การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

นายณัฐวีร์ พูลสมบัติภิญโญ

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2568

การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

นายณัฐวีร์ พูลสมบัติภิญโญ

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2568

ใบรับรองโครงร่างการค้นคว้าอิสระ

บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

เรื่อง การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

โดย นายณัฐวีร์ พูลสมบัติภิญโญ

ได้รับอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ

คณบดีบัณฑิตวิทยาลัย / หัวหน้าภาควิชา

.....

(สุนันฑา สดสี)

คณะกรรมการสอบการค้นคว้าอิสระ

..... ประธานกรรมการ

(สุมิตรา นวลมีศรี)

..... อาจารย์ที่ปรึกษา

(พงศ์ศรัณย์ บุญโญปกรณ์)

..... กรรมการ

(พงศ์ศรัณย์ บุญโญปกรณ์)

..... กรรมการ

(นวพร วิสิฐพงศ์พันธ์)

ชื่อ : นายณัฐวีร์ พูลสมบัติภิญโญ

ชื่อการค้นคว้าอิสระ :

การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

สาขาวิชา : การบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก : พงศ์ศรัณย์ บุญโญปกรณ์

ปีการศึกษา : 2568

บทคัดย่อ

การวิจัยนี้เป็นการเสนอแนวคิดและจำลองระบบการยืนยันตัวตนก่อนเข้าใช้งานระบบ โดยมีวัตถุประสงค์เพื่อเป็นแนวทางในการเพิ่มความปลอดภัยให้ระบบงาน โดยเน้นไปที่การไม่เพิ่มภาระให้กับผู้ใช้งานมากเกินไป ในงานวิจัยนี้ได้จำลองหน้าจอการเข้าสู่ระบบและจำลอง API สำหรับรับส่งข้อมูลในเบื้องต้น และนำ Machine Learning ชนิด Binary Classification ชนิดต่าง ๆ มาใช้ในการสอนแบบจำลอง การตรวจสอบพฤติกรรมการเข้าสู่ระบบ เพื่อค้นหาวิธีที่มีความเหมาะสมกับชุดข้อมูลในการวิจัยนี้มากที่สุด จากผลลัพธ์ของการทดสอบแบบจำลอง พบว่าการทำ Classification วิธี AveragedPerceptron มี F1 Score สูงสุดอยู่ที่ 0.93 ซึ่งสามารถนำไปใช้งานได้ หากมีการนำไปใช้งานโดย สอนแบบจำลอง ด้วยข้อมูลที่ได้จากการใช้งานระบบจะทำให้แบบจำลอง มีความแม่นยำมากยิ่งขึ้น

(มีจำนวนทั้งสิ้น 64 หน้า)

คำสำคัญ : การวิเคราะห์พฤติกรรมการเข้าสู่ระบบ การยืนยันตัวตนหลายปัจจัย การเพิ่มความปลอดภัยเว็บไซต์

_____ อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก

Name : Mr. Nattawee Poolsombatpinyo
Independent Study Title : Two Factor Authentication with Filtering Abnormal Login Behavior
Major Field : Network and Information Security Management King Mongkut's University of Technology North Bangkok
Independent Study Advisor : Pongsarun Boonyopakorn
Academic Year : 2025

ABSTRACT

This research is a conceptual design and simulation of a identification system focus on system security and Focusing on not adding too much steps to users in this research create login screen design and API, initial components, and machine learning., different binary classifications may teach the login verification model. The most validated dataset verification methods in this research are based on the model testing results. The classification method for AveragedPerceptron has the highest value F1 score is 0.93 that sufficient for actual use. The implementation by learning the model with system usage data will make the model reliable.

(Total 64 Pages)

Keywords: Login Behavior Analysis, Multi-Factor Authentication, Website Security Enhancements

_____ Advisor

กิตติกรรมประกาศ

การค้นคว้าอิสระนี้ฉบับนี้สำเร็จลุล่วงได้ด้วยความช่วยเหลือจากผู้ช่วยศาสตราจารย์ ดร.พงศ์ศรัณย์ บุญโญปกรณ์ และอาจารย์ ดร.ธนวรรธน์ โยชะนัง ที่คอยเสนอแนะและให้คำปรึกษา รวมถึงติดตามความคืบหน้าอย่างต่อเนื่อง ซึ่งช่วยให้ผู้วิจัยสามารถทำการค้นคว้าได้อย่างต่อเนื่อง และขอขอบคุณอาจารย์ประจำสาขาวิชาการบริหารเครือข่ายดิจิทัลและความมั่นคงปลอดภัยสารสนเทศทุกท่านที่ได้มอบความรู้และให้ประสบการณ์แก่ผู้วิจัย ซึ่งมีความสำคัญยิ่งในการจัดทำการค้นคว้าอิสระในครั้งนี้

นอกจากนี้ ขอขอบคุณ เพื่อนในภาควิชาฯ และครอบครัว ที่ได้สนับสนุนผู้วิจัยมาโดยตลอด ซึ่งเป็นส่วนสำคัญยิ่งในการทำให้การวิจัยนี้สำเร็จเสร็จสิ้น

สุดท้ายนี้ ผู้วิจัยคาดหวังว่าการจัดทำการค้นคว้าอิสระในครั้งนี้จะเป็นประโยชน์ให้ที่สนใจสามารถนำไปต่อยอดเพื่อพัฒนาสิ่งที่ดีขึ้นต่อไป

ณัฐวีร์ พูลสมบัติภิญโญ

สารบัญ

หน้า

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันหน่วยงานทั้งภาครัฐ และเอกชนมีการใช้ระบบออนไลน์ในการให้บริการมากยิ่งขึ้น ส่งผลให้ มีข้อมูลจำนวนมากจัดเก็บอยู่ในระบบ ซึ่งข้อมูลเหล่านี้มีมูลค่ามากในการนำไปต่อยอดการพัฒนาบริการให้ตรงตามความต้องการของผู้ใช้งาน จุดนี้จึงตกเป็นเป้าหมายของผู้ไม่ประสงค์ดีที่ต้องการทำลายชื่อเสียงของหน่วยงาน ต้องการนำข้อมูลมาเป็นข้อต่อรองในการเรียกเงินจากหน่วยงาน ต้องการนำข้อมูลไปขายที่ดาร์กเว็บ หรือต้องการลบข้อมูลเพื่อให้ระบบการให้บริการหยุดชะงัก ซึ่งคำแนะนำเกี่ยวข้องกับการเพิ่มความปลอดภัยของรหัสผ่านในปัจจุบันยังไม่เพียงพอ เช่น แนะนำให้ใช้รหัสผ่านของแต่ละระบบให้แตกต่างกัน เปลี่ยนรหัสผ่านเป็นประจำทุก 3 เดือน การตั้งรหัสผ่านให้ยาวโดยมีทั้งตัวเลข ตัวอักษร และอักขระพิเศษ แต่วิธีการเหล่านี้ยังมีช่องโหว่สำคัญ คือ เมื่อระบบกำหนดให้เปลี่ยนรหัสใหม่เป็นประจำผู้ใช้งานจะตั้งรหัสที่สั้นและจดจำง่าย เมื่อระบบกำหนดให้ตั้งรหัสที่ยาวและมีตัวอักษรผสมกับตัวเลขผู้ใช้งานจะจำไม่ได้และต้องจดรหัสผ่านไว้ที่หน้าจอ จะเห็นว่าวิธีการเหล่านี้สามารถช่วยได้ในบางส่วนเท่านั้น อีกทั้งบนเว็บไซต์มีการเผยแพร่ไฟล์ wordlist รหัสผ่านที่มีการใช้งานบ่อยอยู่มากมาย หากหน่วยงานใช้การยืนยันตัวตนอื่นมาร่วมด้วยจะช่วยสร้างความปลอดภัยได้อย่างเห็นผลยิ่งขึ้น

จากสถิติของ World Economic Forum [11] พบว่าในปัจจุบันความเสียงจากการถูกโจมตีทางไซเบอร์สูงเป็นอันดับที่ 5 หรือคิดเป็น 39% ของความเสี่ยงทั้งหมด และคาดการว่าในอีก 2 ปีข้างหน้าความเสียงจากการถูกโจมตีทางไซเบอร์จะเพิ่มขึ้นมาอยู่ในลำดับที่ 4 และจากการสำรวจของ Rapid7 พบว่า 41% ของบริษัทที่ถูกโจมตีไม่มีการใช้งานการยืนยันตัวตนแบบหลายปัจจัย ตัวเลขยังคงที่นับจากต้นปี 2023 ที่คิดเป็น 39% ของบริษัทที่ถูกโจมตี

จากสถานการณ์ข้างต้น จึงขอเสนอระบบต้นแบบการใช้การยืนยันตัวตนแบบหลายปัจจัยร่วมกับการกรองพฤติกรรมการยืนยันตัวตนที่ผิดปกติ โดยใช้ข้อมูล log การเข้าสู่ระบบในการตรวจสอบพฤติกรรมที่ผิดปกติ เพื่อส่งการยืนยันตัวตน 2 ปัจจัย ไปยังผู้ใช้งานเฉพาะกรณีที่พบพฤติกรรมที่ผิดปกติเท่านั้น ซึ่งจะช่วยเพิ่มความปลอดภัยให้ระบบ โดยไม่เพิ่มภาระแก่ผู้ใช้งานระบบจนมากเกินไป

1.2 วัตถุประสงค์

1.2.1 เพื่อพัฒนาระบบการตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

1.2.2 เพื่อประเมินประสิทธิภาพของแบบจำลองสำหรับการตรวจจับการเข้าสู่ระบบที่ผิดปกติ

1.3 ขอบเขตของการวิจัย

1.3.1 การค้นคว้าอิสระนี้จะกำหนดขอบเขตการวิจัยโดยเน้นไปที่การใช้ Software ที่เป็น Open Source

1.3.2 การยืนยันตัวตนผู้ใช้งานของระบบ จะใช้การยืนยันตัวตนผ่านโปรโตคอล LDAP เท่านั้น

1.3.3 การยืนยันตัวตน 2 ขั้นตอน (Two factor authentication : 2FA) ในการวิจัยนี้จะใช้ One-time password (OTP) ส่งผ่าน e-Mail หรือ Software ที่ไม่มีค่าใช้จ่ายเท่านั้น

1.3.4 การสอนแบบจำลอง จะใช้ข้อมูลจาก Dataset ที่เผยแพร่บนอินเทอร์เน็ต หรือ ข้อมูลที่เกิดขึ้นจากระบบที่พัฒนาในการวิจัยนี้เท่านั้น

1.3.5 ระบบที่ใช้ทดสอบในการค้นคว้าอิสระนี้ จะใช้ระบบปฏิบัติการ Microsoft Window และ Microsoft Window Server เป็นหลัก

1.4 วิธีการวิจัย

1.4.1 ระบุประเด็นปัญหาที่พบ

1.4.2 กำหนดแนวทางและวิธีการที่ใช้แก้ปัญหา โดยอ้างอิงตามข้อจำกัดของหน่วยงาน

1.4.3 ออกแบบขั้นตอนการทำงาน (Workflow) ที่สามารถแก้ไขปัญหาที่กำหนด

1.4.4 เลือกเทคโนโลยี/เครื่องมือที่สอดคล้องกับแนวทางและวิธีการแก้ไขปัญหา

1.4.5 ออกแบบแผนผัง Network Diagram จุดเชื่อมต่อ บริการต่าง ๆ ของระบบ

1.4.6 สร้างฐานข้อมูลเพื่อจัดเก็บ Log การใช้งานระบบ

1.4.7 สร้าง Server จำลองด้วย Virtual Machine ติดตั้ง Windows Server และติดตั้งบริการ Active Directory

1.4.8 สร้างบัญชี e-Mail และบัญชีผู้ใช้งาน API บริการตรวจสอบ IP

1.4.9 พัฒนา Application ส่วนหน้าจอเข้าสู่ระบบ

1.4.10 พัฒนา API ตรวจสอบการเข้าสู่ระบบ

1.4.10.1 พัฒนาฟังก์ชันการเชื่อมต่อไปยัง Active Directory

1.4.10.2 พัฒนาส่วนจัดเก็บข้อมูลรายละเอียดการเข้าสู่ระบบของผู้ใช้งาน

1.4.10.3 พัฒนาส่วนเชื่อมต่อฐานข้อมูล อ่านและเขียนข้อมูลลงบนฐานข้อมูล

1.4.10.4 พัฒนาส่วนการวิเคราะห์พฤติกรรมการเข้าสู่ระบบ

1.4.10.5 พัฒนาฟังก์ชันการส่ง Email แจ้งเตือน OTP

1.4.11 เพิ่มฟังก์ชัน Application หน้าจอเข้าสู่ระบบ เพิ่มหน้าจอระบุและยืนยันตัวตนด้วย OTP

1.4.12 พัฒนา API ยืนยันตัวตนด้วย OTP

1.4.13 พัฒนา API สอนแบบจำลอง

1.4.14 หาชุดข้อมูล (Dataset) และ Clean ชุดข้อมูล

1.4.15 นำเข้าชุดข้อมูลเพื่อ สอนแบบจำลองผ่าน API

1.4.16 ทดสอบใช้งานระบบ

1.4.16.1 ทดสอบส่วนตรวจสอบ Credential ผ่าน Active Directory

1.4.16.2 ทดสอบการเรียก API บริการตรวจสอบ IP

1.4.16.3 ทดสอบการวิเคราะห์พฤติกรรมการเข้าสู่ระบบ

1.4.16.4 ทดสอบส่วนการส่ง Email แจ้ง OTP

1.4.16.5 ทดสอบการยืนยันตัวตนด้วย OTP

1.4.17 ปรับปรุงแก้ไขและเพิ่มประสิทธิภาพของระบบตามผลที่พบจากการทดสอบ

1.4.18 ทดสอบการเข้าสู่ระบบทั้งแบบปกติและที่ผิดปกติ นับจำนวน true positive, true negative, false positive, false negative พร้อมจัดเก็บผลการทดสอบ

1.4.19 ปรับปรุงแบบจำลอง ให้มีความแม่นยำมากขึ้น

1.4.20 สรุปผลการทดสอบระบบ

1.5 ประโยชน์ของการวิจัย

1.5.1 เป็นแนวทางในการเพิ่มความปลอดภัยให้ระบบงาน โดยไม่เป็นการเพิ่มภาระให้ผู้ใช้งานมากเกินไป

1.5.2 หน่วยงานสามารถนำแนวทางการการตรวจสอบพฤติกรรมการเข้าสู่ระบบไปประยุกต์ใช้กับระบบของหน่วยงานได้ตามความเหมาะสมโดยไม่เสียค่าใช้จ่าย

บทที่ 2
แนวคิดทฤษฎีและงานวิจัยที่เกี่ยวข้อง

งานค้นคว้าอิสระ “การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ” นี้ผู้วิจัยได้ศึกษาแนวคิดทฤษฎี หลักการ และงานวิจัยต่าง ๆ ที่เกี่ยวข้อง ดังนี้

2.1 คลังการเรียนด้วยเครื่อง

คลังการเรียนด้วยเครื่อง (Machine Learning library) เป็น library สำหรับนักพัฒนาระบบที่รวมรวมฟังก์ชันในการอำนวยความสะดวกต่าง ๆ ที่ใช้สำหรับการทำ Machine Learning โดยมีตั้งแต่ฟังก์ชันในการกำหนดตัวแปรนำเข้าข้อมูล ฟังก์ชันในการแปลงข้อมูลให้อยู่ในรูปแบบต่าง ๆ ฟังก์ชันในการแบ่งชุดข้อมูลสำหรับ ทดสอบและสอนฟังก์ชันสำหรับการ สอนแบบจำลอง และ ฟังก์ชันในการทดสอบสรุปประเมินผลความแม่นยำ เป็นต้น ซึ่ง library ถูกพัฒนาจากหลายแหล่งที่มาและมีวัตถุประสงค์ในการพัฒนาที่แตกต่างกัน และเนื่องจาก library มีหลายประเภทให้เลือกจึงควรเลือกใช้ตามความเหมาะสมและรองรับภาษาที่ใช้พัฒนาระบบ โดยมี library ที่เกี่ยวข้อง ได้แก่

2.1.1 ML.Net เป็น library สำหรับทำ Machine Learning เปิดตัวครั้งแรกในปี พ.ศ. 2561 พัฒนาขึ้นสำหรับใช้งานร่วมกับ .Net Framework ซึ่งฟังก์ชันการทำงานด้าน Machine Learning ที่ครอบคลุมรองรับการทำ Classification ได้หลากหลายวิธี ทั้งการจำแนกรูปภาพ จำแนกข้อความ และการทำ Regression ด้วยวิธีต่าง ๆ อีกทั้งยังเป็น library ที่มีการใช้งานอย่างแพร่หลายจึงมีสามารถหาข้อมูลได้ง่ายทั้งจากเอกสารที่ผู้พัฒนาเผยแพร่และจากชุมชนนักพัฒนาระบบ [7]

2.1.2 PyTorch เป็น library ที่ใช้กับภาษา Python เป็นหลักและมีการพัฒนาส่วนเสริมให้สามารถใช้งานร่วมกับภาษา C++ ใช้สำหรับการทำ Deep Learning โดยเปิดตัวครั้งแรกในปี พ.ศ. 2558 ถูกพัฒนาโดย Pytorch Foundation ซึ่งประกอบด้วยบุคลากรทั้งนักพัฒนาและนักวิจัยที่ศึกษาเกี่ยวกับงานด้าน Deep Learning และเปิดให้โอกาสบุคคลทั่วไปสามารถมีส่วนร่วมในการพัฒนา เรียนรู้ และถามปัญหาได้ [9]

2.2 การจำแนกแบบ 2 กลุ่ม

การจำแนกแบบ 2 กลุ่ม (Binary Classification) เป็นรูปแบบหนึ่งของการเรียนรู้แบบต้องมีผู้สอน (Supervied Learning) จำเป็นต้องมีข้อมูลเพื่อนำมาสอนแบบจำลอง โดยยิ่งข้อมูลมีจำนวนมากและการกระจายอย่างเหมาะสมมากเท่าไร แบบจำลองจะยิ่งทำนายแม่นขึ้น โดย Binary Classification ใช้ในการจำแนกหมวดหมู่ โดยจำแนกเพียง 2 สิ่งเช่น จริงหรือเท็จ 1 หรือ 0 Confusion Matrix การวัดประสิทธิภาพของแบบจำลอง การทำ Machine Learning ว่าแบบจำลองสามารถทำนายได้แม่นยำเพียงใด โดยมีขั้นตอนได้แก่

2.2.1 จำแนกผลการทำนายออกเป็น 4 ข้อ ดังนี้

2.2.1.1 True Positive (TP) การที่แบบจำลอง ทำนายว่า ใช่ และผลการทำนาย ถูก

2.2.1.2 True Negative (TN) การที่แบบจำลอง ทำนายว่า ไม่ใช่ และผลการทำนาย ถูก

2.2.1.3 False Positive (FP) การที่แบบจำลอง ทำนายว่า ใช่ แต่ผลการทำนาย ผิด

2.2.1.4 False Negative (FN) การที่แบบจำลอง ทำนายว่า ไม่ใช่ แต่ผลการทำนาย ผิด

2.2.2 นำผลการทำนายมาคำนวณประสิทธิภาพของแบบจำลอง

2.2.2.1 Accuracy อัตราส่วนระหว่างจำนวนที่แบบจำลอง ทำนายถูกเทียบกับจำนวนข้อมูลทั้งหมด

2.2.2.2 Precision อัตราส่วนที่ระหว่างจำนวนที่แบบจำลอง ทำนายว่า ใช่ แล้วถูก เทียบกับจำนวนทั้งหมดที่แบบจำลอง ทำนายว่าใช่

2.2.2.3 Recall อัตราส่วน อัตราส่วนที่ระหว่างจำนวนที่แบบจำลอง ทำนายว่า ใช่ แล้วถูกเทียบกับจำนวนผลที่เกิดขึ้นจริงเป็น ใช่

2.2.2.4 F1 Score ใช้เป็นค่าบ่งบอกถึงประสิทธิภาพของแบบจำลอง ใช้เปรียบเทียบว่าแบบจำลอง มีความเหมาะสมมากกว่ากัน โดยคะแนนจะอยู่ในช่วง 0 ถึง 1 ยิ่งคะแนนสูงหมายถึงแบบจำลอง มีประสิทธิภาพสูง

2.3 การวิเคราะห์สวอต

การวิเคราะห์สวอต (SWOT Analysis) เป็นเครื่องมือสำหรับวิเคราะห์ภาพรวมขององค์กรซึ่งจะช่วยให้องค์กรทราบประเด็นปัญหา และวางแผนในการปรับปรุงแก้ไขกระบวนการเพิ่มประสิทธิภาพองค์กรต่อไป โดย SWOT ประกอบด้วย การวิเคราะห์ 4 ด้าน ดังนี้

