Multiclass Neural Network ใช้เวลา 3 นาที 15 วินาที และ Multiclass Logistic Regression ใช้เวลา 2 นาที 38 วินาที ซึ่งทำให้ Azure ML Designer เป็นตัวเลือกที่ดีกว่าในแง่ของการลดระยะเวลาในการฝึกโมเดล

4.2.4 การเปรียบเทียบข้อดีและข้อจำกัดของแต่ละโมเดล

4.2.4.1 Automated Machine Learning (AutoML)

4.2.4.1.1 ข้อดี

ก) ให้ผลลัพธ์ที่ดีในทั้ง Accuracy และ Micro Precision ซึ่งเหมาะสำหรับการตรวจจับ Smishing ที่มีข้อมูลที่ซับซ้อน

ข) การเลือกอัลกอริธึมที่เหมาะสมได้อัตโนมัติ โดยไม่จำเป็นต้องตั้งค่าด้วยตนเอง

ค) สามารถปรับปรุงประสิทธิภาพได้โดยการเลือกหลายอัลกอริธึมและการปรับแต่งพารามิเตอร์อัตโนมัติ

4.2.4.1.2 ข้อจำกัด

ก) ใช้เวลานานในการฝึกโมเดล ซึ่งอาจไม่เหมาะสำหรับการทำงานที่ต้องการประมวลผลข้อมูลขนาดใหญ่ในเวลาสั้น

ข) การใช้งานทรัพยากรสูง ซึ่งอาจทำให้ค่าใช้จ่ายในการประมวลผลสูงขึ้น

4.2.4.2 Azure Machine Learning Designer (Azure ML Designer)

4.2.4.2.1 ข้อดี

ก) การฝึกโมเดลเร็วและมีประสิทธิภาพในด้านการใช้ทรัพยากร

ข) ง่ายในการสร้างและทดสอบโมเดลที่ไม่ซับซ้อน

ค) เหมาะสำหรับการทดลองเบื้องต้นและงานที่มีการใช้ข้อมูลที่ไม่ซับซ้อนมาก

4.2.4.2.2 ข้อจำกัด

ก) ผลลัพธ์ในเรื่องของ Accuracy และ Micro Precision ไม่ดีเท่ากับ AutoML

ข) โมเดลที่ใช้ไม่ได้รับการปรับปรุงหรือเลือกโดยอัตโนมัติ ซึ่งอาจจำกัดประสิทธิภาพในงานที่ต้องการความแม่นยำสูง

จากการเปรียบเทียบผลลัพธ์ที่ได้จากการใช้ AutoML และ Azure ML Designer สำหรับการตรวจจับ Smishing พบว่า AutoML โดยใช้ Logistic Regression และ Multinomial Naive Bayes มีประสิทธิภาพดีกว่าในแง่ของ Accuracy และ Micro Precision แต่ต้องแลกกับการใช้เวลาฝึกโมเดลที่นานกว่าและใช้ทรัพยากรสูงกว่า ในขณะที่ Azure ML Designer แม้จะมีเวลาในการฝึกโมเดลที่สั้นกว่า แต่ผลลัพธ์ในด้าน Accuracy และ Micro Precision ยังคงต่ำกว่า AutoML ในการตรวจจับ Smishing การเลือกใช้เครื่องมือขึ้นอยู่กับลักษณะของงานและข้อจำกัดด้านทรัพยากรและเวลาในการฝึกโมเดล

บทที่ 5
สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การวิจัยนี้มุ่งศึกษาการใช้เครื่องมือ Azure Machine Learning (AZ ML Studio) ในการสร้างโมเดลการตรวจจับ Smishing ซึ่งเป็นรูปแบบหนึ่งของการโจมตีที่ผ่านการหลอกลวงทางข้อความ โดยการใช้ Data ที่มีการแปลงข้อมูลด้วย Word2Vec และการเลือกใช้ No-code Machine Learning เพื่อเปรียบเทียบประสิทธิภาพของอัลกอริธึมต่าง ๆ ที่ได้จาก Automated Machine Learning (AutoML) และ Azure ML Designer