2.3.1 จุดแข็ง (Strengths) ปัจจัยภายในที่ทำให้หน่วยงานได้เปรียบมากกว่าหน่วยอื่น เป็นสิ่งที่ควรพัฒนาต่อยอดให้ดียิ่งขึ้น

2.3.2 จุดอ่อน (Weaknesses) ปัจจัยภายในที่ทำให้หน่วยงานเสียเปรียบ เป็นข้อจำกัดของหน่วยงาน เป็นสิ่งที่หน่วยงานควรหาทางปรับปรุงแก้ไข

2.3.3 โอกาส (Opportunities) ปัจจัยภายนอกที่หน่วยงานสามารถดำเนินการเพื่อใช้ประโยชน์เพื่อความได้เปรียบหน่วยงานอื่นได้

2.3.4 อุปสรรค (Threats) ปัจจัยภายนอกที่เป็นความเสี่ยงของหน่วยงานอาจส่งผลให้หน่วยงานเสียเปรียบได้ ควรหาแนวทางป้องกันหรือลดผลกระทบ

2.4 กระบวนการคุณภาพ

กระบวนการคุณภาพ (PDCA) เป็นเครื่องมือที่ใช้ในการปรับปรุงกระบวนการอย่างต่อเนื่อง โดยแบ่งเป็น 4 ขั้นตอน ดังนี้

2.4.1 ขั้นตอนการวางแผน (Plan) การวางแผนการดำเนินการอย่างเป็นขั้นตอนตั้งแต่เริ่มต้นจนบรรลุตามวัตถุประสงค์

2.4.2 ขั้นตอนการเริ่มดำเนินการ (Do) การดำเนินการตามที่ขั้นตอนที่ได้วางแผนไว้

2.4.3 ขั้นตอนการตรวจสอบผลการดำเนินการ (Check) การตรวจสอบความถูกต้องของการดำเนินงานและรวบรวมจุดสังเกตที่ยังสามารถพัฒนาต่อยอดได้

2.4.4 ขั้นตอนการปรับปรุงแก้ไข (Act) การปรับปรุงการดำเนินการแก้ไขสิ่งที่พบจากหลังจากการตรวจสอบ เพื่อพัฒนากระบวนการให้ดียิ่งขึ้นและแก้ไขสิ่งที่ยังเป็นปัญหา

2.5 รายงานความเสี่ยงทางด้านความปลอดภัย 10 อันดับ

รายงานความเสี่ยงทางด้านความปลอดภัย 10 อันดับ (Owasp Top 10) เอกสารที่รวบรวมแนวโน้มของภัยคุกคามที่เกิดขึ้นในแต่ละปี โดยเรียงลำดับ 10 รายการที่เป็นที่นิยม มีการเปรียบเทียบรายการของปีปัจจุบันกับรายงานฉบับปีก่อนหน้า และมีการคาดเดาแนวโน้มของภัยคุกคามในอนาคต โดยสามารถตรวจสอบช่องโหว่ของระบบตามรายการ OWASP TOP TEN ได้ด้วยโปรแกรม ZAP ซึ่งมีความสะดวกในการใช้งานพัฒนามาเพื่อใช้ในการตรวจสอบระบบและระบุจำนวนช่องโหว่ที่พบแจกแจงรายละเอียดของช่องโหว่แต่ละรายการและแนวทางการแก้ไข สามารถออกรายงานสรุปผลการตรวจสอบได้ในรายงานจะระบุความเสี่ยงที่พบของระบบแบ่งเป็น 4 ระดับความเสี่ยง ได้แก่ High Medium Low และ Information [8]

2.6 การยืนยันตัวตนหลายขั้นตอน

การยืนยันตัวตนหลายขั้นตอน (Multifactor Authentication : MFA) คือการยืนยันตัวตนโดยใช้มากกว่า 1 ปัจจัยเพื่อยืนยันว่าเป็นบุคคลนั้นจริง โดยทั่วไปปัจจัยที่มีความนิยมใช้ ได้แก่ สิ่งที่รู้ (Something you know) สิ่งที่มี (Something you have) สิ่งที่เป็น (Something you are) ในการวิจัยนี้จะมีการใช้ Password สำหรับเข้าสู่ระบบซึ่งถือเป็นสิ่งที่รู้ (Something you know) และมีการใช้งาน OTP ส่งไปที่ Email ซึ่งถือเป็นสิ่งที่มี (Something you have) ในการยืนยันตัวตน

2.7 บริการตรวจสอบไอพีอันตราย

บริการตรวจสอบไอพีอันตราย เป็นบริการที่ผู้ใช้งานสามารถส่งเลข IP Address ผ่าน API เพื่อนำไปตรวจสอบข้อมูลว่า IP Address นั้นมีการใช้ในการโจมตีมาก่อนหรือไม่ โดยผู้ให้บริการจะมี API สำหรับตรวจสอบและ API สำหรับรับแจ้งรายการ IP Address ที่เป็นอันตรายซึ่งจถูกบันทึกบนฐานข้อมูลของผู้ให้บริการ เช่น

2.7.1 Virustotal เป็นบริการตรวจสอบ IP Address ที่มีการรวบรวมการรับแจ้งข้อมูลจาก Security Vender หลายแหล่งนำมารวบรวมลงในฐานข้อมูล เมื่อมีการเรียกใช้บริการผ่าน API จะได้รับข้อมูลการวิเคราะห์จาก Security Vender ซึ่งแบ่งเป็น 4 สถานะได้แก่ malicious suspicious undetected harmless จากนั้นนำมานับจำนวนและสรุปเป็นคะแนนความอันตราย [10]

2.7.2 Abuseipdb เป็นบริการตรวจ IP Address ที่เปิดให้ผู้ใช้งานสามารถตรวจสอบ IP Address IP ว่ารายการที่ค้นหาเคยมีประวัติการถูกรายงานเป็น IP Address นับเป็นจำนวนครั้งที่ถูกรายงาน และสามารถแจ้ง IP Address ที่เป็นอันตรายเข้าสู่ระบบฐานข้อมูลผ่านทาง API ได้ [4]

2.7.3 Ismalicious เป็นบริการตรวจ IP Address ที่อันตราย โดยผู้ใช้งานสามารถนำ IP Address ไปตรวจสอบผ่าน API โดยจะได้ผลตอบกลับแบ่งประเภทเป็น malicious suspicious harmless undetected นับจำนวนแล้วแสดงเป็น คะแนน Security Score และมี API สำหรับให้ผู้ใช้งานแจ้ง IP Address ที่เป็นอันตรายได้ [5]

2.8 งานวิจัยที่เกี่ยวข้อง

วิจัยเรื่อง “Multifactor Authentication Using Zero Trust” เป็นการนำ Zero Trust Framework มาใช้ โดยมีการเลือกใช้การยืนยันตัวตนแบบลายขั้นตอน ร่วมกับ Active Directory Certificate VPN และ Firewall เพื่อให้เกิดช่องโหว่การโจมตีให้น้อยที่สุด โดยคำนึงถึงความสะดวกของผู้ใช้งานมีการทดสอบเปรียบเทียบเวลาที่ใช้ในการยืนยันตัวตน การเข้าสู่ระบบแบบปกติ และการเข้าสู่ระบบแบบใช้งาน VPN เปรียบเทียบระหว่างการเข้าสู่ระบบแบบเดิม (Single-Factor Authentication) และแบบใช้การยืนยันตัวตนแบบลายขั้นตอน (Multi-Factor Authentication) จากนั้นมีการสรุปผลความพึงพอใจของผู้ใช้งาน จากการสำรวจความพึงพอใจการใช้งานระบบยืนยันตัวตนร่วมกันการใช้ Push Request ส่งผลดีต่อความพึงพอใจของผู้ใช้งาน เนื่องจากช่วยลดความยุ่งยากลงเมื่อเทียบกับการยืนยันตัวตนด้วยวิธีอื่น และช่วยเพิ่มความปลอดภัยโดยไม่กระทบต่อประสิทธิภาพโดยรวม [6]

วิจัยเรื่อง “ปัจจัยที่อิทธิพลต่อความพึงพอใจในการใช้งานการยืนยันตัวตน แบบหลายปัจจัยในโมไบล์แบงก์กิ้งแอปพลิเคชัน” การวิจัยนี้เป็นการวิเคราะห์ประสบการใช้งาน Mobile Banking โดยพิจารณาใน 6 ปัจจัย ได้แก่ 1) ประสบการณ์ของผู้ใช้งาน, 2) การรับรู้ประโยชน์, 3) การรับรู้ความง่ายในการใช้งาน, 4) ความไว้วางใจ, 5) การรับรู้ความปลอดภัย และ 6) ความพึงพอใจ โดยจัดทำแบบสอบถามผ่านช่องทางออนไลน์เป็นระยะเวลา 1 เดือน จากนั้นวิเคราะห์ความสัมพันธ์ระหว่างทางตรงและทางอ้อมของแต่ละปัจจัย โดยผลการวิจัยพบว่าปัจจัยประสบการณ์ของผู้ใช้งานส่งผลทางบวกต่อปัจจัยการรับรู้ประโยชน์ ปัจจัยการรับรู้ความง่ายในการใช้งานส่งผลทางบวกต่อปัจจัยการรับรู้ประโยชน์ และการรับรูปประโยชน์ส่งผลต่อความไว้วางใจ และปัจจัยการรับรู้ประโยชน์ส่งผลทางบวกต่อปัจจัยความไว้วางใจในการใช้งานระบบ [2]

วิจัยเรื่อง “การวิเคราะห์แนวทางการพัฒนาการตรวจสอบและแจ้งเตือน การรักษาความมั่นคงปลอดภัยของระบบปฏิบัติการด้วยการประยุกต์ใช้ โมเดลออโตมาตา” การวิจัยนี้มีการสร้างแบบจำลองสำหรับตรวจจับการโจมตีแบบ Zero Day และการโจมตีด้วยการทำซ้ำ ๆ โดยใช้โครงข่ายประสาทเทียมในการจดจำรูปแบบการโจมตี ประกอบกับการใช้ข้อมูลที่เกี่ยวกับความปลอดภัย เช่น วันที่ เวลาที่เกิดเหตุการณ์ ประเภทกิจกรรม ผู้ใช้งาน หรือแอปพลิเคชันที่ใช้งาน ซึ่งเป็นข้อมูลที่ได้จาก log และรายงานของระบบ IDS เพื่อนำมาสอนแบบจำลองให้สามารถตรวจจับและรายงานเหตุการณ์ที่อาจเป็นสัญญาณของการโจมตีได้แม่นยำมากยิ่งขึ้น จากการทดลองพบว่าการเลือกเครื่องมือและข้อมูลที่ใช้ทดสอบให้มีความเหมาะสมจะช่วยให้แบบจำลองมีประสิทธิภาพในการตรวจจับมากยิ่งขึ้น โดยในการทดสอบแบบจำลองสามารถตรวจจับและปรับปรุงข้อมูลได้อย่างต่อเนื่อง ซึ่งช่วยให้ระบบสามารถส่งข้อมูลและแจ้งเตือนเหตุการณ์ที่พบในระบบได้อย่างมีประสิทธิภาพ [1]

วิจัยเรื่อง “การยืนยันตัวตนสำหรับเว็บแอปพลิเคชันแบบพหุปัจจัย” การวิจัยนี้เป็นการพัฒนาระบบที่มีการใช้งานการยืนยันตัวตนแบบ 2 ขั้นตอนโดยการใช้การ Scan QR Code ผ่านโทรศัพท์มือถือซึ่งผูกกับ Token ID ที่ไม่ซ้ำกัน มีการเปรียบเทียบเวลาที่ใช้ในการทำงาน โดยทดสอบการใช้งานพร้อมกันตั้งแต่ 30 ถึง 240 บัญชีในเวลาเดียวกัน การทดสอบแบ่ง 3 ขั้นตอน ได้แก่ 1) User Login หมายถึงเมื่อผู้ใช้งานเข้าใช้งานครั้งแรกสำเร็จระบบจะสร้าง Token ID กลับมาที่เครื่องของผู้ใช้งาน 2) User Authen หมายถึงเมื่อผู้ใช้งานที่เคยเข้าใช้งานแล้วทำการเข้าสู่ระบบเพื่อรับ Token ID ที่บันทึกไว้ในฐานข้อมูลและ 3) Web Authen หมายถึงการเข้าสู่ระบบตั้งแต่การเข้าสู่ระบบด้วยบัญชีผู้ใช้และรหัสผ่าน จากนั้นระบบขึ้นหน้าจอสแกน QR Code ไปจนถึงผู้ใช้งานสแกน QR Code เข้าสู้ระบบสำเร็จ จากผลการทดสอบหาเวลาการตอบสนองของระบบของทั้ง 3 ส่วนพบว่าขั้นตอน User Login ใช้เวลาไม่เกิน 1 วินาที ขั้นตอน User Authen ใช้เวลา 3 วินาที และขั้นตอน Web Login ใช้เวลา 5 วินาที ที่การใช้งาน 240 บัญชีในขณะเดียวกัน [3]

บทที่ 3
วิธีดำเนินการวิจัย

วิจัยเรื่อง “การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ” เพื่อเสนอแนะแนวทางในการป้องกันระบบงานที่สำคัญของหน่วยงานโดยไม่เพิ่มภาระให้ผู้ใช้งาน โดยการเปรียบเทียบ 2 เครื่องมือ ในการทำ Machine Learning ให้ได้เครื่องมือที่เหมาะสมแล้วจึง และทำเปรียบเทียบแบบจำลอง เพื่อเปรียบเทียบหาวิธีที่มีความเหมาะสมกับรูปแบบการใช้งานมากที่สุด ซึ่งจากที่กล่าวมามีวิธีดำเนินการวิจัยดังนี้

3.1 การวางแผน (PLAN)

3.1.1 ศึกษา วิเคราะห์ ประเด็นปัญหาที่พบในระบบปัจจุบัน ข้อจำกัดของหน่วยงาน และกำหนดความต้องการของระบบ เพื่อพิจารณาคัดเลือกปัญหาที่ระบบต้องแก้ไข โดยใช้วิธีการวิเคราะห์ SWOT

3.1.1.1 จุดแข็ง (Strengths) ได้แก่ เป็นหน่วยงานขนาดเล็กยังมีระบบไม่มากซึ่งสามารถปรับปรุงได้ง่าย และเนื่องจากเป็นหน่วยงานนโยบายจึงยังไม่มีระบบที่ให้บริการผู้ใช้งานภายนอก

3.1.1.2 จุดอ่อน (Weaknesses) ได้แก่ ระบบงานบางระบบไม่มีผู้บำรุงรักษาจึงมีความเสี่ยงต่อการถูกโจมตีและการจ้างบำรุงรักษาเป็นไปได้ยากเนื่องจาก ผู้พัฒนารายอื่นมีการใช้งานเครื่องมือที่ไม่เหมือนกัน และด้านงบประมาณที่มีจำกัดโดยหน่วยงานไม่มีรายได้จึงต้องดำเนินการของบประมาณประจำปีในการบริหารจัดการหน่วยงาน ซึ่งต้องดำเนินการของบประมาณเป็นรายปี ส่งผลให้ในบางปีงบประมาณอาจไม่ได้รับจัดสรรงบประมาณในส่วนของการต่ออายุลิขสิทธิ์ซอฟต์แวร์ซึ่งอาจเกิดผลกระทบต่อระบบที่มีการใช้ลิขสิทธิ์ซอฟต์แวร์แบบรายปีให้ไม่สามารถทำงานได้ครบทุกฟังก์ชัน

3.1.1.3 โอกาส (Opportunities) มีการอบรมผู้ใช้งานจากหน่วยงานภายนอกอย่างสม่ำเสมอ มีการการสนับสนุนประบบป้องกันจากหน่วยงานภายนอก มีเครือข่ายข้อมูลภายคุกคามจากหน่วยงานอื่นที่อยู่ใน Sector เดียวกัน

3.1.1.4 ภัยคุกคาม (Threats) การโจมตีจากภายนอกที่พัฒนาอย่างต่อเนื่อง อินเทอร์เน็ตมี bandwidth ที่จำกัด และต้องเชื่อมต่อผ่านเครือข่ายของหน่วยงานต้นสังกัดก่อนจึงสามารถเชื่อมต่อไปสู่ภายนอกได้ การซื้ออุปกรณ์หรือระบบต้องเป็นไปตามขั้นตอนทำให้ล่าช้า

3.1.1.5 ความต้องการของระบบ

1) จึงอยากประยุกต์ใช้ยืนยันตัวตนแบบหลายปัจจัยในการเพิ่มความปลอดภัยให้ระบบในเบื้องต้น

2) ผู้ใช้งานไม่ต้องการวิธีการที่มีความยุ่งยากหลายขั้นตอน

3.1.2 หาวิธีการ และเครื่องมือที่สามารถแก้ไขปัญหาที่กำหนด และเหมาะกับบริบทของหน่วยงาน

3.1.3 ออกแบบขั้นตอนการทำงาน (Workflow) ของระบบ

3.1.3.1 ขั้นตอนการทำงาน (Workflow) ของระบบตรวจจับพฤติกรรม

1) ผู้ใช้งานเข้าสู่เว็บไซต์สำหรับเข้าสู่ระบบ

2) ผู้ใช้งานระบุบัญชีผู้ใช้และรหัสผ่าน

3) ระบบส่งค่าบัญชีผู้ใช้และรหัสผ่านไปตรวจสอบที่ Active Directory

3.1) หากไม่ถูกต้อง ย้อนกลับไปทำข้อ 2)

3.2) หากถูกต้องดำเนินการตามข้อถัดไป

4) ระบบเรียก API ตรวจ IP Address ที่ผิดอันตราย

4.1) หากเป็น IP อันตราย ย้อนกลับไปทำข้อ 2)

4.2) หากเป็น IP ปกติดำเนินการตามข้อถัดไป

5) ตรวจสอบว่าบัญชีผู้ใช้อยู่ระหว่างการรอยืนยันตัวตนผ่าน OTP หรือไม่

5.1) หากใช้ ไปทำข้อ 14)

5.2) หากไม่ใช่ดำเนินการตามข้อถัดไป

6) นำข้อมูลที่ได้เพิ่มเข้าสู่แบบจำลอง

7) วิเคราะห์พฤติกรรมการเข้าสู่ระบบ

7.1) หากปกติระบบอนุญาตให้เข้าร่วมได้

7.2) หาก ปกติดำเนินการตามข้อถัดไป

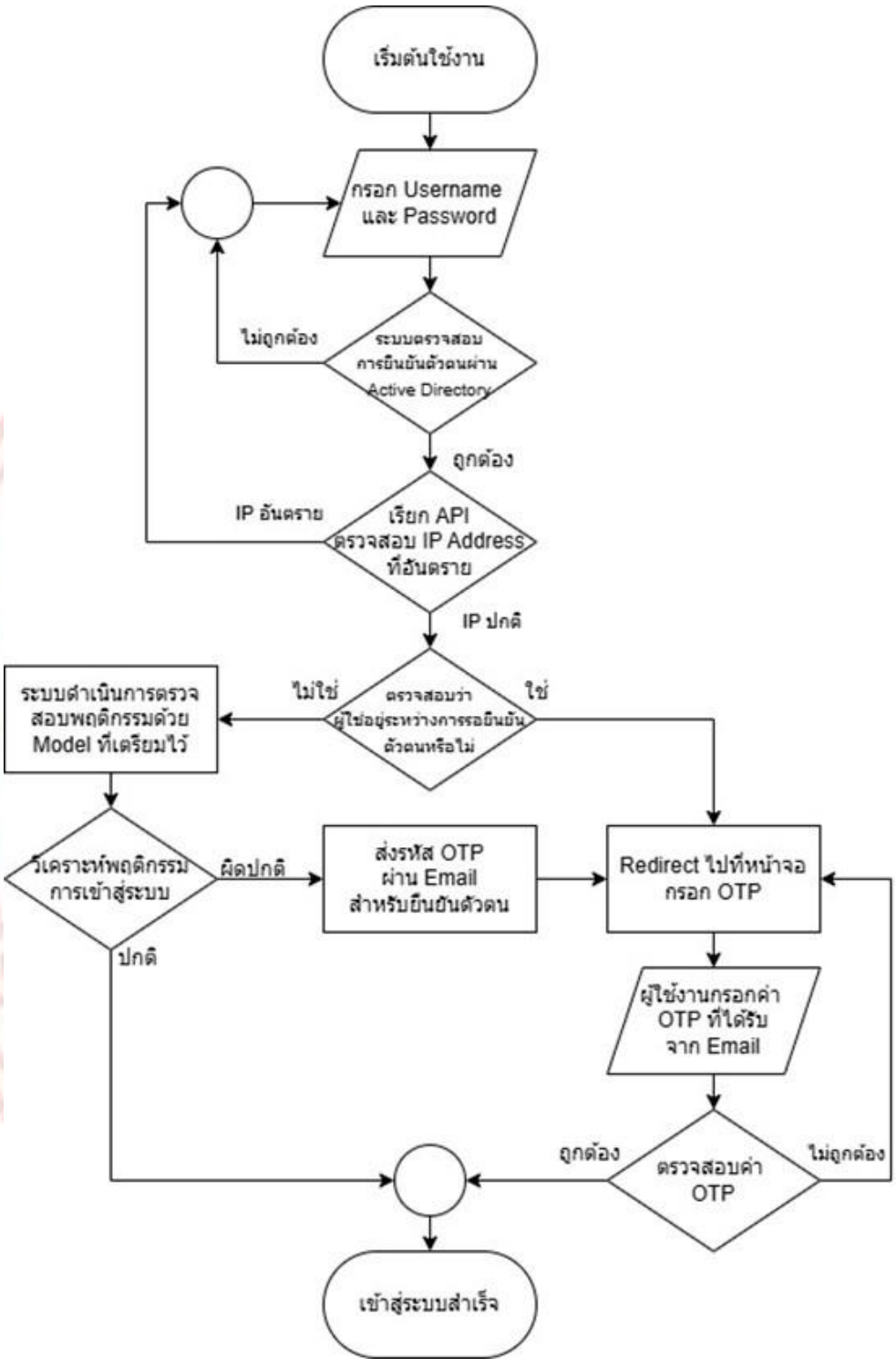
8) ระบบส่งไปที่หน้าจอ OTP

9) ระบบให้ผู้ใช้กรอก

9.1) หากส่งไปยังลำโรงไป

10) ส่งค่า IP ไปบริการตรวจสอบ IP ผ่านช่องทาง API

โดยสามารถสรุปขั้นตอนการทำงานได้ดังภาพที่ 3-1 จากภาพแสดงถึง ขั้นตอนการทำงานของระบบตรวจจับพฤติกรรมตั้งแต่เริ่มต้นกรอกบัญชีผู้ใช้งานไปจนถึงยืนยันตัวตนสำเร็จ

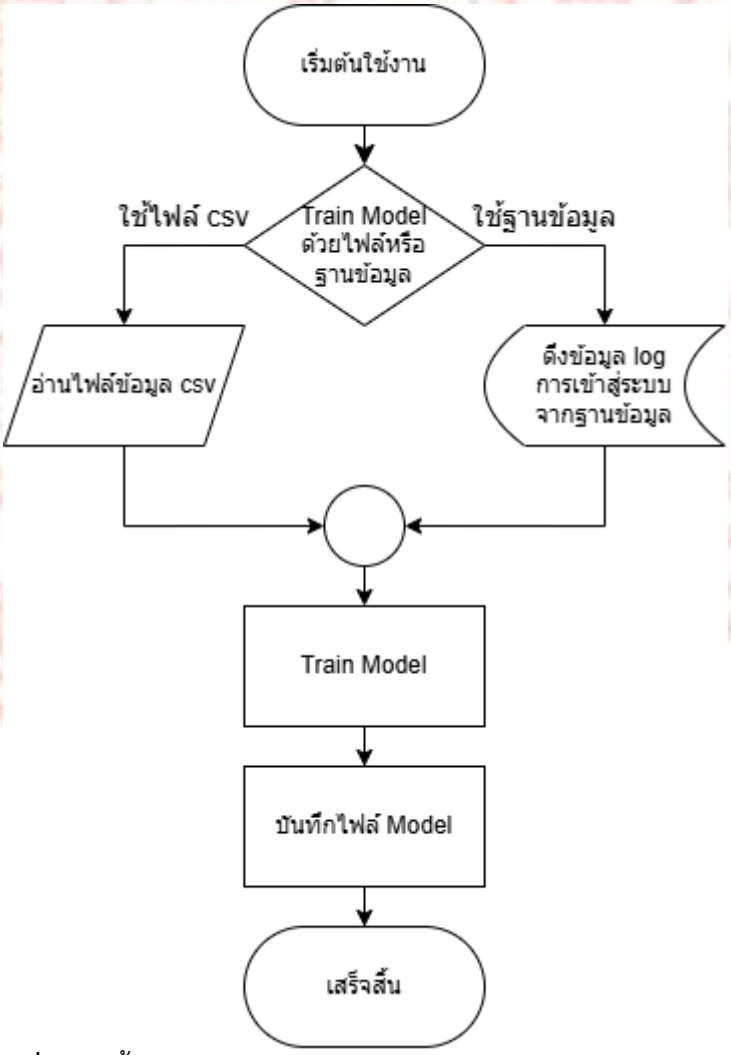


ภาพที่ 3-1 ขั้นตอนการทำงาน (Workflow) ของระบบตรวจจับพฤติกรรม

3.1.3.2 ขั้นตอนการทำงาน (Workflow) ของ API สอนแบบจำลอง

1) เรียก API สอนแบบจำลอง
2) ระบบอ่านไฟล์ตั้งค่า ดังนี้
   2.1) ระบบดำเนินการอ่านข้อมูลขากไฟล์ .csv
   2.2) ระบบอ่านไฟล์ข้อมูล log จากฐานข้อมูล
3) แบบจำลองเริ่มต้นการเรียนรู้จากชุดข้อมูลที่เลือก
4) ระบบบันทึกแบบจำลองจากนั้นวางไฟล์แบบจำลองในพื้นทีที่กำหนด

โดยสามารถสรุปขั้นตอนการทำงานได้ดังภาพที่ 3-2 จากภาพแสดงถึง ขั้นตอนการทำงานในส่วนของการสอนแบบจำลองตั้งแต่เริ่มต้นกำหนดแหล่งที่มาของข้อมูลไปจนถึงการบันทึกไฟล์แบบจำลองลงในระบบ



ภาพที่ 3-2 ขั้นตอนการทำงาน (Workflow) ของ API สอนแบบจำลอง

3.1.4 พิจารณาเลือกเทคโนโลยีที่เหมาะสมตามประเด็นปัญหาที่อยากแก้ไข และข้อจำกัดของหน่วยงาน

3.1.4.1 เนื่องจากข้อจำกัดของหน่วยงานทางผู้วิจัยจึงกำหนดหลักการคัดเลือกเครื่องมือดังนี้

1) ต้องเป็นเครื่องมือ Open source ที่มี Package ใช่งานแบบไม่เสียค่าใช้จ่าย เนื่องจากหน่วยงานมีข้อจำกัดในเรื่องงบประมาณ

2) มีความน่าเชื่อถือ เนื่องจากเครื่องมือต่าง ๆ มีการพัฒนาจากหลากหลายแหล่งที่มา ซึ่งอาจมีผู้พัฒนาเครื่องมืออย่างไม่ได้มาตรฐาน อาจส่งผลให้เกิดเป็นช่องโหว่ต่าง ๆ ของระบบ

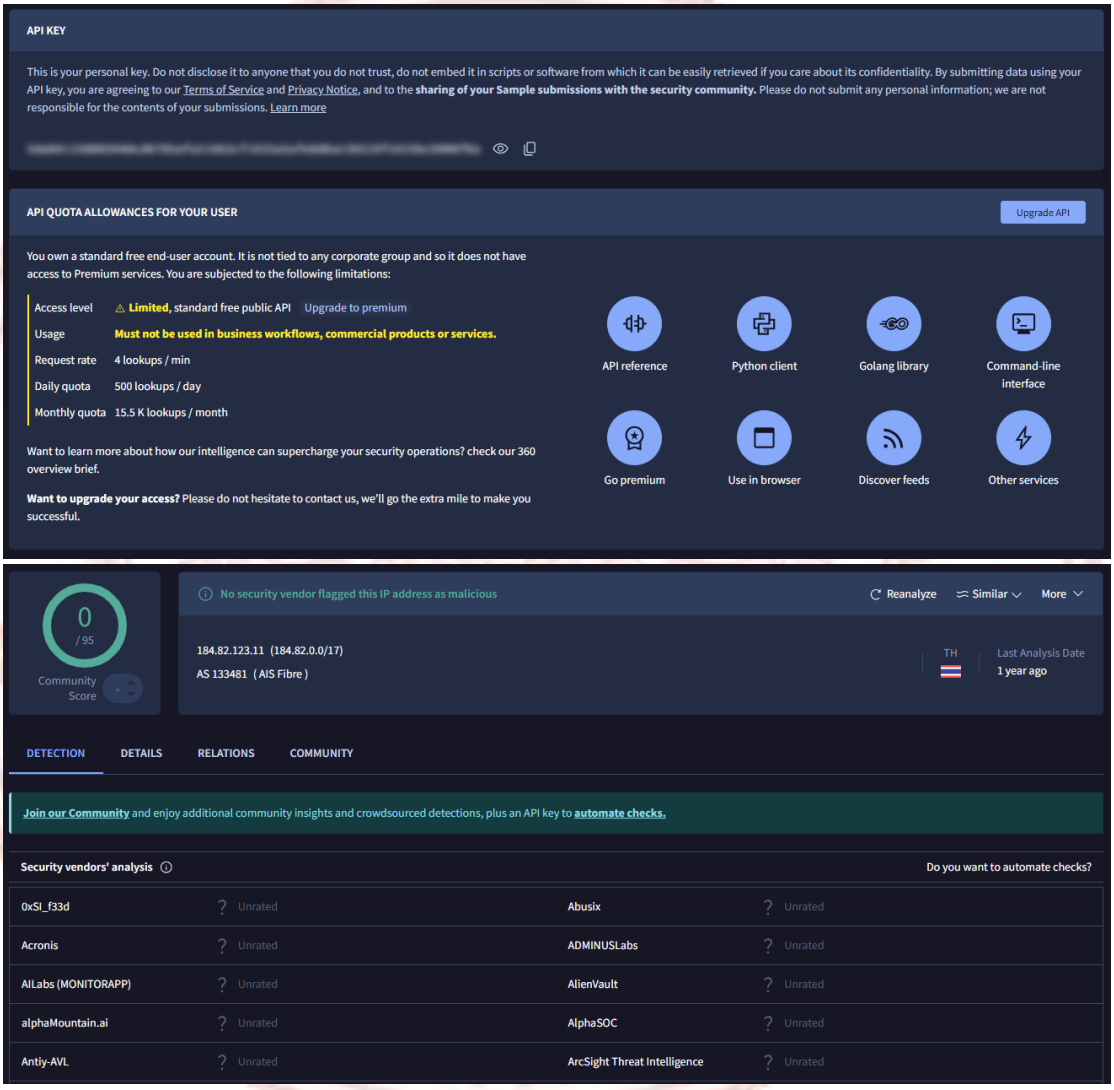
3) เป็นเครื่องมือที่ยังมีการพัฒนาอย่างต่อเนื่องจะมีการแก้ไขช่องโหว่ของระบบอย่างต่อเนื่อง

4) มีแหล่งค้นหาข้อมูลทั้งจากเอกสารของผู้พัฒนาหรือจากสังคมออนไลน์ เพื่อทราบข้อมูลการใช้งานเครื่องมือและรายละเอียดต่าง ๆ อีกทั้งเมื่อพบปัญหาระหว่างการพัฒนาจะสามารถหาแนวทางการแก้ไขจากจากแหล่งข้อมูลดังกล่าว

3.1.4.2 ทางผู้วิจัยได้เลือกพัฒนาระบบด้วยภาษา C#.Net เนื่องจากเป็นภาษาที่มีการใช้งานอย่างกว้างขวาง มีการใช้มาเป็นระยะเวลานาน และมีการพัฒนามาอย่างต่อเนื่อง และมี library ให้ใช้งานอย่างหลากหลาย และเป็นภาษาที่ทางผู้วิจัยมีความชำนาญในการพัฒนามากกว่าภาษาอื่น

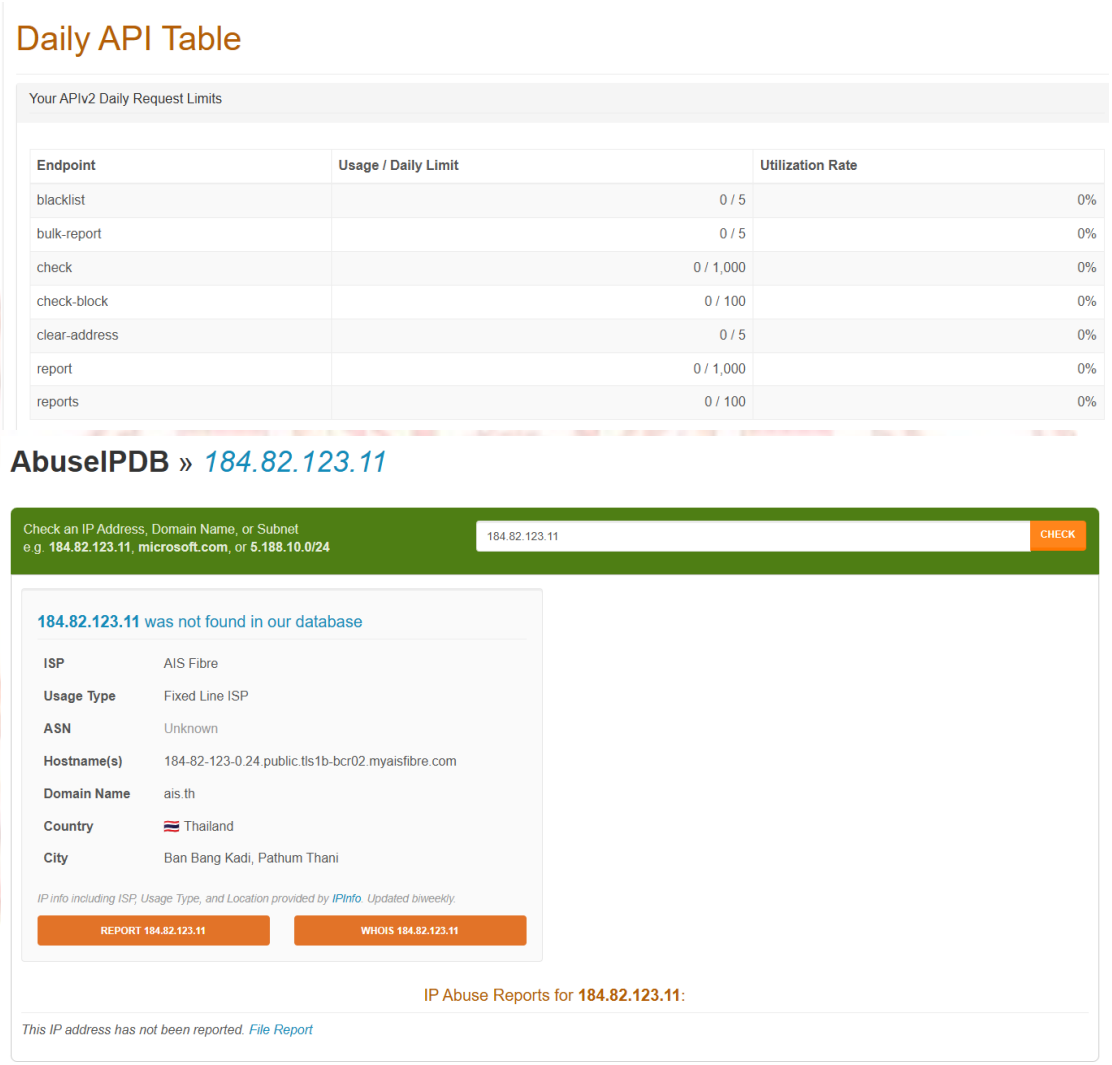
3.1.4.3 การเลือก API ตรวจสอบ IP อันตราย ผู้วิจัยเปรียบเทียบบริการจากผู้ให้บริการจำนวน 3 บริการ โดยประกอบด้วย ผู้ให้บริการ 2 รายที่เป็นที่รู้จัก และผู้ให้บริการที่พึ่งเปิดให้บริการ 1 ราย เพื่อเปรียบเทียบโครงสร้างข้อมูลที่ได้รับ และปริมาณที่สามารถส่งได้ต่อเดือน ดังนี้

1) Virustotal มีการเปิดให้ใช้งานแบบไม่เสียค่าใช้จ่ายโดยมีข้อจำกัด ดังนี้ สามารถค้นหาได้ไม่เกิน 15,500 ครั้ง/เดือน โดยต้องไม่เกิน 500 ครั้ง/วัน และความถี่ไม่เกิน 4 ครั้ง/นาที ดังภาพที่ 3-3 จากภาพแสดงถึงปริมาณการใช้งานในแพ็คเกจฟรี และ ตัวอย่างการเรียกใช้ API Virustotal



ภาพที่ 3-3 บริการ API สำหรับตรวจสอบ IP อันตรายของ Virustotal

2) AbuseIPDB มีการเปิดให้ใช้งานแบบไม่เสียค่าใช้จ่ายโดยมีข้อจำกัด ดังนี้ สามารถแจ้งรายงาน IP อันตรายได้ 5 ครั้ง/วัน สามารถค้นหาได้ 1,000 ครั้ง/วัน ดังภาพที่ 3-4 จากภาพแสดงถึงปริมาณการใช้งานในแพ็คเกจฟรี และ ตัวอย่างการเรียกใช้ API AbuseIPDB



Daily API Table

Your APIv2 Daily Request Limits

Endpoint	Usage / Daily Limit	Utilization Rate
blacklist	0 / 5	0%
bulk-report	0 / 5	0%
check	0 / 1,000	0%
check-block	0 / 100	0%
clear-address	0 / 5	0%
report	0 / 1,000	0%
reports	0 / 100	0%

AbuseIPDB » *184.82.123.11*

Check an IP Address, Domain Name, or Subnet
e.g. **184.82.123.11**, **microsoft.com**, or **5.188.10.0/24**

184.82.123.11 CHECK

184.82.123.11 was not found in our database

ISP	AIS Fibre
Usage Type	Fixed Line ISP
ASN	Unknown
Hostname(s)	184-82-123-0.24.public.tls1b-bcr02.myaisfibre.com
Domain Name	ais.th
Country	Thailand
City	Ban Bang Kadi, Pathum Thani

IP info including ISP, Usage Type, and Location provided by IPInfo. Updated biweekly.