เลือก Azure ML Studio ใช้เป็นเครื่องมือในการสร้างโมเดล Machine Learning ในการตรวจจับ Smishing เนื่องจากมีคุณสมบัติที่เหมาะสมกับการทำงานแบบไม่ต้องเขียนโค้ด ซึ่งช่วยให้กระบวนการพัฒนาโมเดลเป็นไปอย่างรวดเร็วและไม่ซับซ้อน นอกจากนี้ยังรองรับการใช้โมดูลที่หลากหลายในการประมวลผลและการเลือกโมเดล รวมถึงสามารถทำงานร่วมกับ Azure Cloud ในการประมวลผลข้อมูลและฝึกโมเดลได้สะดวก

หา Data Smishing จาก Public Data ข้อมูลที่ใช้ในการฝึกโมเดลมาจากแหล่งข้อมูลสาธารณะ (Public Data) ซึ่งมีข้อความจาก Smishing หรือการโจมตีผ่านข้อความหลอกลวง ข้อมูลเหล่านี้จะถูกนำมาผ่านขั้นตอนการเตรียมและแปลงก่อนที่จะนำไปฝึกโมเดลใน Azure ML Studio

แปลง Data จาก Word2Vec การแปลงข้อมูลข้อความ (Text) เป็นเวกเตอร์ (Vector) ใช้ Word2Vec ซึ่งเป็นเทคนิคการแปลงคำให้เป็นเวกเตอร์ในพื้นที่ตัวเลข เทคนิคนี้ช่วยให้เครื่องมือ Machine Learning สามารถเข้าใจข้อความในรูปแบบที่เหมาะสมกับการคำนวณโมเดล เนื่องจากคำที่มีความหมายใกล้เคียงกันจะมีเวกเตอร์ที่ใกล้เคียงกัน ซึ่งช่วยให้การตรวจจับการหลอกลวงมีประสิทธิภาพมากยิ่งขึ้น

ใช้วิธี No-Code ใน AZ ML การใช้ No-code Machine Learning ใน Azure ML Studio ทำให้การสร้างโมเดล Smishing Detection เป็นเรื่องง่ายและรวดเร็ว โดยผู้ใช้ไม่จำเป็นต้องเขียนโค้ดเอง เครื่องมือ Drag-and-Drop ใน Azure ML Designer ช่วยให้ผู้ใช้สามารถจัดการกับข้อมูลและเลือกอัลกอริธึมต่าง ๆ ได้อย่างง่ายดาย ทำให้กระบวนการพัฒนาโมเดลเร็วขึ้นและเข้าถึงได้ง่ายสำหรับผู้ที่ไม่มีทักษะการเขียนโปรแกรม

ใช้ Algorithm Type โดยอัลกอริธึมที่เลือกใช้ในการฝึกโมเดล คือ Automated Machine Learning (AutoML) โดยใช้ Logistic Regression และ Multinomial Naive Bayes ส่วน Azure ML Designer เลือกใช้ Multiclass Neural Network และ Multiclass Logistic Regression

ผลลัพธ์ที่ได้จากการทดลอง

Logistic Regression ใน AutoML ได้ Accuracy และ Micro Precision สูงสุดที่ร้อยละ 93 ซึ่งแสดงถึงความสามารถในการจำแนกข้อมูลที่ดี

Multinomial Naive Bayes ใน AutoML ได้ Accuracy และ Micro Precision ที่ร้อยละ92 ซึ่งใกล้เคียงกับ Logistic Regression

โมเดลใน Azure ML Designer ที่ใช้ Multiclass Neural Network และ Multiclass Logistic Regression ได้ Accuracy และ Micro Precision เท่ากับร้อยละ 82 ซึ่งต่ำกว่าโมเดลที่ได้จาก AutoML

5.2 การอภิปรายผล

5.2.1 ประสิทธิภาพของโมเดล

AutoML สามารถสร้างโมเดลที่มีประสิทธิภาพสูงกว่า Azure ML Designer โดยทั้ง Logistic Regression และ Multinomial Naive Bayes ได้ผลลัพธ์ที่ดีทั้งในแง่ของ Accuracy และ Precision ซึ่งอาจเนื่องจาก AutoML สามารถทำการปรับแต่งและเลือกอัลกอริธึมที่ดีที่สุดสำหรับข้อมูลได้โดยอัตโนมัติ ในขณะที่ Azure ML Designer ต้องพึ่งการเลือกโมเดลจากตัวเลือกที่มีให้ ซึ่งอาจไม่สามารถจับลักษณะของข้อมูลได้ดีเท่ากับ AutoML