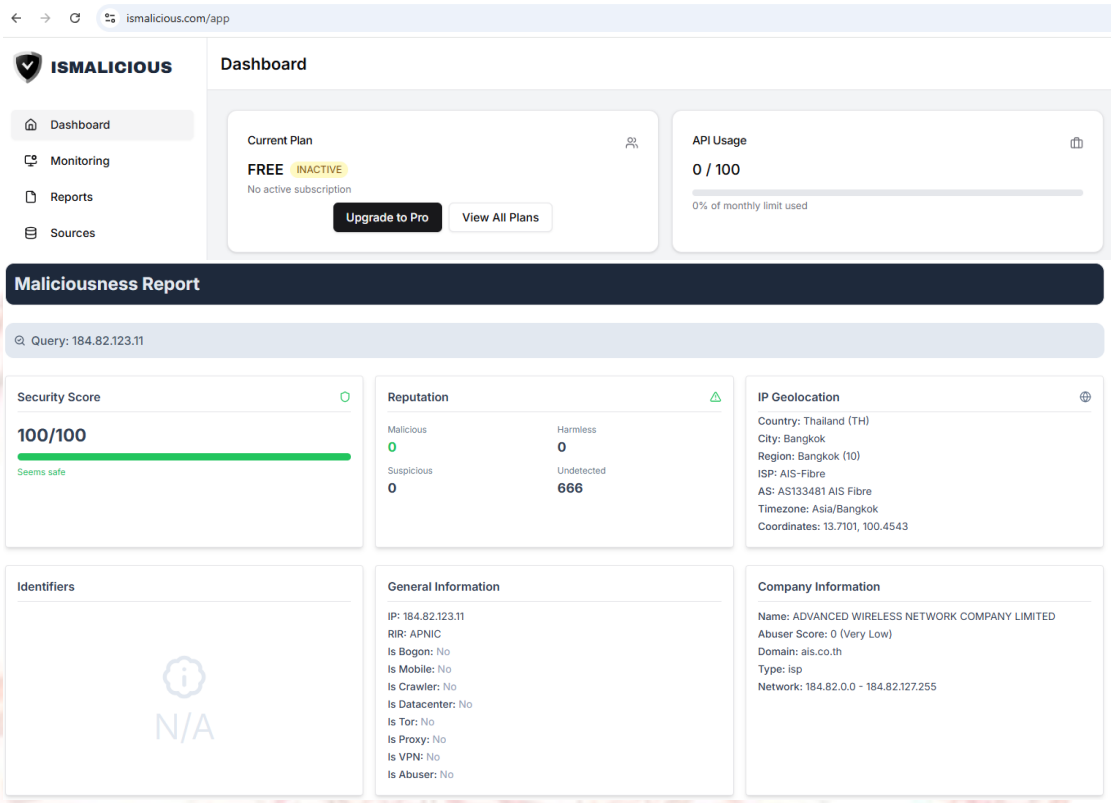
REPORT 184.82.123.11 WHOIS 184.82.123.11

IP Abuse Reports for **184.82.123.11**:

This IP address has not been reported. File Report

ภาพที่ 3-4 บริการ API สำหรับตรวจสอบ IP อันตรายของ AbuseIPDB

3) isMalicious มีการเปิดให้ใช้งานแบบไม่เสียค่าใช้จ่ายโดยมีข้อจำกัด ดังนี้ สามารถแจ้งรายงาน IP อันตรายได้ 100 ครั้ง/เดือน ดังภาพที่ 3-5 จากภาพแสดงถึงปริมาณการใช้งานในแพ็คเกจฟรี และ ตัวอย่างการเรียกใช้ API isMalicious



ภาพที่ 3-5 บริการ API สำหรับตรวจสอบ IP อันตรายของ isMalicious

3.1.4.4 การเลือกฐานข้อมูล ผู้วิจัยได้พิจารณาฐานข้อมูล 2 ประเภทได้แก่

1) ฐานข้อมูลเชิงสัมพันธ์ (Relational Database) ซึ่งมีข้อดีในการดำเนินการกับข้อมูลที่มีโครงสร้างชัดเจน สามารถตรวจสอบความครบถ้วนของข้อมูลได้ง่าย เหมาะกับการจัดเก็บข้อมูลขนาดใหญ่ที่มีโครงสร้างชัดเจน

2) ฐานข้อมูลไม่ใช่เชิงสัมพันธ์ (Non-Relational Database) ซึ่งมีข้อดีในการจัดเก็บข้อมูลที่ยืดหยุ่น เหมาะกับการจัดเก็บข้อมูลที่มีการเปลี่ยนแปลงโครงสร้างบ่อย

3) ทางผู้วิจัยได้ทดลองใช้งานฐานข้อมูลทั้ง 2 ประเภท พบว่า ทั้ง 2 ประเภทมีความเร็วในการอ่าน/เขียนข้อมูลที่ใกล้เคียงกัน และสามารถบันทึกข้อมูลลงในฐานข้อมูลได้อย่างไม่มีปัญหา แต่ในการดึงข้อมูลจำนวนมากเพื่อนำมาสอนแบบจำลอง พบว่า Relational Database มีความสะดวกในการใช้งานและสามารถตรวจสอบความครบถ้วนของข้อมูลได้มากกว่า Non-Relational Database อีกทั้ง Relational Database จัดเก็บข้อมูลในรูปแบบตารางซึ่งสามารถ Export ออกมาเป็นไฟล์ csv ได้ง่ายกว่า Non-Relational Database ที่จัดเก็บในรูปแบบที่ไม่ใช่ตาราง เช่น JSON ทางผู้วิจัยจึงเลือกใช้งาน Relational Database เป็นระบบฐานข้อมูลในการวิจัย

3.1.4.5 ผู้วิจัยได้ศึกษาการทำส่วนวิเคราะห์การเข้าสู่ระบบด้วยการทำ Classification ผู้วิจัยได้ดำเนินการปรับปรุงข้อมูล Dataset ที่ได้รับ และเลือก library ดังนี้

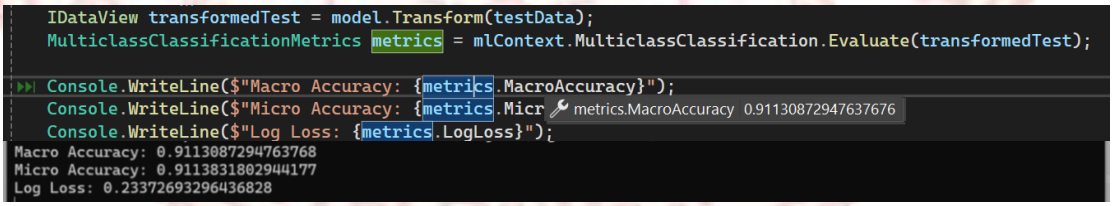
1) การเตรียมข้อมูลผู้วิจัยได้ค้นหา Dataset จาก platform ที่เผยแพร่ Dataset ข้อมูล log การเข้าสู่ระบบเดือนกุมภาพันธ์ 2563 จากนั้นนำมาวิเคราะห์พบว่าข้อมูลมีขนาด 8.4 GB และมีข้อมูลจำนวน 31,269,263 Record ซึ่งเป็นปริมาณที่มากจึงไม่สามารถเปิดได้ด้วย Microsoft Excel หรือ โปรแกรม Notepad ได้ ทางผู้วิจัยจึงต้องใช้โปรแกรมในการอ่านข้อมูลทีละ record แล้วจึงบันทึกแบ่งเป็น ไฟล์ละ 1,000,000 Record และคัดเลือกข้อมูลในปริมาณที่น้อยลงโดยคัดเลือกชุดข้อมูลตัวอย่างจำนวน 1,000,000 Record เพื่อให้ระบบสามารถ สอนแบบจำลอง ได้ จากนั้นจึงดำเนินการ Clean ข้อมูลโดยผู้วิจัยพบว่ามีข้อมูลที่จัดเก็บไว้แต่ไม่ได้ใช้งานจึงดำนเนินการลบข้อมูลคอลัมน์ที่ไม่ได้ใช้ และตรวจสอบค่าของคอลัมน์ IsAccountTakeover ซึ่งพบว่าข้อมูลคอลัมน์ IsAccountTakeover ใน Dataset ที่ผู้วิจัยจะนำไปใช้เป็นผลลัพธ์การทำ Classification เพื่อนำไปสอนแบบจำลอง นั้นมีข้อมูลที่มีค่าเป็น true เพียง 97 Record จากทั้งหมด ทางผู้วิจัยจึงเพิ่มคอลัมน์ใหม่ชื่อ IsSuspicious โดยมีเงื่อนไขคือหากมีพฤติกรรมการเข้าใช้งานระบบด้วย IP อันตรายสำเร็จ (LoginSuccessful และ isAttackIP เป็น true) หรือการเข้าถึงที่โดยไม่ได้รับอนุญาต (IsAccountTakeover) ให้ระบุคอลัมน์ IsSuspicious เป็น true ซึ่งมีข้อมูลเป็น true อยู่ที่ 32,108 Record

3.1.4.6 ผู้วิจัยวิเคราะห์การเลือกใช้ library โดยยกตัวอย่างมา 2 library ดังนี้

1) ML.Net ซึ่งสามารถพัฒนาได้ด้วย .NET framework ได้

2) Pytorch ซึ่งต้องพัฒนาโดย Framework ภายนอก เช่น Jupier Notebook

3) จากการทดสอบผู้วิจัยเลือกใช้ Classification ประเภท Multi Classification เพื่อใช้จำแนกพฤติกรรมการเข้าใช้งานที่มีความเสี่ยง โดยจากการทดสอบสร้างแบบจำลอง ด้วย 2 library ที่คัดเลือกเพื่อทดสอบประสิทธิภาพและความเหมาะสม ได้แก่ Pytorch โดยใช้วิธี Multi Classification โดยการ Import ข้อมูลไฟล์ Dataset และนำมาผลลัพธ์มาคิดเป็นคะแนน ได้คะแนนความแม่นยำอยู่ที่ 0.95 คะแนน และทดสอบ ML.NET โดยทำการ Import Dataset ชุดเดียวกันกับแบบจำลอง ก่อนหน้านำมาผลลัพธ์มาคิดเป็นคะแนน ได้คะแนนความแม่นยำอยู่ที่ 0.91 คะแนน เมื่อนำผลคะแนนความแม่นยำมาเทียบกันระหว่างแบบจำลอง ที่พัฒนาด้วย Pytorch และ ML.net ซึ่งพบว่า ความแม่นยำและความเร็วมีความแตกต่างกันเพียงเล็กน้อย ผู้วิจัยจึงดำเนินการเปรียบเทียบแบบจำลอง ในลำดับถัดไปโดยเลือกจากความสะดวกในการใช้งาน ซึ่งการนำแบบจำลอง ที่พัฒนาด้วย Pytorch ไปใช้งานร่วมกับระบบที่พัฒนาจำเป็นต้องใช้คำสั่งของ Jupiter Notebook เพื่อ ส่งออกแบบจำลอง ออกมาในรูปแบบไฟล์ Open Neural Network Exchange (ONNX) ก่อนจากนั้นจึงนำไปวางใน Path ที่ระบุในโปรแกรม ซึ่งต้องดำเนินการหลายขั้นตอนและยากต่อการพัฒนาให้เป็นรูปแบบอัตโนมัติ ซึ่งต่างจากแบบจำลอง ที่พัฒนาด้วย ML.Net ซึ่งเป็น library ที่พัฒนาเพื่อใช้งานร่วมกับ .NET Framework สามารถพัฒนาได้ในโปแกรม Virtual Studio และสามารถ Export เป็นแบบจำลอง เป็น Zip file วางที่ Path ในระบบได้โดยอัตโนมัติ ทางผู้วิจัยจึงเลือกใช้ ML.Net ซึ่งมีความสะดวก เหมาะสมกับเครื่องมือที่ใช้พัฒนามากกว่า Pytorch ในการพัฒนาส่วนตรวจสอบพฤติกรรม ดังภาพที่ 3-6 และ ภาพที่ 3-7 จากภาพแสดงถึงการวัดความแม่นยำของแบบจำลองที่พัฒนาด้วย ML.Net และ Pytorch ตามลำดับ



ภาพที่ 3-6 ผลการทดสอบ Classification ด้วย library ML.Net

ภาพที่ 3-7 ผลการทดสอบ Classification ด้วย library Pytorch

3.1.5 ในการเลือกใช้วิธีการยืนยันตัวตนแบบหลายขั้นตอน ซึ่งมีหลายวิธีให้เลือกใช้ แต่ด้วยงานวิจัยนี้เพียงต้องการเพิ่มการป้องกันเพื่อไม่ให้บัญชีถูกเข้าใช้งานโดยไม่ได้รับอนุญาตโดยเน้นไปที่การไม่เพิ่มภาระให้ผู้ใช้งานมากจนเกินไปและเป็นการดำเนินการที่ไม่มีค่าใช้จ่าย จากเงื่อนไขดังกล่าว ผู้วิจัยจึงเห็นว่าการใช้ยืนยันตัวตนแบบหลายขั้นตอน ชนิดการส่งข้อความยืนยันผ่านช่องทาง Email เพื่อให้ผู้ใช้งานเข้าไปยืนยันตัวตนเป็นทางเลือกที่สะดวก เหมาะสม โดยผู้วิจัยได้เพิ่มรหัส OTP ไว้แล้วด้วย กรณีผู้ใช้งานเห็นว่าการคลิกลิงก์ผ่านช่องทาง Email มีความเสี่ยง

3.1.6 ออกแบบแผนผัง Network Diagram ชุดเชื่อมต่อรับส่งข้อมูลระหว่างอุปกรณ์ภายในและส่วนที่สื่อสารกับภายนอก ประกอบด้วย 6 ส่วนหลัก ได้แก่

3.1.6.1 ส่วน Application ที่ติดต่อกับผู้ใช้งาน

1) ส่วนหน้าจอเข้าสู่ระบบ

2) หน้ายืนยันตัวตนด้วย OTP

3.1.6.2 ส่วน API ตรวจสอบการเข้าสู่ระบบ

1) API วิเคราะห์พฤติกรรมการเข้าสู่ระบบ

2) API ยืนยันตัวตน OTP

3) API สอนแบบจำลอง

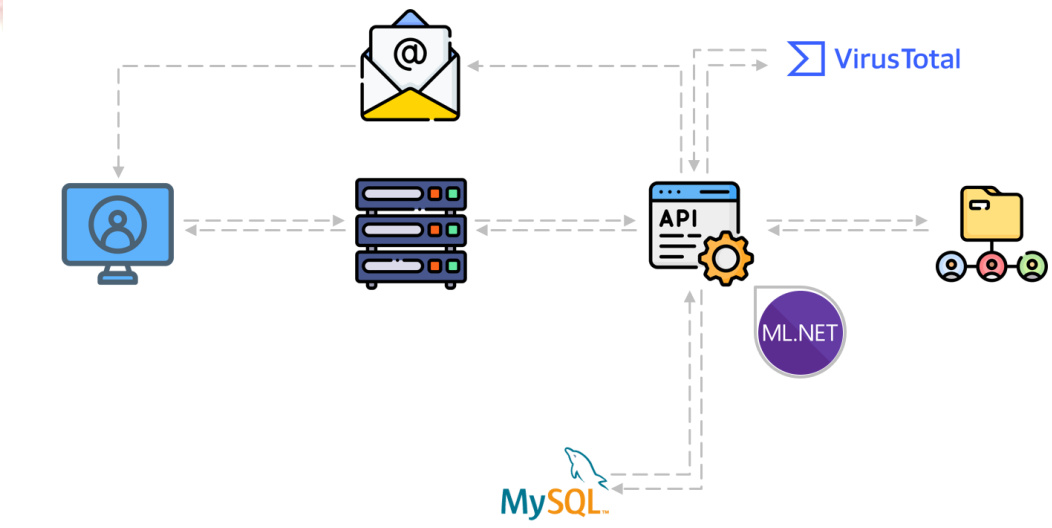
3.1.6.3 ส่วน Server ที่ติดตั้งบริการ Active Directory

3.1.6.4 ส่วนฐานข้อมูล

3.1.6.5 ส่วนการตรวจสอบ IP ด้วยบริการจากภายนอก

3.1.6.6 ส่วนการส่ง Email แจ้งเตือน OTP

สามารถสรุป Network Diagram ได้ดังภาพที่ 3-8



ภาพที่ 3-8 ภาพรวม Network Diagram ของระบบ

3.2 ดำเนินการ (DO)

3.2.1 หลังจากผู้วิจัยได้พิจารณาใช้ภาษา .Net C# ในการพัฒนาระบบจึงดำเนินการติดตั้งโปรแกรม Microsoft Visual Studio และสร้าง Project เป็น ASP.NET Core Web API

3.2.2 ผู้วิจัยได้เลือกใช้ฐานข้อมูลชนิด Relational Database โดยดำเนินการติดตั้ง Microsoft SQL Server และเชื่อมเข้ากับโปรแกรม Microsoft Visual Studio ที่ใช้พัฒนาโดยได้สร้างตารางที่เกี่ยวข้องกับระบบ 3 ตาราง ดังนี้

3.2.2.1 tb_allow_login เป็นตารางที่ใช้ในการตรวจสอบสถานะการรอการยืนยันตัวตน/ยืนยันตัวตนของผู้ใช้งาน ใช้ในการกำหนดอายุของรหัส OTP และความถี่ในการส่ง OTP หาผู้ใช้งานเพื่อยืนยันตัวตน ซึ่งประกอบด้วยข้อมูล 7 คอลัมน์ ได้แก่ Id guid otp_hash isAllow userid createdDat updatedDate โดยมีคำอธิบายตาราง tb_allow_login ดังตารางที่ 3-1

ตารางที่ 3-1 ตาราง tb_allow_login

ลำดับ	ชื่อคอลัมน์	คำอธิบาย
1	Id	ใช้เป็น Primary Key
2	guid	ใช้ส่งข้อมูลเพื่อระบุตัวตนแทน userid
3	otp_hash	จัดเก็บ OTP เพื่อตรวจสอบความถูกต้อง
4	isAllow	true ยืนยันตัวตนแล้ว false อยู่ระหว่างรอการยืนยันตัวตน
5	userid	ชื่อผู้ใช้งาน
6	createdDate	วันที่สร้าง
7	updatedDate	วันที่แก้ไขล่าสุด

3.2.2.2 tb_log เป็นตารางที่ใช้ในการตรวจสอบการทำงานของระบบ ซึ่งประกอบด้วยข้อมูล 7 คอลัมน์ ได้แก่ Id userId activity status description ipAddress createdDate โดยมีคำอธิบายตาราง tb_log ดังตารางที่ 3-2

ตารางที่ 3-2 ตาราง tb_log

ลำดับ	ชื่อคอลัมน์	คำอธิบาย
1	Id	ใช้เป็น Primary Key
2	userId	ชื่อผู้ใช้งาน

ตารางที่ 3-2 ตาราง tb_log (ต่อ)

ลำดับ	ชื่อคอลัมน์	คำอธิบาย
3	activity	การดำเนินการ
4	status	Success ยืนยันตัวตนแล้ว Fail อยู่ระหว่างรอการยืนยันตัวตน
5	description	รายละเอียดการดำเนินการ
6	ipAddress	IP Address ของผู้ดำเนินการ
7	createdDate	วันที่สร้าง

3.2.2.3 tb_login_log เป็นตารางที่ใช้ในบันทึกข้อมูลการเข้าสู่ระบบของผู้ใช้งาน เพื่อจัดเก็บเป็นข้อมูลสำหรับการ สอนแบบจำลอง ซึ่งประกอบด้วยข้อมูล 13 คอลัมน์ ดังตาราง Id userId loginTimeStamp ipAddress country userAgentString deviceType loginSuccessful isAttackIp isAccountTakeover isSuspicious createdDate updatedDate โดยมีคำอธิบายตาราง tb_login_log ดังตารางที่ 3-3

ตารางที่ 3-3 ตาราง tb_login_log

ลำดับ	ชื่อคอลัมน์	คำอธิบาย
1	Id	ใช้เป็น Primary Key
2	userId	ชื่อผู้ใช้งาน
3	loginTimeStamp	เวลาที่เข้าสู่ระบบ
4	ipAddress	IP Address ที่เข้าสู่ระบบ
5	country	ประเทศที่เข้าสู่ระบบ
6	userAgentString	ข้อมูล User-Agent ที่ได้จากผู้ใช้งาน
7	deviceType	ระบบปฏิบัติการที่ใช้เข้าสู่ระบบ
8	loginSuccessful	True เข้าสู่ระบบสำเร็จ Fale เข้าสู่ระบบไม่สำเร็จ
9	isAttackIp	True เป็น IP Address ที่อันตราย / ไม่สามารถตรวจสอบได้ False เป็น IP Address ปกติ
10	isAccountTakeover	True เป็นการเข้าถึงโดนไม่ได้รับอนุญาต False เป็นเข้าสู่ระบบแบบปกติ

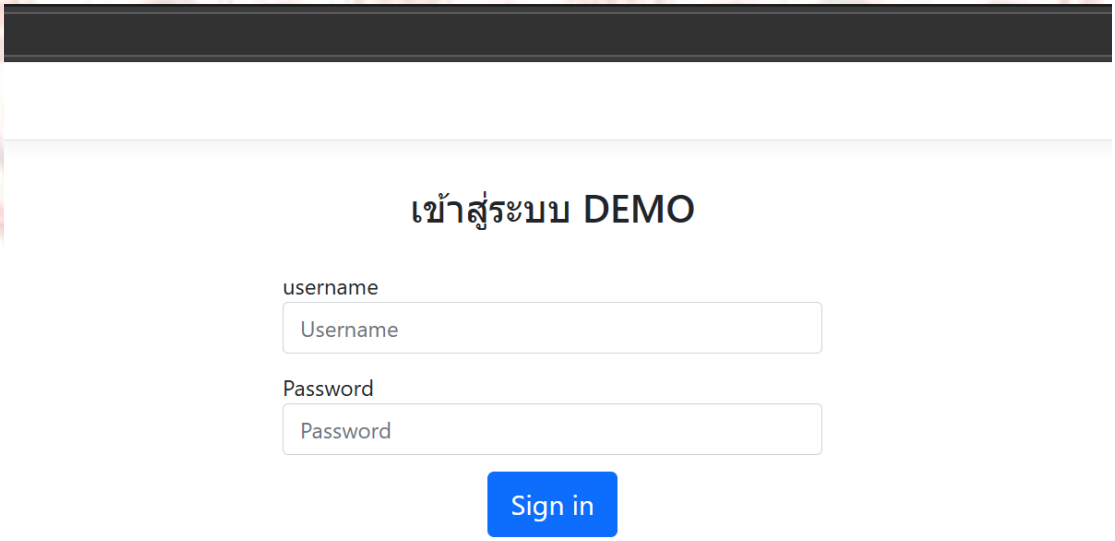
ตารางที่ 3-3 ตาราง tb_login_log (ต่อ)

ลำดับ	ชื่อคอลัมน์	คำอธิบาย
11	isSuspicious	True มีพฤติกรรมที่น่าสงสัย False มีพฤติกรรมปกติ
12	createdDate	วันที่สร้าง
13	updatedDate	วันที่แก้ไข

3.2.3 ดำเนินการติดตั้งโปรแกรม Oracle VirtualBox เพื่อสร้างเครื่องคอมพิวเตอร์เสมือนสำหรับติดตั้ง Window Server และติดตั้งบริการ Active Directory จากนั้นเพิ่มผู้ใช้งานสำหรับทดสอบระบบ

3.2.4 สร้างบัญชีผู้ใช้งาน Email และบัญชีผู้ใช้งานบริการตรวจสอบ IP ผ่าน API

3.2.5 พัฒนาหน้าจอเข้าสู่ระบบโดยการ New Project ตั้งชื่อ “LoginBehavior” จากนั้นสร้างหน้าจอเข้าสู่ระบบประกอบด้วยการ input ข้อมูล ชื่อผู้ใช้งาน รหัสผ่าน และปุ่มเข้าสู่ระบบ ดังภาพที่ 3-9 จากภาพแสดงถึงตัวอย่างหน้าจอการเข้าสู่ระบบ



ภาพที่ 3-9 ตัวอย่างหน้าจอการเข้าสู่ระบบ

3.2.6 พัฒนา API สำหรับตรวจสอบการเข้าสู่ระบบ โดยการ New Project ตั้งชื่อ “API” จากนั้นสร้าง APILogin ประกอบด้วย

3.2.6.1 ฟังก์ชันการเชื่อมต่อไปยัง Active Directory ดังภาพที่ 3-10 จากภาพแสดงถึงการพัฒนาส่วนเชื่อมต่อ Active Directory ในส่วนของ API

```
[HttpPost(Name = "Login")]
0 references
public async Task<APIResponse> Get(APILoginModel loginModel)
{
    SQLLogClass sqlLog = new SQLLogClass();
    GetUserInfoClass getUserInfo = new GetUserInfoClass();
    string ipAddress = getUserInfo.GetIPAddress(Request);
    string os = getUserInfo.GetOS(Request);
    string region = getUserInfo.GetRegion(Request);
    bool isIPSave = false;
    bool isAuthenticated = false;
    try
    {
        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("appsettings.json", optional: false, reloadOnChange: true)
            .Build();
        var ldapServer = configuration["Ldap:Server"];
        var userAttribute = configuration["Ldap:UserAttribute"];
        string ldapErrorMessage = null;

        var authenticator = new LDAPClass(ldapServer, userAttribute);
        isAuthenticated = authenticator.AuthenticateUser(loginModel.username, loginModel.password, out ldapErrorMessage);
```

ภาพที่ 3-10 ตัวอย่าง API ตรวจสอบการยืนยันตัวตนผ่าน Active Directory

3.2.6.2 ส่วนจัดเก็บข้อมูลรายละเอียดการเข้าสู่ระบบของผู้ใช้งาน และส่วนการเรียกใช้บริการตรวจสอบ IP อันตรายผ่าน API หากพบว่าระบบจะไม่อนุญาตให้เข้าใช้งาน ดังภาพที่ 3-11 จากภาพแสดงถึงการพัฒนาส่วนเชื่อมต่อไปยังบริการตรวจสอบ IP Address ของ Virustotal

```
        SQLIsAllowClass sqlIsAllow = new SQLIsAllowClass();
        SQLIsAllowModel checkIsAllow = sqlIsAllow.getIsAllow(loginModel.username, "");
        if (checkIsAllow.isAllow == "false")
        {
            sqlLog.insertLog(loginModel.username, "Login", "Fail", ipAddress, "Wait Confirm OTP");
            return new APIResponse("Wait Confirm OTP", checkIsAllow.guid);
        }
        else if (checkIsAllow.isAllow == "true")
        {
            sqlLog.insertLog(loginModel.username, "Login", "Success", ipAddress, "");
            return new APIResponse("Login Success", checkIsAllow.guid);
        }

        string ipRegexPattern = "(?:\\d{1,3}\\.){3}\\d{1,3}";
        if(Regex.IsMatch(ipAddress, ipRegexPattern))
        {

            try
            {
                using var client = new HttpClient();
                string url = $"https://www.virustotal.com/api/v3/ip_addresses/{ipAddress}";
                var apikey = configuration["ApiVirusTotal:apikey"];
                client.DefaultRequestHeaders.Add("x-apikey", apikey);
                HttpResponseMessage response = await client.GetAsync(url);
                response.EnsureSuccessStatusCode();
                APIIpAddressesResponse responseBody = await response.Content.ReadFromJsonAsync<APIIpAddressesResponse>();
                isIPSave = responseBody.data.attributes.last_analysis_stats.malicious > 0 ? false : true;
                sqlLog.insertLog(loginModel.username, "Call Virus Total", "Success", ipAddress, "");
            }
            catch (Exception ex)
            {
                sqlLog.insertLog(loginModel.username, "Call Virus Total", "Fail", ipAddress, ex.Message);
```

```
    if(!isIPSave) {
        sqlLog.insertLog(loginModel.username, "Login", "Fail", ipAddress, "Login By Attacker IP");
        return new APIResponse("Login By Attacker IP", checkIsAllow.guid);
    }
    else if (!IsSuspicious)
    {
        sqlLog.insertLog(loginModel.username, "Login", "Success", ipAddress, "");
        return new APIResponse("Login Success", checkIsAllow.guid);
    }
```

ภาพที่ 3-11 ตัวอย่าง API ตรวจสอบ IP Address อันตราย

3.2.6.3 ส่วนเชื่อมต่อฐานข้อมูล อ่านและเขียนข้อมูลลงบนฐานข้อมูลดังภาพที่ 3-12 จากภาพแสดงถึงการเรียกใช้ข้อมูลและการสร้างข้อมูลจากฐานข้อมูล

```
10      public int getKey()
11      {
12          int Key = 0;
13          try
14          {
15              string queryString = "SELECT MAX(id) FROM dbo.tb_log;";
16              using (SqlConnection connection = new SqlConnection(connectionString))
17              {
18                  using (SqlCommand command = new SqlCommand(queryString, connection))
19                  {
20                      if (connection.State == System.Data.ConnectionState.Closed)
21                      {
22                          connection.Open();
23                          Console.WriteLine("Start MySQL...");
24                      }
25                      using (SqlDataReader reader = command.ExecuteReader())
26                      {
27                          while (reader.Read())
28                          {
29                              Console.WriteLine(String.Format("{0}", reader[0]));
30                              Key = Int32.Parse(reader[0].ToString());
31                          }
32                      }
33
34                      int rowsAffected = command.ExecuteNonQuery();
35                      Console.WriteLine($"Successfully select {rowsAffected} row(s).");
36                  }
37              }
38          }
39          catch (SqlException ex)
40          {
41              Console.WriteLine($"SQL Error: {ex.Message}");
42              return 1;
43          }
44          catch (Exception ex)
52      public string insertLog(string userid, string activity, string status, string ipAddress, string description)
53      {
54          try
55          {
56              string sqlInsert = "INSERT INTO dbo.tb_log (id, userId, activity, status, description, ipAddress, cr
57              using (SqlConnection connection = new SqlConnection(connectionString))
58              {
59                  using (SqlCommand command = new SqlCommand(sqlInsert, connection))
60                  {
61                      if (connection.State == System.Data.ConnectionState.Closed)
62                      {
63                          connection.Open();
64                          Console.WriteLine("Start MySQL...");
65                      }
66                      command.Parameters.AddWithValue("@id", getKey());
67                      command.Parameters.AddWithValue("@userId", userid);
68                      command.Parameters.AddWithValue("@activity", activity);
69                      command.Parameters.AddWithValue("@status", status);
70                      command.Parameters.AddWithValue("@description", description);
71                      command.Parameters.AddWithValue("@ipAddress", ipAddress);
72                      command.Parameters.AddWithValue("@createdDate", DateTime.Now);
73
74                      int rowsAffected = command.ExecuteNonQuery();
75                      Console.WriteLine($"Successfully inserted {rowsAffected} row(s).");
76                  }
77              }
```

ภาพที่ 3-12 ตัวอย่าง API บันทึกข้อมูลลงฐานข้อมูล

3.2.6.4 พัฒนาส่วนการวิเคราะห์พฤติกรรมการเข้าสู่ระบบดังภาพที่ 3-13 จากภาพแสดงถึงการเรียกใช้ไฟล์แบบจำลองที่ผ่านการสอนแล้วจากนั้นนำเข้าข้อมูลเพื่อจำแนก IP Address อันตราย

```
94          var mlContext = new MLContext();
95          string modelPath = "TrainModel/LoginLogModel.zip";
96          DataViewSchema modelSchema;
97          ITransformer loadedModel = mlContext.Model.Load(modelPath, out modelSchema);
98          var predictionEngine = mlContext.Model.CreatePredictionEngine<LoginDataForTrain, LoginPrediction>(loadedModel);
99          DateTime LoginTimestamp = DateTime.Now;
100         LoginDataForTrain PredictData = new LoginDataForTrain
101         {
102             DayOfWeek = LoginTimestamp.DayOfWeek.ToString(),
103             Hour = LoginTimestamp.Hour.ToString(),
104             IPAddress = ipAddress,
105             Country = region,
106             OS = os,
107             Device = device,
108             Browser = browser
109         };
110         var prediction = predictionEngine.Predict(PredictData);
111         Console.WriteLine($"Prediction: {prediction.IsSuspicious} | Probability: {prediction.Probability:P2}");
112         bool IsSuspicious = prediction.IsSuspicious == "true" ? true : false;
```

ภาพที่ 3-13 ตัวอย่าง API ส่วนการวิเคราะห์พฤติกรรมการเข้าสู่ระบบ

3.2.6.5 พัฒนาฟังก์ชันการส่ง Email แจ้งเตือน OTPดังภาพที่ 3-14 จากภาพแสดงถึงขั้นตอนการส่ง Email ไปถึงผู้ใช้งานโดยใช้ข้อมูล Email จาก Active Directory

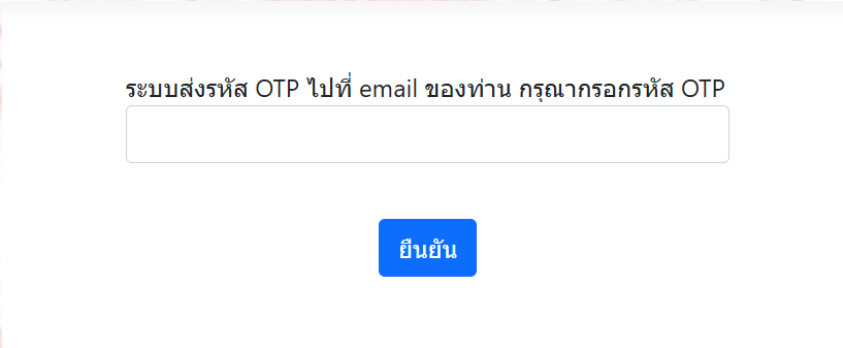
```
var fromAddress = new MailAddress("nattawee.kmutnb@gmail.com", "From Name");
var toAddress = new MailAddress(ldapModel.value, "To Name");
const string fromPassword = "XXXX XXXX XXXX XXXX";
const string subject = "No Reply: ยืนยันการเข้าสู่ระบบ";
string body = "";
body += "<!DOCTYPE html>";
body += "<html>";
body += "<head>";
body += "<title> ยืนยันตัวตนเข้าใช้งานระบบ DEMO </title>";
body += "</head>";
body += "<body>";
body += "<h1> ระบบ DEMO </h1>";
body += "<p> ท่านได้รับอีเมลนี้เนื่องจากระบบมีการตรวจพบการพยายามเข้าสู่ระบบที่ผิดปกติ </p>";
body += "<p> กรุณากรอก OTP: " + OTP6Digit + " ที่หน้าจอเข้าสู่ระบบ </p>";
body += "<p> หรือกด <a href='https://localhost:7082/checkotp?GUID=" + GUID + "&OTP=" + OTP6Digit + "'>ยืนยัน</a> ก่อนดำเนินการเข้าสู่ระบบอีกครั้ง</p>";
body += "<p> หากท่านไม่ได้เข้าสู่ระบบ โปรดเปลี่ยนรหัสผ่านเข้าระบบเนื่องจากรหัสผ่านของท่านมีการรั่วไหล </p>";
body += "</body>";
body += "</html>";

var smtp = new SmtpClient
{
    Host = "smtp.gmail.com",
    Port = 587,
    EnableSsl = true,
    DeliveryMethod = SmtpDeliveryMethod.Network,
    UseDefaultCredentials = false,
    Credentials = new NetworkCredential(fromAddress.Address, fromPassword)
};
////debug: C
using (var message = new MailMessage(fromAddress, toAddress)
{
    Subject = subject,
    Body = body,
    IsBodyHtml = true
})
```

ภาพที่ 3-14 ตัวอย่าง API ส่วนการส่ง Email ยืนยันตัวตน

3.2.7 พัฒนาหน้าจอสำหรับให้ผู้ใช้งานยื่นยันตัวตนโดยการคลิกปุ่ม ยืนยันตัวตน พร้อมทั้งลิงก์สำหรับให้ผู้ใช้งานคลิกเพื่อไปสู่หน้าจอการยืนยันตัวตนผ่าน OTP ดังภาพที่ 3-15 โดยจากภาพแสดงถึงหน้าจอการยืนยันตัวตนผ่าน Email

ภาพที่ 3-15 ตัวอย่างหน้าจอยืนยันตัวตน

3.2.8 พัฒนาหน้าจอการยืนยันตัวตนผ่าน OTP โดยมีช่องรับค่า OTP เมื่อผู้ใช้งานระบุ OTP จากนั้นคลิกยืนยัน ระบบจะไปส่งข้อมูลไปยัง API ยืนยันตัวตน ดังภาพที่ 3-16 โดยจากภาพแสดงถึงหน้าจอระบุรหัส OTP



ภาพที่ 3-16 ตัวอย่างหน้าจอยืนยันตัวตนผ่าน OTP

3.2.9 พัฒนา API สำหรับตรวจสอบการยืนยันตัวตนด้วย OTP โดยรับข้อมูล OTP และ guid ที่ได้จากหน้าจอนำมาตรวจอสอบกับข้อมูลที่จัดเก็บในตาราง tb_allow_login หากถูกต้องจะ Update ข้อมูลรายการ guid ของผู้ใช้งานรายนั้นเป็นอนุญาต (isAllow มีค่าเท่ากับ True) และ Redirect ไปสู่หน้าจอเข้าสู่ระบบสำเร็จ ดังภาพที่ 3-17 จากภาพแสดงถึง API การตรวจสอบการยืนยันตัวตนผ่าน OTP

```
namespace API.Controllers
{
    [ApiController]
    [Route("[controller]")]
    3 references
    public class CheckAuthenController : ControllerBase
    {
        private readonly ILogger<CheckAuthenController> _logger;

        0 references
        public CheckAuthenController(ILogger<CheckAuthenController> logger)
        {
            _logger = logger;
        }

        [HttpPost(Name = "CheckAuthen")]
        0 references
        public APIResponse Get(APICheckAuthenModel checkAuthenModel)
        {
            SQLLogClass sqlLog = new SQLLogClass();
            SQLIsAllowClass sqlIsAllow = new SQLIsAllowClass();
            GetUserInfoClass getUserInfo = new GetUserInfoClass();
            string ipAddress = getUserInfo.GetIPAddress(Request);

            SQLIsAllowModel isAllowModel = sqlIsAllow.getIsAllow("", checkAuthenModel.GUID);

            GetHashMD5 md5 = new GetHashMD5();
            if (md5.CreateMD5(checkAuthenModel.OTP) == isAllowModel.otp)
            {
                string checkIsAllow = sqlIsAllow.updateIsAllow(isAllowModel.guid);
                sqlLog.insertLog(isAllowModel.userid, "Authen", "Success", ipAddress, "");
                return new APIResponse("Authen Success", checkAuthenModel.GUID);
            }
            sqlLog.insertLog(isAllowModel.userid, "Authen", "Success", ipAddress, "");
            return new APIResponse("Authen Fail", checkAuthenModel.GUID);
        }
    }
}
```

```
namespace API.Class
{
    4 references
    public class GetHashMD5
    {
        2 references
        public string CreateMD5(string input)
        {
            using (System.Security.Cryptography.MD5 md5 = System.Security.Cryptography.MD5.Create())
            {
                byte[] inputBytes = System.Text.Encoding.ASCII.GetBytes(input);
                byte[] hashBytes = md5.ComputeHash(inputBytes);

                return Convert.ToHexString(hashBytes);
            }
        }
    }
}
```

ภาพที่ 3-17 ตัวอย่าง API ตรวจสอบการยืนยันตัวตน

3.2.10 พัฒนา API สำหรับการ สอนแบบจำลอง โดยการเรียกนำเข้าข้อมูลจากไฟล์ Dataset ที่วางในตำแหน่งที่ระบุมาใส่ในตัวแปร dataView จากนั้นนำมาแบ่งข้อมูลเป็นข้อมูลสำหรับสอน ร้อยละ 80 และข้อมูลสำหรับ Test ร้อยละ 20 กำหนดคอลัมน์ข้อมูลนำเข้าสำหรับ สอนแบบจำลอง และข้อมูลคอลัมน์ที่ต้องการให้แบบจำลอง ทำการแยกประเภท นำข้อมูลตัวแปรสำหรับสอน ข้อมูลเข้าสู่แบบจำลอง จากนั้นนำข้อมูลสำหรับ Test เข้าสู่แบบจำลอง เพื่อทดสอบ คำนวณคะแนนความแม่นยำ จากนั้น Export แบบจำลอง จัดเก็บไว้ที่ Path ที่ระบุไว้ ดังภาพที่ 3-18 จากภาพแสดงถึงการนำเข้าข้อมูลผ่านช่องทาง ไฟล์CSV

```
//// ***Train by Database***
//SQLLoginLogClass sqlLoginLog = new SQLLoginLogClass();
//List<SQLLoginLogModel> sqlLoginLogModel = sqlLoginLog.getLoginLog();
//List<LoginDataForTrain> sqlLoginLogList = new List<LoginDataForTrain>();
//foreach (SQLLoginLogModel temp in sqlLoginLogModel)
//{
//    sqlLoginLogList.Add(new LoginDataForTrain()
//    {
//        DayOfWeek = temp.dayOfWeek,
//        Hour = temp.hour.ToString(),
//        IPAddress = temp.ipAddress,
//        Country = temp.country,
//        OS = temp.os,
//        Device = temp.device,
//        Browser = temp.browser,
//        IsSuspicious = temp.isSuspicious
//    });
//}
//IEnumerable<LoginDataForTrain> dbData = sqlLoginLogList;
//IDataView dataView = mlContext.Data.LoadFromEnumerable(dbData);

//// ***Train by file***
IDataView dataView = mlContext.Data.LoadFromTextFile<LoginDataForTrain>(
    Path.GetFullPath(@"TrainModel/Data2V4.csv"),
    separatorChar: ',',
    hasHeader: true
    );

DataOperationsCatalog.TrainTestData dataSplit = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
IDataView trainData = dataSplit.TrainSet;
IDataView testData = dataSplit.TestSet;

Console.WriteLine($"Start: {DateTime.Now.ToString("HH:mm:ss")}");
var pipeline = mlContext.Transforms.Concatenate("AllText",
        nameof(LoginDataForTrain.DayOfWeek),
        nameof(LoginDataForTrain.Hour),
        nameof(LoginDataForTrain.IPAddress),
        nameof(LoginDataForTrain.Country),
        nameof(LoginDataForTrain.OS),
        nameof(LoginDataForTrain.Device),
        nameof(LoginDataForTrain.Browser))
    .Append(mlContext.Transforms.Text.FeaturizeText("Features", "AllText"))
    .Append(mlContext.BinaryClassification.Trainers.AveragedPerceptron("Label", "Features"));
```

ภาพที่ 3-18 ตัวอย่าง API สำหรับ สอนแบบจำลอง ส่วนอ่านไฟล์ csv

ภาพที่ 3-19 จากภาพแสดงถึงการสอนแบบจำลองด้วยวิธี Averaged Perceptron พร้อมทั้งการคำนวณคะแนนต่าง ๆ ของแบบจำลอง

```
70              DataOperationsCatalog.TrainTestData dataSplit = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
71              IDataView trainData = dataSplit.TrainSet;
72              IDataView testData = dataSplit.TestSet;
73
74              Console.WriteLine($"Start: {DateTime.Now.ToString("HH:mm:ss")}");
75              var pipeline = mlContext.Transforms.Concatenate("AllText",
76                      nameof(LoginDataForTrain.DayOfWeek),
77                      nameof(LoginDataForTrain.Hour),
78                      nameof(LoginDataForTrain.IPAddress),
79                      nameof(LoginDataForTrain.Country),
80                      nameof(LoginDataForTrain.OS),
81                      nameof(LoginDataForTrain.Device),
82                      nameof(LoginDataForTrain.Browser))
83                  .Append(mlContext.Transforms.Text.FeaturizeText("Features", "AllText"))
84                  .Append(mlContext.BinaryClassification.Trainers.AveragedPerceptron("Label", "Features"));
85
86              ITransformer model = pipeline.Fit(trainData);
87              IDataView transformedTest = model.Transform(testData);
88
89              var testSetTransform = model.Transform(testData);
90              var metrics = mlContext.BinaryClassification.EvaluateNonCalibrated(transformedTest);
91
92              Console.WriteLine("=============================================================");
93              Console.WriteLine("AveragedPerceptron ");
94              Console.WriteLine("=============================================================");
95              Console.WriteLine(metrics.ConfusionMatrix.GetFormattedConfusionTable());
96              Console.WriteLine($"Accuracy: {metrics.Accuracy}");
97              Console.WriteLine($"Precision: {metrics.PositivePrecision}");
98              Console.WriteLine($"Recall: {metrics.PositiveRecall}");
99              Console.WriteLine($"F1Score: {metrics.F1Score}");
100             Console.WriteLine("=============================================================");
101
102             Console.WriteLine($"Complete: {DateTime.Now.ToString("HH:mm:ss")}");
103             mlContext.Model.Save(model, dataView.Schema, "TrainModel/LoginLogModel.zip");
104
105                 sqlLog.insertLog(trainModel.username, "Train Model", "Success", ipAddress, "");
106             return new APIResponse("Train Model", "");
```