5.2.2 เวลาที่ใช้ในการคำนวณ

การฝึกโมเดลใน AutoML ใช้เวลานานถึง 6 ชั่วโมง 15 นาที ซึ่งอาจเกิดจากการที่ AutoML มีการเลือกอัลกอริธึมที่หลากหลายและใช้เวลาคำนวณหลายรอบเพื่อหาค่าที่ดีที่สุด ในขณะที่การฝึกโมเดลใน Azure ML Designer เช่น Multiclass Logistic Regression ใช้เวลาเพียง 2 นาที 38 วินาที ซึ่งเร็วกว่ามาก เนื่องจาก Azure ML Designer ใช้โมเดลที่ถูกเลือกไว้ล่วงหน้าและมีการคำนวณที่น้อยกว่า

5.2.3 ความแม่นยำ

ความแม่นยำของโมเดลจาก AutoML สูงกว่าผลลัพธ์จาก Azure ML Designer โดยเฉพาะ Logistic Regression ที่ได้ Accuracy สูงสุดที่ร้อยละ 93 ซึ่งแสดงถึงการทำงานที่มีประสิทธิภาพสูงในการตรวจจับ Smishing

5.3 ข้อเสนอแนะ

5.3.1 การใช้ AutoML สำหรับข้อมูลที่ซับซ้อน

หากมีข้อมูลที่ซับซ้อนและต้องการประสิทธิภาพสูง ควรใช้ AutoML เนื่องจากมันสามารถเลือกอัลกอริธึมที่เหมาะสมและปรับแต่งโมเดลได้โดยอัตโนมัติ

5.3.2 การใช้ Azure ML Designer สำหรับงานที่ไม่ซับซ้อน

หากงานมีลักษณะที่ไม่ซับซ้อนหรือเวลาในการฝึกโมเดลเป็นสิ่งที่สำคัญ ควรพิจารณาการใช้ Azure ML Designer เนื่องจากมันใช้เวลาในการคำนวณน้อยและมีความรวดเร็วในการฝึกโมเดล

5.3.3 การปรับปรุงเวลาในการคำนวณ

ควรพิจารณาการใช้ทรัพยากรการคำนวณที่มีประสิทธิภาพสูงกว่าใน Azure หรือใช้การฝึกโมเดลแบบขนานเพื่อลดเวลาการคำนวณและเพิ่มประสิทธิภาพการทำงาน

5.3.4 การทดลองอัลกอริธึมที่หลากหลาย

ควรทำการทดลองอัลกอริธึมที่หลากหลายมากขึ้นและทดสอบโมเดลเพิ่มเติมใน Azure ML Designer และ AutoML เพื่อหาวิธีที่ดีที่สุดในการตรวจจับ Smishing ที่เหมาะสมที่สุดสำหรับชุดข้อมูลที่ใช้

การวิจัยนี้แสดงให้เห็นถึงประสิทธิภาพของ AutoML ที่สามารถให้ผลลัพธ์ที่ดีกว่า Azure ML Designer ในการตรวจจับ Smishing โดย Logistic Regression ได้ Accuracy สูงถึงร้อยละ 93 และ Micro Precision ที่ร้อยละ 93 อย่างไรก็ตามการใช้เวลาในการคำนวณที่นานใน AutoML ก็เป็นข้อจำกัดที่ต้องพิจารณา โดยการเลือกเครื่องมือที่เหมาะสมกับความต้องการของงานจะช่วยให้ได้ผลลัพธ์ที่ดีที่สุดในเวลาอันสั้น

แพลตฟอร์ม No-Code มีฟังก์ชันสำเร็จรูปให้ใช้งาน แต่ไม่สามารถปรับแต่งโมเดลได้อย่างอิสระเท่ากับการใช้โค้ดโดยตรง เครื่องมือ No-Code มักมีข้อจำกัดด้าน รูปแบบข้อมูล และ ขนาดของชุดข้อมูล ที่สามารถรองรับได้ อัลกอริธึมบางประเภทอาจถูกจำกัดอยู่ในขอบเขตที่แพลตฟอร์มรองรับ และ อาจไม่สามารถปรับปรุงประสิทธิภาพของโมเดลได้ดีพอ ควรพิจารณาการใช้โค้ดแบบ LowCode เพื่อให้สามารถปรับแต่งโมเดลและประสิทธิภาพได้สูงสุด ซึ่งสามารถเพิ่มโค้ดแบบ Custom ได้ Low-Code เหมาะสำหรับการพัฒนาแอปหรือ Machine Learning ที่ต้องการความเร็วและความยืดหยุ่น โดยยังสามารถ ปรับแต่งโค้ดได้บางส่วน ซึ่งเป็นทางเลือกที่สมดุลระหว่าง No-Code และ Full-Code

ปัจจุบันซอฟต์แวร์ No-Code/Low-Code ได้รับความนิยมอย่างมาก เนื่องจากช่วยลดความซับซ้อนในการพัฒนาแอปพลิเคชัน โดยไม่จำเป็นต้องเขียนโค้ดหรือเขียนโค้ดเพียงเล็กน้อย เหมาะสำหรับผู้ที่ไม่มีทักษะการเขียนโปรแกรมหรือองค์กรที่ต้องการลดเวลาและค่าใช้จ่ายในการพัฒนา มีการใช้งานในหลายด้าน เช่น การพัฒนาเว็บแอป, แอปมือถือ, ระบบอัตโนมัติ (Automation) และการวิเคราะห์ข้อมูล

ซอฟต์แวร์ No-Code/Low-Code แบบ Non-Commercial (โอเพ่นซอร์ส/ฟรี) ในปัจจุบันมีซอฟต์แวร์ No-Code/Low-Code แบบโอเพ่นซอร์สและใช้งานฟรีหลายตัวที่สามารถนำไปใช้ในการพัฒนาแอปพลิเคชันโดยไม่เสียค่าใช้จ่าย เหมาะสำหรับนักพัฒนาอิสระ กลุ่มนักศึกษา หรือองค์กรไม่แสวงหากำไร เช่น Budibase [31] NocoDB [32] Appsmith [33] Node-RED [35] เป็นต้น

ซอฟต์แวร์ No-Code/Low-Code แบบ Non-Commercial ที่กล่าวถึงข้างต้นมีความยืดหยุ่นและสามารถนำไปใช้ได้โดยไม่มีค่าใช้จ่าย เหมาะสำหรับผู้ที่ต้องการพัฒนาแอปพลิเคชันภายในองค์กรหรือเพื่อการเรียนรู้ โดยแพลตฟอร์มอย่าง Budibase, NocoDB, และ Appsmith เหมาะสำหรับสร้างเครื่องมือภายใน ขณะที่ Node-RED เหมาะกับระบบ IoT และการเชื่อมต่อบริการต่าง ๆ

การเลือกใช้ขึ้นอยู่กับวัตถุประสงค์ เช่น การสร้างแอปธุรกิจ, การจัดการข้อมูล, หรือการควบคุมอุปกรณ์อัจฉริยะ โดยทั้งหมดนี้สามารถโฮสต์บนเซิร์ฟเวอร์ของตนเองได้ จึงมีความยืดหยุ่นสูงในด้านการปรับแต่งและความปลอดภัย