ภาพที่ 3-19 ตัวอย่าง API สำหรับ สอนแบบจำลอง

3.3 การตรวจสอบ (CHECK)

3.3.1 การตรวจสอบความปลอดภัยของเว็บไซต์ตามคำแนะนำของ OWASP Top Ten ผู้วิจัยได้ใช้โปรแกรม ZAP เพื่อทำการ Scan ในเบื้องต้นพบว่าระบบไม่มีช่องโหว่ที่เป็นอันตรายระดับ High โดยมีรายละเอียด ดังตารางที่ 3-4

ตารางที่ 3-4 ตารางผลการตรวจสอบเว็บไซต์ด้วยโปรแกรม Zap

		Confidence				
		User Confirmed	High	Medium	Low	Total
Risk	High	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	Medium	0 (0.0%)	1 (10.0%)	1 (10.0%)	0 (0.0%)	2 (20.0%)
	Low	0 (0.0%)	1 (10.0%)	2 (20.0%)	0 (0.0%)	3 (30.0%)
	Informational	0 (0.0%)	1 (10.0%)	2 (20.0%)	2 (20.0%)	5 (50.0%)
	Total	0 (0.0%)	3 (30.0%)	5 (50.0%)	2 (20.0%)	10 (100%)

Alert type	Risk	Count
Content Security Policy (CSP) Header Not Set	Medium	2 (20.0%)
Missing Anti-clickjacking Header	Medium	1 (10.0%)
Cookie Without Secure Flag	Low	1 (10.0%)
Strict-Transport-Security Header Not Set	Low	10 (100.0%)
X-Content-Type-Options Header Missing	Low	8 (80.0%)
GET for POST	Informational	1 (10.0%)
Information Disclosure - Suspicious Comments	Informational	2 (20.0%)
Re-examine Cache-control Directives	Informational	1 (10.0%)
Session Management Response Identified	Informational	2 (20.0%)
User Agent Fuzzer	Informational	24 (240.0%)
Total		10

3.3.1.1 ส่วนของ ความเสียงระดับ Medium ที่หลงเหลือยู่มีรายละเอียดดังนี้

Content Security Policy (CSP) Header Not Set เป็นความเสี่ยงระดับ Medium ซึ่งอาจช่องโหว่ให้การโจมตีแบบ Cross-Site Scripting (XSS) และการโจมตีแบบ Injection ดังภาพที่ 3-20 จากภาพแสดงถึงรายละเอียดของช่องโหว่ระดับ Medium: Content Security Policy (CSP) Header Not Set พร้อมทั้งแนวทางการแก้ไข

Content Security Policy (CSP) Header Not Set

Source	raised by a passive scanner (Content Security Policy (CSP) Header Not Set)
CWE ID	693
WASC ID	15
Reference	▪ https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP ▪ https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html ▪ https://www.w3.org/TR/CSP/ ▪ https://w3c.github.io/webappsec-csp/

ภาพที่ 3-20 รายละเอียด Content Security Policy (CSP) Header Not Set

Missing Anti-clickjacking Header เป็นความเสี่ยงระดับ Medium อาจเกิดการโจมตีแบบ Clickjacking โดยการหลอกให้ผู้ใช้คลิกตามตามตำแหน่งที่กำหนดโดยปกปิดหน้าจอไว้ ดังภาพที่ 3-21 จากภาพแสดงถึงรายละเอียดของช่องโหว่ระดับ Medium: Missing Anti-clickjacking Header พร้อมทั้งแนวทางการแก้ไข

Missing Anti-clickjacking Header

Source	raised by a passive scanner (Anti-clickjacking Header)
CWE ID	1021
WASC ID	15
Reference	▪ https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/X-Frame-Options

ภาพที่ 3-21 รายละเอียด Missing Anti-clickjacking Header

การตรวจสอบการ สอนแบบจำลอง ผู้วิจัยพบว่าข้อมูลที่ใช้มีปริมาณของข้อมูลที่น่าสงสัยน้อยเกินไปเพียง 32,108 Record จากข้อมูลทั้งหมด 1,000,000 Record คิดเป็น 3.21% ซึ่งเป็นอัตราส่วนที่ไม่เหมาะสม โดยผู้วิจัยได้แบ่งออกข้อมูลออกเป็น ข้อมูลสำหรับ สอนแบบจำลอง ร้อยละ 80 และสำหรับ Test ร้อยละ 20 ได้ผลการทดสอบ ดังภาพที่ 3-22 จากภาพแสดงถึงผลการทดสอบแบบจำลองด้วยวิธี LinearSvm

```
Start: 18:11:01
=====================================================================
LinearSvm
=====================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED ||  positive | negative | Recall
TRUTH     ||======================
 positive ||         0 |    6,476 | 0.0000
 negative ||         0 |  193,263 | 1.0000
          ||======================
Precision ||    0.0000 |   0.9676 |

Accuracy: 0.9675776888839936
Precision: 0
Recall: 0
F1Score: 0
=====================================================================
Complete: 18:11:30
```

ภาพที่ 3-22 ผลการทดสอบด้วยวิธี LinearSvm

3.4 การปรับปรุง (ACT)

3.4.1 จากผลคะแนนที่ได้ผู้วิจัยพบว่าข้อมูลที่นำมาใช้ สอนแบบจำลอง ยังมีบางคอลัมน์ที่สามารถปรับปรุงให้ชัดเจนยิ่งขึ้นโดยการแยกคอลัมน์ Timestamp แบ่งออกเป็นวันในสัปดาห์ (Day of Week) ชั่วโมง(Hour) และแบ่งคอลัมน์ UserAgentString เป็น OS และ Browser ซึ่งผู้วิจัยคิดว่าจะช่วยให้การจำแนกข้อมูลมีความชัดเจนมากขึ้น อีกทั้งผู้วิจัยปรับปรุงข้อมูลให้ตัวอย่างทั้ง 2 มีความใกล้เคียงกันโดยเพิ่มตัวอย่างข้อมูลที่น่าสงสัยเพิ่มขึ้นอีก แต่ด้วยข้อจำกัดของโปรแกรม Microsoft Excel ซึ่งแสดงข้อมูลได้สูงสุดที่ 1,048,576 Record รวมหัวตาราง ผู้วิจัยจึงปรับตัวอย่างชุดข้อมูลใหม่ ส่งผลให้ข้อมูลที่ใช้มีข้อมูลที่น่าสงสัยจำนวน 515,026 Record จากข้อมูลทั้งหมด 1,048,575 คิดเป็น 49.12%

3.4.2 ผู้วิจัยได้ทดลองเปลี่ยนชนิดการทำ Classification ของแบบจำลอง ที่นำมาวิเคราะห์พฤติกรรม โดยทดลองแบบจำลอง ดังนี้ AveragedPerceptron, LdSvm, LbfgsLogisticRegression, SdcaNonCalibrated, SdcaLogisticRegression, SgdNonCalibrated, SgdCalibrated, LinearSvm เพื่อหาวิธีที่เหมาะสมกับการใช้งานมากที่สุดและมีความแม่นยำมากขึ้น

3.4.3 หลังจากผู้วิจัยได้ทดสอบใช้งานระบบพบว่าระบบยังมีส่วนการ สอนแบบจำลอง ที่สามารถปรับปรุงให้ระบบมีความสะดวกในการใช้งานมากยิ่งขึ้น จากเดิมผู้ใช้งานต้องดึงข้อมูลจากตาราง tb_login_log จากนั้น Export ข้อมูลเป็นไฟล์ csv นำไปวางที่ Path ที่กำหนดจากนั้นเรียกใช้ API สอนModel เพื่อเริ่ม สอนแบบจำลอง ซึ่งทางผู้วิจัยปรับปรุงขั้นตอนการ สอนแบบจำลอง รายละเอียดดังนี้ ปรับปรุงให้ระบบสามารถดึงข้อมูลจากตาราง tb_login_log มาใส่ตัวแปร dataView โดยตรง จากนั้นนำมาแบ่งข้อมูลเป็นข้อมูลสำหรับ สอน ร้อยละ 80 และข้อมูลสำหรับ Test ร้อยละ 20 กำหนดคอลัมน์ข้อมูลนำเข้าสำหรับ สอนแบบจำลอง และข้อมูลคอลัมน์ที่ต้องการให้แบบจำลอง ทำการแยกประเภท นำข้อมูลตัวแปรสำหรับ สอน ข้อมูลเข้าสู่แบบจำลอง จากนั้นนำข้อมูลสำหรับ Test เข้าสู่แบบจำลอง เพื่อทดสอบ คำนวณคะแนนความแม่นยำ จากนั้น Exportแบบจำลอง จัดเก็บไว้ที่ Path ที่ระบุไว้ ซึ่งจากการตรวจสอบไฟล์ Dataset ดังภาพที่ 3-22 จากภาพแสดงถึงการนำเข้าข้อมูลผ่านฐานข้อมูล

```
42      // ***Train by Database***
43      SQLLoginLogClass sqlLoginLog = new SQLLoginLogClass();
44      List<SQLLoginLogModel> sqlLoginLogModel = sqlLoginLog.getLoginLog();
45      List<LoginDataForTrain> sqlLoginLogList = new List<LoginDataForTrain>();
46      foreach (SQLLoginLogModel temp in sqlLoginLogModel)
47      {
48          sqlLoginLogList.Add(new LoginDataForTrain()
49          {
50              DayOfWeek = temp.dayOfWeek,
51              Hour = temp.hour.ToString(),
52              IPAddress = temp.ipAddress,
53              Country = temp.country,
54              OS = temp.os,
55              Device = temp.device,
56              Browser = temp.browser,
57              IsSuspicious = temp.isSuspicious
58          });
59      }
60      IEnumerable<LoginDataForTrain> dbData = sqlLoginLogList;
61      IDataView dataView = mlContext.Data.LoadFromEnumerable(dbData);
```

ภาพที่ 3-23 API ส่วนการ สอนแบบจำลอง ด้วยข้อมูลจากฐานข้อมูล

3.4.4 ทางผู้วิจัยพบว่าระบบมีการจัดเก็บรหัส OTP ไว้ตรวจสอบความถูกต้อง ซึ่งเป็นการเก็บที่ไม่ได้นำมาวิเคราะห์ต่อ จึงสามารถปรับเปลี่ยนการจัดเก็บเป็นรูปแบบ Hash ได้ซึ่งยังคงสามารถนำ OTP ที่กรอกมาแปลงเป็นค่า Hash และนำไปตรวจสอบความถูกต้องได้

บทที่ 4
ผลการวิจัย

ผู้วิจัยได้ดำเนินการทดสอบเปรียบเทียบด้วยการสอนแบบจำลองต่าง ๆ และได้คำนวณคะแนนเพื่อเปรียบเทียบประสิทธิภาพของแบบจำลอง ดังนี้

4.1 ผลการศึกษา วิเคราะห์ การตรวจสอบพฤติกรรมการเข้าสู่ระบบที่ผิดปกติด้วยการทำ Binary Classification

การทดสอบด้วย Confusion Matrix ผู้วิจัยได้ทดสอบด้วยข้อมูล ดังนี้

4.1.1 ผู้วิจัยได้ดำเนินการเรียกใช้ API สอนแบบจำลอง โดยนำเข้าข้อมูลผ่านไฟล์ พบว่าระบบสามารถ สอนแบบจำลอง ได้สำเร็จ โดยผู้วิจัยได้ทำการทดลอง Binary Classification ด้วยข้อมูล Test จำนวน 199,739 Record สามารถคำนวณค่า Accuracy, Recall, Precision และ F1 Score ได้ตามสูตร

4.1.1.1 Accuracy

= (จำนวนที่ทำนายถูก) / (จำนวนทั้งหมด)

= (TP + TN) / (TP + FP + TN + FN)

4.1.1.2 Recall

= (จำนวนที่ทำนายว่าน่าสงสัยแล้วถูก) / (จำนวนที่ทำนายว่าน่าสงสัยทั้งหมด)

= TP / (TP + FN)

4.1.1.3 Precision

= (จำนวนที่ทำนายว่าน่าสงสัยแล้วถูก) / (จำนวนข้อมูลที่น่าสงสัยทั้งหมด)

= TP / (TP + FP)

4.1.1.4 F1 Score

= 2 x (Recall + Precision) / (Recall x Precision)

4.1.2 จากสูตรข้างต้นผู้วิจัยได้ดำเนินการคำนวณคะแนนจากแบบจำลอง ทั้ง 8 วิธี

4.2 วิธี Averaged Perceptron ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.2.1 ใช้เวลา 38 วินาที

4.2.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 1,719 ครั้ง

4.2.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 4,757 ครั้ง

4.2.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 2,453 ครั้ง

4.2.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 190,810 ครั้ง

4.2.6 ความแม่นยำ (Accuracy) อยู่ที่ (1,719 + 190,810) / (1,719 + 4,757 + 2,453 + 190,810) = 0.9639

4.2.7 Recall อยู่ที่ 1,719 / (1,719 + 2,453) = 0.412

4.2.8 Precision อยู่ที่ 1,719 / (1,719 + 4,757) = 0.2654

4.2.9 F1 Score = 2 x (0.412 + 0.2654) / (0.412 x 0.2654) = 0.3229

จากภาพที่ 4-24 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี Averaged Perceptron จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากที่สุด จำนวน 4,172 ครั้ง และมีคะแนน F1 Score มากที่สุด แต่มีจำนวนการทำนายถูกน้อยที่สุดเมื่อเทียบกับวิธีอื่น ๆ

```
Start: 16:51:50
================================================================
AveragedPerceptron
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||    1,719 |    4,757 | 0.2654
 negative ||    2,453 |  190,810 | 0.9873
          ||======================
Precision ||   0.4120 |   0.9757 |

Accuracy: 0.9639028932757249
Precision: 0.412032598274209
Recall: 0.2654416306361952
F1Score: 0.32287753568745303
================================================================
Complete: 16:52:28
```

ภาพที่ 4-24 ผลการทดสอบด้วยวิธี AveragedPerceptron ครั้งที่ 1

4.3 วิธี Limited-memory Broyden-Fletcher-Goldfarb-Shanno Logistic Regression (LbfgsLogisticRegression) ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.3.1 ใช้เวลา 1 นาที 14 วินาที

4.3.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 151 ครั้ง

4.3.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,325 ครั้ง

4.3.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 162 ครั้ง

4.3.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,101 ครั้ง

4.3.6 ความแม่นยำ (Accuracy) อยู่ที่ (151 + 193,101) / (151 + 6,325 + 162 + 193,101) = 0.9675

4.3.7 Recall อยู่ที่ 151 / (151 + 162) = 0.4824

4.3.8 Precision อยู่ที่ 151 / (151 + 6,325) = 0.0233

4.3.9 F1 Score = 2 x (0.4824 + 0.0233) / (0.4824 x 0.0233) = 0.0445

จากภาพที่ 4-25 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LbfgsLogistic Regression จะสังเกตได้ว่าเป็นวิธีที่ใช้เวลาในการสอนแบบจำลองเร็วเป็นลำดับที่ 2 และมีค่า Recall มากเป็นลำกับที่ 2

```
Start: 16:54:42
================================================================
LbfgsLogisticRegression
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||      151 |    6,325 | 0.0233
 negative ||      162 |  193,101 | 0.9992
          ||======================
Precision ||   0.4824 |   0.9683 |

Accuracy: 0.9675226170152048
Precision: 0.48242811501597443
Recall: 0.023316862260654724
F1Score: 0.044483723670643 69
================================================================
Complete: 16:55:56
```

ภาพที่ 4-25 ผลการทดสอบด้วยวิธี LbfgsLogisticRegression ครั้งที่ 1

4.4 วิธี Least Squares Support Vector Machine (LdSvm) ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.4.1 ใช้เวลา 36 นาที 4 วินาที

4.4.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 0 ครั้ง

4.4.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,476 ครั้ง

4.4.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 0 ครั้ง

4.4.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,263 ครั้ง

4.4.6 ความแม่นยำ (Accuracy) อยู่ที่ (0 + 193,263) / (0 + 6,476 + 0 + 193,263) = 0.9676

4.4.7 Recall อยู่ที่ 0 / (0 + 0) = 0

4.4.8 Precision อยู่ที่ 0 / (0 + 6,476) = 0

4.4.9 F1 Score = 2 x (0 + 0) / (0 x 0) = 0

จากภาพที่ 4-26 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LdSvm จะสังเกตได้ว่าเป็นวิธีใช้เวลาในการสอนแบบจำลองนานที่สุดและทำนายถูกมากที่สุดเมื่อเทียบกับวิธีอื่น ๆ อีกทั้งเป็นวิธีที่ทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติจำนวน 0 ครั้ง

```
Start: 17:33:04
================================================================
LdSvm
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||=====================
PREDICTED || positive | negative | Recall
TRUTH     ||=====================
 positive ||        0 |    6,476 | 0.0000
 negative ||        0 |  193,263 | 1.0000
          ||=====================
Precision ||   0.0000 |   0.9676 |

Accuracy: 0.9675776888839936
Precision: 0
Recall: 0
F1Score: 0
================================================================
Complete: 18:09:08
```

ภาพที่ 4-26 ผลการทดสอบด้วยวิธี LdSvm ครั้งที่ 1

4.5 วิธี Linear Support Vector Machine (LinearSvm) ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.5.1 ใช้เวลา 29 วินาที

4.5.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 0 ครั้ง

4.5.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,476 ครั้ง

4.5.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 0 ครั้ง

4.5.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,263 ครั้ง

4.5.6 ความแม่นยำ (Accuracy) อยู่ที่ (0 + 193,263) / (0 + 6,476 + 0 + 193,263) = 0.9676

4.5.7 Recall อยู่ที่ 0 / (0 + 0) = 0

4.5.8 Precision อยู่ที่ 0 / (0 + 6,476) = 0

4.5.9 F1 Score = 2 x (0 + 0) / (0 x 0) = 0

จากภาพที่ 4-27 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LinearSvm สังเกตได้ว่าเป็นวิธีใช้เวลาในการสอนแบบจำลองเร็วที่สุดเมื่อเทียบกับวิธีอื่น ๆ และทำนายถูกมากเป็นลำดับที่ 2 และเป็นวิธีที่ทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติจำนวน 0 ครั้ง

```
Start: 18:11:01
================================================================
LinearSvm
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||        0 |    6,476 | 0.0000
 negative ||        0 |  193,263 | 1.0000
          ||======================
Precision ||   0.0000 |   0.9676 |

Accuracy: 0.9675776888839936
Precision: 0
Recall: 0
F1Score: 0
================================================================
Complete: 18:11:30
```

ภาพที่ 4-27 ผลการทดสอบด้วยวิธี LinearSvm ครั้งที่ 1

4.6 วิธี Stochastic Dual Coordinate Ascent Logistic Regression (SdcaLogisticRegression) ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.6.1 ใช้เวลา 3 นาที 35 วินาที

4.6.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 246 ครั้ง

4.6.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,230 ครั้ง

4.6.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 297 ครั้ง

4.6.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 192,966 ครั้ง

4.6.6 ความแม่นยำ (Accuracy) อยู่ที่ (246 + 192,966) / (246 + 6,230 + 297 + 192,966) = 0.9673

4.6.7 Recall อยู่ที่ 246 / (246 + 297) = 0.453

4.6.8 Precision อยู่ที่ 246 / (246 + 6,230) = 0.038

4.6.9 F1 Score = 2 x (0.453 + 0.038) / (0.453 x 0.038) = 0.0701

จากภาพที่ 4-28 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SdcaLogistic Regression จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากเป็นลำดับที่ 2 จำนวน 543 ครั้ง และมีคะแนน F1 Score มากเป็นลำดับที่ 2

```
Start: 18:21:44
================================================================
SdcaLogisticRegression
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||      246 |    6,230 | 0.0380
 negative ||      297 |  192,966 | 0.9985
          ||======================
Precision ||   0.4530 |   0.9687 |

Accuracy: 0.9673223556741548
Precision: 0.4530386740331492
Recall: 0.03798641136504015
F1Score: 0.07009545519304744
================================================================
Complete: 18:25:19
```