บรรณานุกรม

1. Bangkokbiznews. (2022). [ออนไลน์]. เปิดสถิติใช้ “ดิจิทัล” ทั่วโลก “ไทย” ติดอันดับโลก เพียบ!! สืบค้นจาก https://www.bangkokbiznews.com
2. สุรศักดิ์ ภักดีวัฒนะกุล. (2016). [ออนไลน์]. เทคโนโลยีโทรศัพท์เคลื่อนที่ (Mobile Technology) สามารถเพิ่มผลการตอบสนองกลับและการมีส่วนร่วมของลูกค้าได้ สืบค้นจาก https://www.callcentermaster.com
3. เรวัต แสงสุริยงค์. (2018). [ออนไลน์]. ความพยายามลดช่องว่างด้านดิจิทัลในสังคมไทย สืบค้นจาก Veridian E-Journal, Silpakorn University
3. KuCoin Learn. (2025). [ออนไลน์]. What Is Smishing, and How to Safeguard Yourself Against It ? สืบค้นจาก https://www.kucoin.com
4. Guzella, T. S., & Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. Expert Systems with Applications, สืบค้นจาก https://www.researchgate.net
5. กระทรวงดิจิทัลเพื่อเศรษฐกิจและสังคม. (2016). [ออนไลน์]. รู้ทัน Scammer ภัยออนไลน์ใกล้ตัว สืบค้นจาก https://www.mdes.go.th
6. ACIS Professional. (2025). [ออนไลน์]. ภัยคุกคามทางโทรศัพท์ของ Scammer: ภัยเงียบที่ต้องระวัง สืบค้นจากhttps://www.acisonline.net
7. สภาองค์กรของผู้บริโภค. (2021). [ออนไลน์]. ข้อเสนอต่อกรณี SMS ละเมิดสิทธิผู้บริโภค สืบค้นจาก https://www.tcc.or.th
8. Government Contact Center. (2024). [ออนไลน์]. ลักษณะคดีในรูปแบบการกระทำผิดอาชญากรรมทางเทคโนโลยี สืบค้นจาก https://www.gcc.go.th
9. Electronic Transactions Development Agency. (2018). [ออนไลน์]. การหลอกลวงทางอินเทอร์เน็ต (Scam) สืบค้นจาก https://www.etda.or.th
10. สำนักงานตำรวจแห่งชาติ (Royal Thai Police) (2023). [ออนไลน์]. การหลอกลวงผ่านข้อความสั้น (Smishing). สืบค้นจาก https://www.royalthaipolice.go.th
11. กระทรวงดิจิทัลเพื่อเศรษฐกิจและสังคม (MDES) (2022). [ออนไลน์]. การป้องกันภัยจากการหลอกลวงผ่านช่องทางดิจิทัล. สืบค้นจาก https://www.mdes.go.th
12. กระทรวงเทคโนโลยีสารสนเทศและการสื่อสาร (ETDA) (2023). [ออนไลน์]. ความรู้เกี่ยวกับการหลอกลวงทางออนไลน์. สืบค้นจาก https://www.etda.or.th

13. กระทรวงดิจิทัลเพื่อเศรษฐกิจและสังคม. (2023). [ออนไลน์]. การป้องกันภัยจากการหลอกลวงผ่านช่องทางดิจิทัล. สืบค้นจาก https://www.mdes.go.th
14. สำนักงานตำรวจแห่งชาติ. (2023). [ออนไลน์]. การหลอกลวงผ่านข้อความสั้น: แนวทางการป้องกันและการตรวจจับ. สืบค้นจาก https://www.royalthaipolice.go.th
15. สำนักงานพัฒนาธุรกรรมทางอิเล็กทรอนิกส์ (ETDA). (2023). [ออนไลน์]. ภัยจากการหลอกลวงผ่านโทรศัพท์และข้อความสั้น (Smishing) และวิธีการป้องกัน. สืบค้นจาก https://www.etda.or.th
16. กรมสอบสวนคดีพิเศษ. (2023). [ออนไลน์]. การตรวจจับและป้องกันการหลอกลวงผ่านเทคโนโลยีสารสนเทศในประเทศไทย. สืบค้นจาก https://www.dsi.go.th
17. กระทรวงดิจิทัลเพื่อเศรษฐกิจและสังคม (MDES). (2023). [ออนไลน์].แนวทางการพัฒนาแอปพลิเคชันโดยใช้เครื่องมือ Low-Code/No-Code. สืบค้นจาก https://www.mdes.go.th
18. สำนักงานพัฒนาธุรกรรมทางอิเล็กทรอนิกส์ (ETDA). (2023). [ออนไลน์]. เครื่องมือที่ช่วยให้การพัฒนาแอปพลิเคชันเป็นเรื่องง่ายโดยไม่ต้องเขียนโค้ด. สืบค้นจาก https://www.etda.or.th
19. กรมส่งเสริมอุตสาหกรรม. (2023). [ออนไลน์]. เครื่องมือ Low-Code/No-Code: การพัฒนาแอปพลิเคชันที่ไม่ต้องเขียนโค้ด. สืบค้นจาก https://www.dip.go.th
20. Ramanujam, E., Shankar, K., Sharma, A. (2024). [ออนไลน์]. A Review on Artificial IntelligenceTechniques for Multilingual SMS Spam Detection สืบค้นจาก https://www.researchgate.net/
21. Rish, I. (2001). [ออนไลน์]. An empirical study of the naive Bayes classifier. สืบค้นจาก https://www.researchgate.net/
22. A. I. B. Awan, A. M. Shaikh, and M. M. Aslam. (2018) Naive Bayes vs Logistic Regression for Text Classification. Proceedings of the International Conference on Computer Science and Information Technology (CSIT), 2018. สืบค้นจาก https://www.researchgate.net/

23. K. C. L. K. Mohan and M. P. N. Suraj, A Comparative Analysis of Naive Bayes and Logistic Regression for Sentiment Analysis. International Journal of Computer Applications. สืบค้นจาก https://www.researchgate.net/

24. D. S. C. R. Mohan, (2019). [ออนไลน์]. Introduction to Azure Machine Learning สืบค้นจาก https://www.jcse.org/

25. Ramanujam, E., Shankar, K., Sharma, A. (2024). [ออนไลน์]. A Review on Artificial Intelligence Techniques for Multilingual SMS Spam Detection. สืบค้นจาก https://www.researchgate.net/

26. Roy, Pradeep Kumar, Singh, Jyoti Prakash, Banerjee, Snehasish. (2020). [ออนไลน์] Deep learning to filter SMS Spam. สืบค้นจาก https://www.researchgate.net/

27. Almeida, Tiago A., Hidalgo, José María G., Yamakami, Akebo. (2011). [ออนไลน์] Contributions to the study of SMS spam filtering: New collection and results. สืบค้นจาก https://www.researchgate.net/

28. Padmaja, B. , Bala, M.M. , Patro, E.K.R. (2024). [ออนไลน์] An intelligent auto-response short message service categorization model using semantic index. สืบค้นจาก https://www.researchgate.net/

29. D. M. Williams, (2021). [ออนไลน์] The Rise of Low-Code Platforms for Machine Learning. สืบค้นจาก https://www.journalaidatascience.com

30. J. B. Patel, S. D. Kumawat, and M. K. Gupta, (2020). [ออนไลน์] A Survey on No-Code and Low-Code Development Platforms for AI and Machine Learning. สืบค้นจาก https://www.journalaidatascience.com

31. Budibase. [2026]. [ออนไลน์] Budibase The low code platform you'll enjoy using. สืบค้นจาก https://github.com/

32. NocoDB. [2026]. [ออนไลน์] NocoDB The Open Source Airtable Alternative สืบค้นจาก https://github.com/

33. Appsmith. [2026]. [ออนไลน์]. Appsmith in 100 second สืบค้นจาก https://github.com/

34. Node-RED. [2026]. [ออนไลน์]. Low-code programming for event-driven applications สืบค้นจาก https://nodered.org

ประวัติผู้เขียน

ชื่อ นายนิวัฒน์ นาคจันทร์

ชื่อการค้นคว้าอิสระ การประยุกต์ใช้โมเดลการเรียนรู้ของเครื่องโดยไม่ต้องเขียนโค้ดสำหรับการตรวจจับการหลอกลวงผ่านข้อความสั้น

สาขาวิชา การบริหารเครือข่ายและเทคโนโลยีสารสนเทศ

ประวัติ ประวัติการศึกษา

พ.ศ. 2560 สำเร็จการศึกษาระดับปริญญาตรี วิทยาศาสตรบัณฑิต สาขาเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีปทุมวัน

ประวัติการทำงาน

พ.ศ.2550 - พ.ศ.2551 ตำแหน่ง Desktop Engineer, บริษัท Atos Origin(Thailand) จำกัด

พ.ศ.2551 - พ.ศ.2558 ตำแหน่ง Desksite Support, บริษัท Manpower(Thailand) จำกัด

พ.ศ.2558 - พ.ศ.2562 ตำแหน่ง Advanced Operations Specialist บริษัท ไอบีเอ็ม โซลูชั่น ดิลิเวอรี่ จำกัด

พ.ศ.2562 - ปัจจุบัน ตำแหน่ง Senior IT Desktop & Client Management ธนาคารกรุงศรีอยุธยา จำกัด (มหาชน)