ภาพที่ 4-28 ผลการทดสอบด้วยวิธี SdcaLogisticRegression ครั้งที่ 1

4.7 วิธี Stochastic Dual Coordinate Ascent Logistic Regression Non Calibrated (SdcaNonCalibrated) ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.7.1 ใช้เวลา 2 นาที 7 วินาที

4.7.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 185 ครั้ง

4.7.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,291 ครั้ง

4.7.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 220 ครั้ง

4.7.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,043 ครั้ง

4.7.6 ความแม่นยำ (Accuracy) อยู่ที่ (185 + 193,043) / (185 + 6,291 + 220 + 193,043) = 0.9674

4.7.7 Recall อยู่ที่ 185 / (185 + 220) = 0.4568

4.7.8 Precision อยู่ที่ 185 / (185 + 6,291) = 0.0286

4.7.9 F1 Score = 2 x (0.4568 + 0.0286) / (0.4568 x 0.0286) = 0.0538

จากภาพที่ 4-29 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SdcaNon Calibrated จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากเป็นลำดับที่ 3 จำนวน 405 ครั้ง และมีคะแนน F1 Score มากเป็นลำดับที่ 3

```
Start: 18:26:29
================================================================
SdcaNonCalibrated
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||=======================
PREDICTED || positive | negative | Recall
TRUTH     ||=======================
 positive ||      185 |    6,291 | 0.0286
 negative ||      220 |  193,043 | 0.9989
          ||=======================
Precision ||   0.4568 |   0.9684 |

Accuracy: 0.9674024602105749
Precision: 0.4567901234567901
Recall: 0.02856701667696109
F1Score: 0.05377125417817178
================================================================
Complete: 18:28:36
```

ภาพที่ 4-29 ผลการทดสอบด้วยวิธี SdcaNonCalibrated ครั้งที่ 1

4.8 วิธี SgdCalibrated ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.8.1 ใช้เวลา 4 นาที 30 วินาที

4.8.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 135 ครั้ง

4.8.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,341 ครั้ง

4.8.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 144 ครั้ง

4.8.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,119 ครั้ง

4.8.6 ความแม่นยำ (Accuracy) อยู่ที่ (135 + 193,119) / (135 + 6,341 + 144 + 193,119) = 0.9675

4.8.7 Recall อยู่ที่ 135 / (135 + 144) = 0.4839

4.8.8 Precision อยู่ที่ 135 / (135 + 6,341) = 0.0208

4.8.9 F1 Score = 2 x (0.4839 + 0.0208) / (0.4839 x 0.0208) = 0.04

จากภาพที่ 4-30 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SgdCalibrated จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายถูกมากเป็นลำดับที่ 3 และมีค่า Recall มากที่สุด เมื่อเทียบกับวิธีอื่น ๆ

```
Start: 18:30:47
================================================================
SgdCalibrated
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||      135 |    6,341 | 0.0208
 negative ||      144 |  193,119 | 0.9993
          ||======================
Precision ||   0.4839 |   0.9682 |

Accuracy: 0.9675326300822573
Precision: 0.4838709677419355
Recall: 0.020846201358863496
F1Score: 0.039970392301998524
================================================================
Complete: 18:35:17
```

ภาพที่ 4-30 ผลการทดสอบด้วยวิธี SgdCalibrated ครั้งที่ 1

4.9 วิธี SgdNonCalibrated ครั้งที่ 1 ได้ผลลัพธ์ ดังนี้

4.9.1 ใช้เวลา 3 นาที 59 วินาที

4.9.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 135 ครั้ง

4.9.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 6,341 ครั้ง

4.9.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 151 ครั้ง

4.9.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 193,112 ครั้ง

4.9.6 ความแม่นยำ (Accuracy) อยู่ที่ (135 + 193,112) / (135 + 6,341 + 151 + 193,112) = 0.9675

4.9.7 Recall อยู่ที่ 135 / (135 + 151) = 0.472

4.9.8 Precision อยู่ที่ 135 / (135 + 6,341) = 0.0208

4.9.9 F1 Score = 2 x (0.472 + 0.0208) / (0.472 x 0.0208) = 0.0399

จากภาพที่ 4-31 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SgdNon Calibrated จะสังเกตได้ว่าเป็นวิธีที่ใช้เวลาในการสอนแบบจำลองนานเป็นลำดับที่ 3 และมีค่า Recall มากเป็นลำดับที่ 3

```
Start: 18:36:11
================================================================
SgdNonCalibrated
================================================================
TEST POSITIVE RATIO:    0.0324 (6476.0/(6476.0+193263.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||      135 |    6,341 | 0.0208
 negative ||      151 |  193,112 | 0.9992
          ||======================
Precision ||   0.4720 |   0.9682 |

Accuracy: 0.96749758434757 36
Precision: 0.47202797202797203
Recall: 0.020846201358863496
F1Score: 0.03992901508429459
================================================================
Complete: 18:40:10
```

ภาพที่ 4-31 ผลการทดสอบด้วยวิธี SgdNonCalibrated ครั้งที่ 1

4.10 การปรับปรุงชุดข้อมูล

ผู้วิจัยได้นำผลลัพธ์ของแต่ละวิธีมาคำนวณ Accuracy, Recall, Precision และ F1 Score ซึ่งจากการทดลองจะเห็นว่าแบบจำลอง ให้ค่า Accuracy สูง แต่ค่า Recall Precision และ F1 Score ต่ำ โดยจะพบว่าวิธี AveragedPerceptron มีค่า F1 Score สูงที่สุด ตามมาด้วยวิธี SdcaLogisticRegression, SdcaNonCalibrated, LbfgsLogisticRegression, SgdCalibrated , SgdNonCalibrated, LdSvm และ LinearSvm ตามลำดับ แต่ค่า F1 Score ที่ได้ยังมีค่าที่ต่ำมากไม่สามารถนำไปใช้งานจริงได้ ผู้วิจัยจึงเตรียมชุดข้อมูลพฤติกรรมการเข้าสู่ระบบชุดใหม่จำนวน 1,048,575 Record แบ่งเป็น โดยเพิ่มสัดส่วนข้อมูล ดังนี้

ข้อมูลที่การเข้าสู่ระบบที่ปกติ จำนวน 533,549 Record

ข้อมูลที่การเข้าสู่ระบบที่น่าสงสัย จำนวน 515,026 Record

และเนื่องจาก Dataset ที่ได้มาจากอินเทอร์เน็ตยังเก็บข้อมูลในรูปแบบคอลัมน์ Timestamp และ UserAgentString ซึ่งนำมาวิเคราะห์ได้ยาก ผู้วิจัยจึงเปลี่ยนเป็นจัดเก็บในรูปแบบวันในสัปดาห์ (Day of Week) ชั่วโมง(Hour) แทนคอลัมน์ Timestamp และในรูปแบบ OS และ Browser แทนคอลัมน์ UserAgentString จากนั้นนำข้อมูล Timestamp และ UserAgentString ออก โดยผู้วิจัยได้ใช้โปรแกรม Microsoft Excel ทำการจำแนกข้อมูลคอลัมน์ OS และ Browser ออกมาจาก UserAgent ตามสูตรซึ่งเป็นเงื่อนไขเดียวกับที่ใช้ในระบบที่พัฒนาขึ้น ดังนี้

4.10.1 จากภาพที่ 4-32 เป็นการแยกประเภท OS จาก AgentString ด้วยโปรแกรม Microsoft Excel

```
=IF(ISNUMBER(SEARCH("Windows NT",G2)),"Windows",
IF(OR(ISNUMBER(SEARCH("Macintosh",G2)),ISNUMBER(SEARCH("Mac OS X",G2))),"macOS",
IF(ISNUMBER(SEARCH("Linus",G2)),"Linux",
IF(ISNUMBER(SEARCH("Android",G2)),"Android",
IF(OR(ISNUMBER(SEARCH("iPhone",G2)),ISNUMBER(SEARCH("iPad",G2)),ISNUMBER(SEARCH("iPod",G2))),"IOS","Unknown")))))
```

ภาพที่ 4-32 สูตร Excel สำหรับจำแนกคอลัมน์ OS จากคอลัมน์ UserAgent

4.10.2 จากภาพที่ 4-33 เป็นการแยกประเภท OS จาก AgentString ด้วยโปรแกรม Visual Studio

```
string userAgent = request.Headers["User-Agent"].ToString();
if (!string.IsNullOrEmpty(userAgent))
{
    if (userAgent.Contains("Windows NT"))
    {
        return "Windows";
    }
    else if (userAgent.Contains("Macintosh") || userAgent.Contains("Mac OS X"))
    {
        return "macOS";
    }
    else if (userAgent.Contains("Linux"))
    {
        return "Linux";
    }
    else if (userAgent.Contains("Android"))
    {
        return "Android";
    }
    else if (userAgent.Contains("iPhone") || userAgent.Contains("iPad") || userAgent.Contains("iPod"))
    {
        return "iOS";
    }
}
return "Unknown";
```

ภาพที่ 4-33 สูตร C# สำหรับจำแนกคอลัมน์ OS จากคอลัมน์ UserAgent

4.10.3 จากภาพที่ 4-34 เป็นการแยกประเภท Browser จาก AgentString ด้วยโปรแกรม Visual Studio

```
=IF(OR(ISNUMBER(SEARCH("Edge",G2)),ISNUMBER(SEARCH("Edg",G2))),"Edge",
IF(ISNUMBER(SEARCH("Chrome",G2)),"Chrome",
IF(ISNUMBER(SEARCH("Firefox",G2)),"Firefox",
IF(OR(ISNUMBER(SEARCH("Safari",G2)),ISNUMBER(SEARCH("Version",G2))),"Safari",
IF(OR(ISNUMBER(SEARCH("MSIE",G2)),ISNUMBER(SEARCH("Trident",G2))),"Internet Explorer","Unknown")))))
```

ภาพที่ 4-34 สูตร Excel สำหรับจำแนกคอลัมน์ Browser จากคอลัมน์ UserAgent

4.10.4 จากภาพที่ 4-35 เป็นการแยกประเภท Browser จาก AgentString ด้วยโปรแกรม Microsoft Excel

```
string userAgent = request.Headers["User-Agent"].ToString();
if (!string.IsNullOrEmpty(userAgent))
{
    if (userAgent.Contains("Edge") || userAgent.Contains("Edg"))
    {
        return "Edge";
    }
    else if (userAgent.Contains("Chrome"))
    {
        return "Chrome";
    }
    else if (userAgent.Contains("Firefox"))
    {
        return "Firefox";
    }
    else if (userAgent.Contains("Safari") || userAgent.Contains("Version"))
    {
        return "Safari";
    }
    else if (userAgent.Contains("MSIE") || userAgent.Contains("Trident"))
    {
        return "Internet Explorer";
    }
}
return "Unknown";
```

ภาพที่ 4-35 สูตร C# สำหรับจำแนกคอลัมน์ Browser จากคอลัมน์ UserAgent

4.10.5 ผู้วิจัยได้ดำเนินการเรียกใช้ API TrainModel ที่ปรับปรุงให้สอดคล้องกับชุดข้อมูลใหม่ โดยนำเข้าข้อมูล Test จำนวน 209,507 Record สามารถคำนวณค่า Accuracy, Recall, Precision และ F1 Score จากสูตรข้างต้นของแบบจำลอง ทั้ง 8 วิธี ดังนี้

4.11 วิธี Averaged Perceptron ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.11.1 ใช้เวลา 15 วินาที

4.11.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 97,291 ครั้ง

4.11.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 5,760 ครั้ง

4.11.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 8,670 ครั้ง

4.11.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 97,786 ครั้ง

4.11.6 ความแม่นยำ (Accuracy) อยู่ที่ (97,291 + 97,786) / (97,291 + 5,760 + 8,670 + 97,786) = 0.9311

4.11.7 Recall อยู่ที่ 97,291 / (97,291 + 8,670) = 0.9182

4.11.8 Precision อยู่ที่ 97,291 / (97,291 + 5,760) = 0.9441

4.11.9 F1 Score = 2 x (0.9182 + 0.9441) / (0.9182 x 0.9441) = 0.931

จากภาพที่ 4-36 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี Averaged Perceptron ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากที่สุดจำนวน 105,961 ครั้ง มีการทำนายถูกมากที่สุด และมีคะแนน F1 Score มากที่สุด เมื่อเทียบกับวิธีอื่น ๆ

```
Start: 21:56:23
=======================================================================
AveragedPerceptron
=======================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   97,291 |    5,760 | 0.9441
 negative ||    8,670 |   97,786 | 0.9186
          ||======================
Precision ||   0.9182 |   0.9444 |

Accuracy: 0.9311240197224914
Precision: 0.9181774426439917
Recall: 0.9441053458966919
F1Score: 0.9309609017664058
=======================================================================
Complete: 21:56:38
```

ภาพที่ 4-36 ผลการทดสอบด้วยวิธี AveragedPerceptron ครั้งที่ 2

4.12 วิธี Limited-memory Broyden-Fletcher-Goldfarb-Shanno Logistic Regression (LbfgsLogisticRegression) ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.12.1 ใช้เวลา 42 วินาที

4.12.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 94,307 ครั้ง

4.12.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 8,744 ครั้ง

4.12.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 10,635 ครั้ง

4.12.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 95,821 ครั้ง

4.12.6 ความแม่นยำ (Accuracy) อยู่ที่ (94,307 + 95,821) / (94,307 + 8,744 + 10,635 + 95,821) = 0.9075

4.12.7 Recall อยู่ที่ 94,307 / (94,307 + 10,635) = 0.8987

4.12.8 Precision อยู่ที่ 94,307 / (94,307 + 8,744) = 0.9151

4.12.9 F1 Score = 2 x (0.8987 + 0.9151) / (0.8987 x 0.9151) = 0.9068

จากภาพที่ 4-37 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LbfgsLogistic Regression ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายถูกมากเป็นลำดับที่ 2 และมีค่า Precision มากเป็นลำดับที่ 2

```
Start: 21:58:59
================================================================
LbfgsLogisticRegression
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   94,307 |    8,744 | 0.9151
 negative ||   10,635 |   95,821 | 0.9001
          ||======================
Precision ||   0.8987 |   0.9164 |

Accuracy: 0.907501897311307
Precision: 0.8986583064931105
Recall: 0.915148809812617
F1Score: 0.9068285951931075
================================================================
Complete: 21:59:41
```

ภาพที่ 4-37 ผลการทดสอบด้วยวิธี LbfgsLogisticRegression ครั้งที่ 2

4.13 วิธี Least Squares Support Vector Machine (LdSvm) ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.13.1 ใช้เวลา 9 นาที 2 วินาที

4.13.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 93,655 ครั้ง

4.13.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 9,396 ครั้ง

4.13.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 8,173 ครั้ง

4.13.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 98,283 ครั้ง

4.13.6 ความแม่นยำ (Accuracy) อยู่ที่ (93,655 + 98,283) / (93,655 + 9,396 + 8,173 + 98,283) = 0.9161

4.13.7 Recall อยู่ที่ 93,655 / (93,655 + 8,173) = 0.9197

4.13.8 Precision อยู่ที่ 93,655 / (93,655 + 9,396) = 0.9088

4.13.9 F1 Score = 2 x (0.9197 + 0.9088) / (0.9197 x 0.9088) = 0.9142

จากภาพที่ 4-38 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LdSvm ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่ใช้เวลาในการสอบแบบจำลองมากที่สุด และมีคะแนน Recall มากที่สุดเมื่อเทียบกับวิธีอื่น ๆ

```
Start: 22:08:47
================================================================
LdSvm
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   93,655 |    9,396 | 0.9088
 negative ||    8,173 |   98,283 | 0.9232
          ||======================
Precision ||   0.9197 |   0.9127 |

Accuracy: 0.91614122678478 53
Precision: 0.9197372039124798
Recall: 0.9088218454939787
F1Score: 0.914246945758228
================================================================
Complete: 22:17:49
```

ภาพที่ 4-38 ผลการทดสอบด้วยวิธี LdSvm ครั้งที่ 2

4.14 วิธี Linear Support Vector Machine (LinearSvm) ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.14.1 ใช้เวลา 10 วินาที

4.14.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 84,141 ครั้ง

4.14.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 18,910 ครั้ง

4.14.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 10,297 ครั้ง

4.14.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 96,159 ครั้ง

4.14.6 ความแม่นยำ (Accuracy) อยู่ที่ (84,141 + 96,159) / (84,141 + 18,910 + 10,297 + 96,159) = 0.8606

4.14.7 Recall อยู่ที่ 84,141 / (84,141 + 10,297) = 0.891

4.14.8 Precision อยู่ที่ 84,141 / (84,141 + 18,910) = 0.8165

4.14.9 F1 Score = 2 x (0.891 + 0.8165) / (0.891 x 0.8165) = 0.8521

จากภาพที่ 4-39 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี LinearSvm ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่ใช้เวลาในการสอนแบบจำลองน้อยที่สุด มีการทำนายถูกน้อยที่สุด และมีคะแนน F1 Score น้อยที่สุดเมื่อเทียบกับวิธีอื่น ๆ

```
Start: 22:19:16
================================================================
LinearSvm
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||=====================
PREDICTED || positive | negative | Recall
TRUTH     ||=====================
 positive ||   84,141 |   18,910 | 0.8165
 negative ||   10,297 |   96,159 | 0.9033
          ||=====================
Precision ||   0.8910 |   0.8357 |

Accuracy: 0.8605917702033822
Precision: 0.8909655011753743
Recall: 0.81649862689348
F1Score: 0.8521082186855977
================================================================
Complete: 22:19:26
```

ภาพที่ 4-39 ผลการทดสอบด้วยวิธี LinearSvm ครั้งที่ 2

4.15 วิธี Stochastic Dual Coordinate Ascent Logistic Regression (SdcaLogisticRegression) ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.15.1 ใช้เวลา 1 นาที 6 วินาที

4.15.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 94,284 ครั้ง

4.15.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 8,767 ครั้ง

4.15.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 11,277 ครั้ง

4.15.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 95,179 ครั้ง

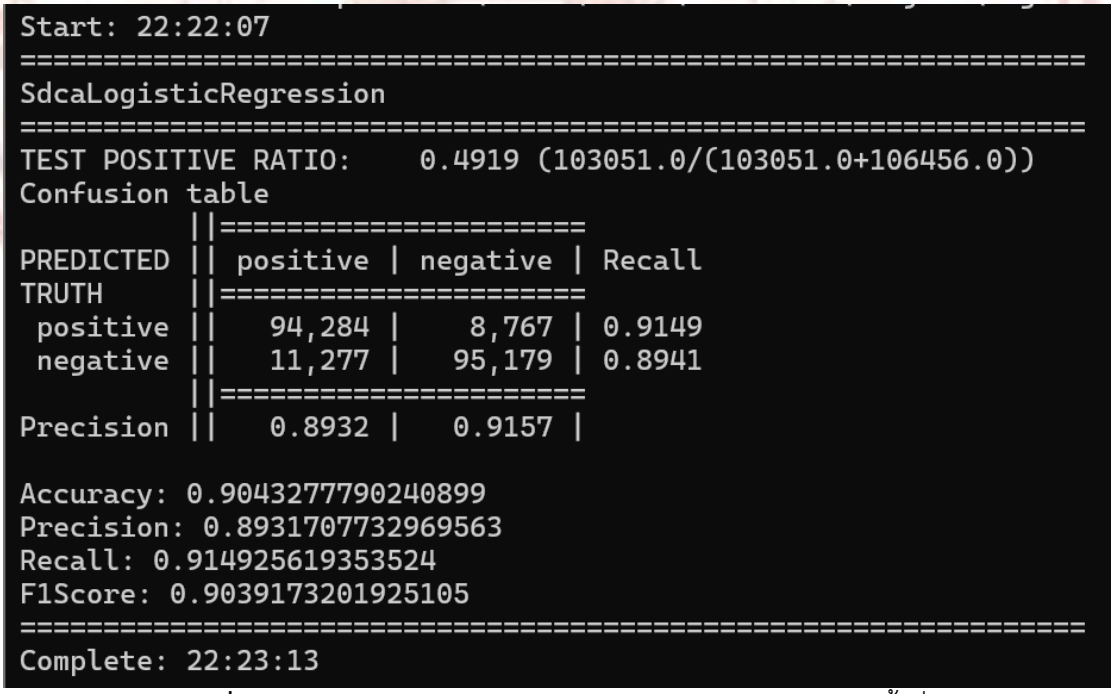
4.15.6 ความแม่นยำ (Accuracy) อยู่ที่ (94,284 + 95,179) / (94,284 + 8,767 + 11,277 + 95,179) = 0.9043

4.15.7 Recall อยู่ที่ 94,284 / (94,284 + 11,277) = 0.8932

4.15.8 Precision อยู่ที่ 94,284 / (94,284 + 8,767) = 0.9149

4.15.9 F1 Score = 2 x (0.8932 + 0.9149) / (0.8932 x 0.9149) = 0.9039

จากภาพที่ 4-40 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SdcaLogistic Regression ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากเป็นลำดับที่ 2 จำนวน 105,561 ครั้ง และมีค่า Precision มากเป็นลำดับที่ 3



```
Start: 22:22:07
================================================================
SdcaLogisticRegression
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   94,284 |    8,767 | 0.9149
 negative ||   11,277 |   95,179 | 0.8941
          ||======================
Precision ||   0.8932 |   0.9157 |

Accuracy: 0.9043277790240899
Precision: 0.8931707732969563
Recall: 0.914925619353524
F1Score: 0.9039173201925105
================================================================
Complete: 22:23:13
```

ภาพที่ 4-40 ผลการทดสอบด้วยวิธี SdcaLogisticRegression ครั้งที่ 2

4.16 วิธี Stochastic Dual Coordinate Ascent Logistic Regression Non Calibrated (SdcaNonCalibrated) ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.16.1 ใช้เวลา 1 นาที 33 วินาที

4.16.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 94,124 ครั้ง

4.16.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 8,927 ครั้ง

4.16.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 10,464 ครั้ง

4.16.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 95,992 ครั้ง

4.16.6 ความแม่นยำ (Accuracy) อยู่ที่ (94,124 + 95,992) / (94,124 + 8,927 + 10,464 + 95,992) = 0.9074

4.16.7 Recall อยู่ที่ 94,124 / (94,124 + 10,464) = 0.9

4.16.8 Precision อยู่ที่ 94,124 / (94,124 + 8,927) = 0.9134

4.16.9 F1 Score = 2 x (0.9 + 0.9134) / (0.9 x 0.9134) = 0.9066

จากภาพที่ 4-41 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SdcaNon Calibrated ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่มีคะแนน Recall มากเป็นลำดับที่ 3

```
Start: 22:28:26
================================================================
SdcaNonCalibrated
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   94,124 |    8,927 | 0.9134
 negative ||   10,464 |   95,992 | 0.9017
          ||======================
Precision ||   0.9000 |   0.9149 |

Accuracy: 0.907444619988831
Precision: 0.8999502811029946
Recall: 0.9133729900728765
F1Score: 0.9066119563280501
================================================================
Complete: 22:29:59
```

ภาพที่ 4-41 ผลการทดสอบด้วยวิธี SdcaNonCalibrated ครั้งที่ 2

4.17 วิธี SgdCalibrated ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.17.1 ใช้เวลา 3 นาที 15 วินาที

4.17.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 93,524 ครั้ง

4.17.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 9,527 ครั้ง

4.17.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 10,796 ครั้ง

4.17.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 95,660 ครั้ง

4.17.6 ความแม่นยำ (Accuracy) อยู่ที่ (93,524 + 95,660) / (93,524 + 9,527 + 10,796 + 95,660) = 0.903

4.17.7 Recall อยู่ที่ 93,524 / (93,524 + 10,796) = 0.8965

4.17.8 Precision อยู่ที่ 93,524 / (93,524 + 9,527) = 0.9076

4.17.9 F1 Score = 2 x (0.8965 + 0.9076) / (0.8965 x 0.9076) = 0.902

จากภาพที่ 4-42 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SgdCalibrated ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่ใช้เวลาสอนแบบจำลองมากเป็นลำดับที่ 2 มีการทำนายถูกน้อยเป็นลำดับที่ 3 และมีคะแนน F1 Score ต่ำเป็นลำดับที่ 2

```
Start: 22:31:03
================================================================
SgdCalibrated
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED ||  positive |  negative | Recall
TRUTH     ||======================
 positive ||    93,524 |     9,527 | 0.9076
 negative ||    10,796 |    95,660 | 0.8986
          ||======================
Precision ||    0.8965 |    0.9094 |

Accuracy: 0.9029960812765206
Precision: 0.896510736196319
Recall: 0.9075506302704486
F1Score: 0.901996904099416
================================================================
Complete: 22:34:18
```

ภาพที่ 4-42 ผลการทดสอบด้วยวิธี SgdCalibrated ครั้งที่ 2

4.18 วิธี SgdNonCalibrated ครั้งที่ 2 ได้ผลลัพธ์ ดังนี้

4.18.1 ใช้เวลา 2 นาที 12 วินาที

4.18.2 ทำนายว่าน่าสงสัยแล้วถูก (True Positive : TP) จำนวน 93,914 ครั้ง

4.18.3 ทำนายน่าสงสัยแล้วผิด (False Positive : FP) จำนวน 9,137 ครั้ง

4.18.4 ทำนายปกติแล้วผิด (False Negative : FN) จำนวน 11,218 ครั้ง

4.18.5 ทำนายปกติแล้วถูก (True Negative : TN) จำนวน 95,238 ครั้ง

4.18.6 ความแม่นยำ (Accuracy) อยู่ที่ (93,914 + 95,238) / (93,914 + 9,137 + 11,218 + 95,238) = 0.9028

4.18.7 Recall อยู่ที่ 93,914 / (93,914 + 11,218) = 0.8933

4.18.8 Precision อยู่ที่ 93,914 / (93,914 + 9,137) = 0.9113

4.18.9 F1 Score = 2 x (0.8933 + 0.9113) / (0.8933 x 0.9113) = 0.9022

จากภาพที่ 4-43 จะแสดงเวลาเริ่มต้น-สิ้นสุด และค่าต่าง ๆ ที่ได้จากการจำแนกด้วยวิธี SgdNon Calibrated ในการทดลอง ครั้งที่ 2 จะสังเกตได้ว่าเป็นวิธีที่มีการทำนายว่าเป็นการเข้าสู่ระบบที่ผิดปกติมากเป็นลำดับที่ 3 ที่สุดจำนวน 105,132 ครั้ง มีการทำนายถูกน้อยเป็นลำดับที่ 2 และมีคะแนน F1 Score ต่ำเป็นลำดับที่ 3

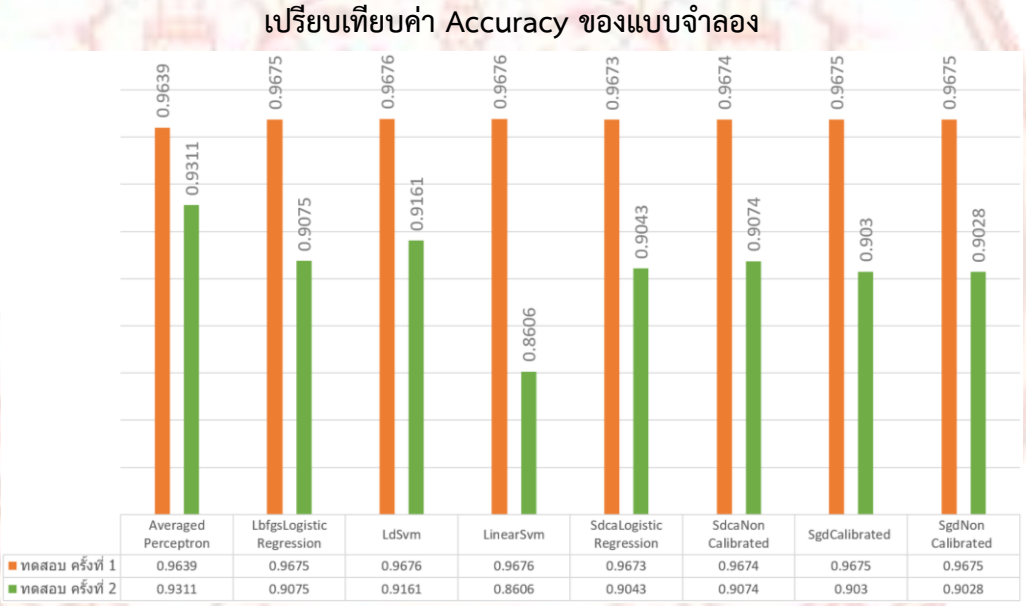
```
Start: 22:35:18
================================================================
SgdNonCalibrated
================================================================
TEST POSITIVE RATIO:    0.4919 (103051.0/(103051.0+106456.0))
Confusion table
          ||======================
PREDICTED || positive | negative | Recall
TRUTH     ||======================
 positive ||   93,914 |    9,137 | 0.9113
 negative ||   11,218 |   95,238 | 0.8946
          ||======================
Precision ||   0.8933 |   0.9125 |

Accuracy: 0.90284334174991 76
Precision: 0.8932960468744056
Recall: 0.9113351641420268
F1Score: 0.9022254458817482
================================================================
Complete: 22:37:31
```

ภาพที่ 4-43 ผลการทดสอบด้วยวิธี SgdNonCalibrated ครั้งที่ 2

4.19 การเปรียบเทียบระหว่างการทดลองครั้งที่ 1 และครั้งที่ 2

จากการทดลอง จะเห็นว่าการทำ Binary Classification ด้วยวิธีที่แตกต่างกัน ให้ผลลัพธ์ไม่เหมือนกัน ผู้วิจัยได้นำผลลัพธ์ของแต่ละวิธีมาคำนวณ Recall และ Precision เพื่อไปคำนวณหา F1 Score โดยพบว่าจะพบว่าวิธี AveragedPerceptron มีค่า F1 Score สูงที่สุด และตามมาด้วยวิธี LdSvm, LbfgsLogisticRegression, SdcaNonCalibrated, SdcaLogisticRegression, SgdNonCalibrated, SgdCalibrated, LinearSvm ตามลำดับ

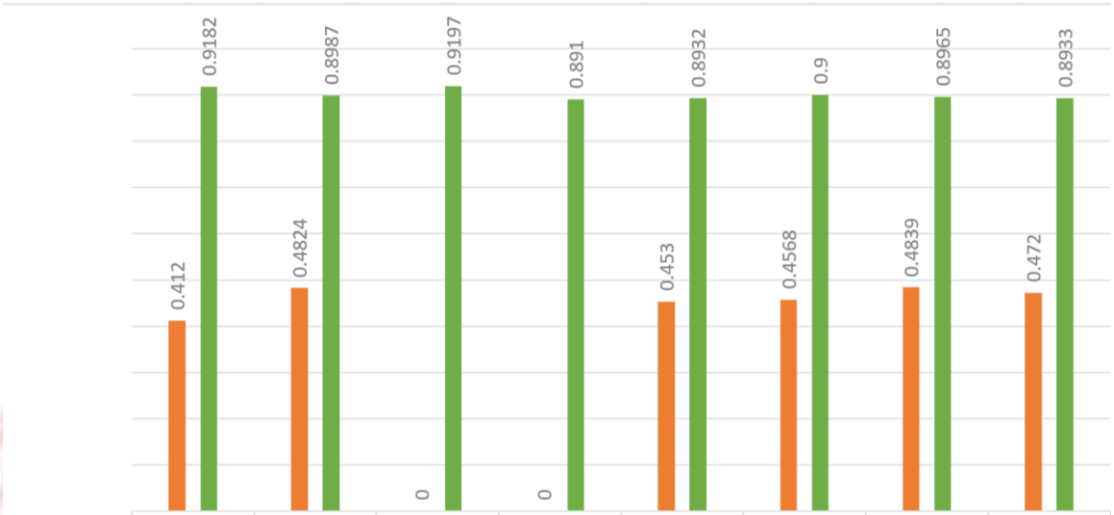
4.19.1 ผู้วิจัยได้นำค่าคะแนน Accuracy, Recall, Precision และ F1 Score มาทำการเปรียบเทียบ โดยได้สรุปผลในรูปแบบกราฟ ดังนี้



ภาพที่ 4-44 กราฟเปรียบเทียบค่า Accuracy ของแบบจำลอง

4.19.1.1 จากภาพที่ 4-44 กราฟเปรียบเทียบค่า Accuracy จะเห็นว่าในการทดสอบครั้งที่ 1 ค่าที่ได้อยู่ในช่วง 0.9639 ถึง 0.9676 และ ครั้งที่ 2 ค่าที่ได้อยู่ในช่วง 0.8606 ถึง 0.9311 ซึ่งจากการทดสอบทั้ง 2 ครั้ง ค่า Accuracy มีคะแนนในระดับดี โดยการทดสอบครั้งที่ 1 ค่าที่ได้มีความใกล้เคียงกัน ซึ่งเกิดจากชุดข้อมูลที่ได้มามีข้อมูล Record การเข้าสู่ระบบที่ผิดปกติในอัตราส่วนที่น้อยเกินไป เมื่อนำชุดข้อมูลมา สอน จะทำให้แบบจำลอง ทำนายว่า Record ส่วนใหญ่เป็นการใช้งานปกติ ซึ่งข้อมูลที่ใช้ทดสอบก็มาจากการแบ่งชุดข้อมูลชุดเดียวกัน จึงมีความแม่นยำสูง ต่างจากในการทดลองครั้งที่ 2 ซึ่งข้อมูลที่ใช้ทดสอบทั้งปกติและผิดปกติ มีอัตราส่วนที่ใกล้เคียงกัน โมเดลจึงทำนายผลได้ทั้งปกติและผิดปกติ

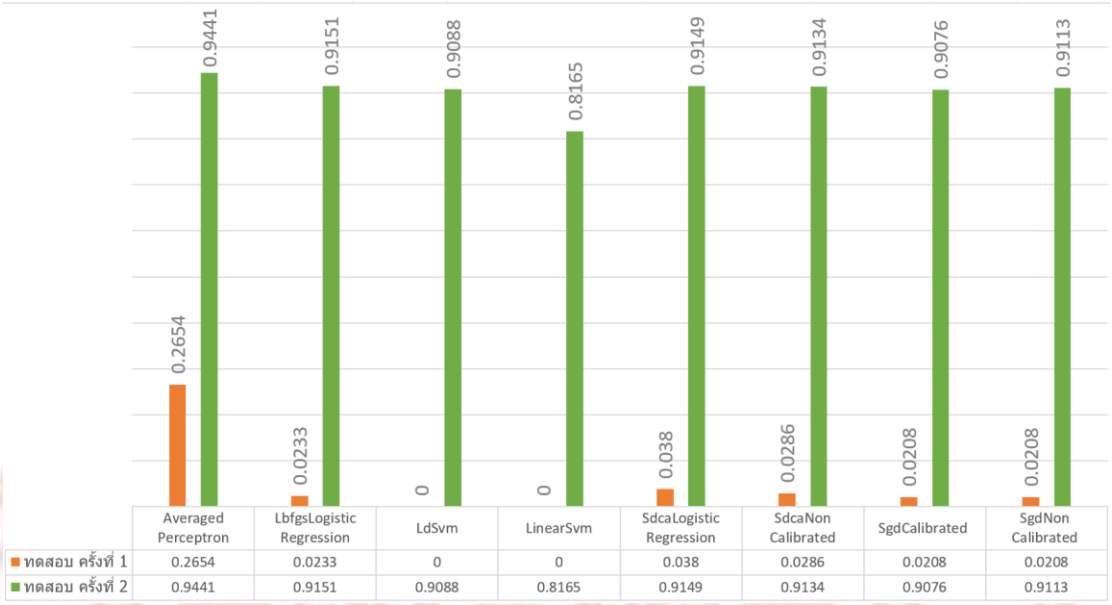
เปรียบเทียบค่า Recall ของแบบจำลอง



	Averaged Perceptron	LbfgsLogistic Regression	LdSvm	LinearSvm	SdcaLogistic Regression	SdcaNon Calibrated	SgdCalibrated	SgdNon Calibrated
ทดสอบ ครั้งที่ 1	0.412	0.4824	0	0	0.453	0.4568	0.4839	0.472
ทดสอบ ครั้งที่ 2	0.9182	0.8987	0.9197	0.891	0.8932	0.9	0.8965	0.8933

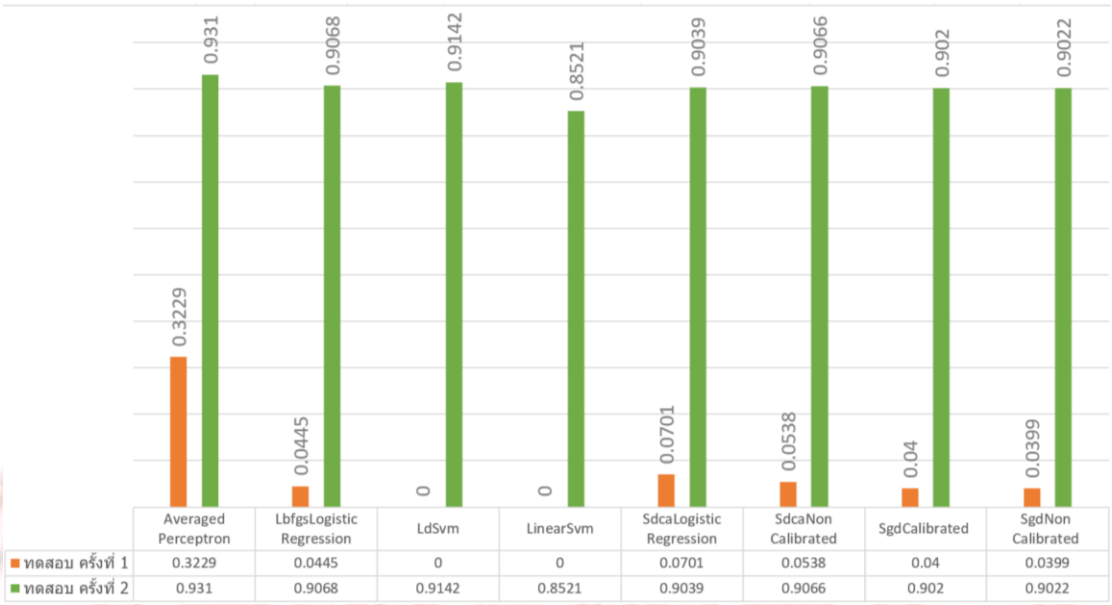
ภาพที่ 4-45 กราฟเปรียบเทียบค่า Recall ของแบบจำลอง

4.19.1.2 จากภาพที่ 4-45 กราฟเปรียบเทียบค่า Recall จะเห็นว่าในการทดสอบครั้งที่ 1 ค่าที่ได้อยู่ในช่วง 0 ถึง 0.4839 และ ครั้งที่ 2 ค่าที่ได้อยู่ในช่วง 0.8932 ถึง 0.9197 ซึ่งจากการทดสอบทั้ง 2 ครั้ง ค่า Recall ในการทดสอบ ครั้งที่ 1 มีคะแนนต่ำกว่า 50 และการทดสอบด้วยวิธี LdSvm และ LinearSvm ได้คะแนน 0 และจะเห็นได้ว่าการทดสอบครั้งที่ 2 คะแนนที่ได้มีค่าสูงกว่า ครั้งที่ 1 ทุกแบบจำลอง โดยค่า Recall ที่สูงหมายถึง ระบบสามารถตรวจจับ Record ผิดปกติได้มาก



ภาพที่ 4-46 กราฟเปรียบเทียบค่า Precision ของแบบจำลอง

4.19.1.3 จากภาพที่ 4-46 กราฟเปรียบเทียบค่า Precision จะเห็นว่าในการทดสอบครั้งที่ 1 ค่าที่ได้อยู่ในช่วง 0 ถึง 0.2654 และ ครั้งที่ 2 ค่าที่ได้อยู่ในช่วง 0.8165 ถึง 0.9441 ซึ่งจากการทดสอบทั้ง 2 ครั้ง ค่า Precision ในการทดสอบ ครั้งที่ 1 มีคะแนนใกล้เคียง 0 เมื่อเปรียบกับครั้งที่ 2 และจะเห็นได้ว่าการทดสอบครั้งที่ 2 คะแนนที่ได้มีค่าสูงกว่า ครั้งที่ 1 ทุกรายการอย่างเห็นได้ชัด โดยค่า Precision ที่สูงหมายถึง ระบบสามารถการแยกแยะ Record ผิดปกติได้ความถูกต้องมากน้อยแค่ไหน



ภาพที่ 4-47 กราฟเปรียบเทียบค่า F1 Score ของแบบจำลอง

4.19.1.4 จากภาพที่ 4-47 กราฟเปรียบเทียบค่า F1 Score จะเห็นว่าในการทดสอบครั้งที่ 1 ค่าที่ได้อยู่ในช่วง 0 ถึง 0.3229 และ ครั้งที่ 2 ค่าที่ได้อยู่ในช่วง 0.8521 ถึง 0.931 จากการทดสอบพบว่าทดสอบครั้งที่ 2 คะแนนที่ได้มีค่าสูงกว่า ครั้งที่ 1 อย่างเห็นได้ชัด โดยค่า F1 Score ที่สูงหมายถึง ระบบมีประสิทธิภาพสูง

4.19.1.5 จึงสรุปได้ว่าจากการทดสอบครั้งที่ 1 ชุดข้อมูลที่ได้จาก Dataset ที่เผยแพร่บนอินเทอร์เน็ตเมื่อนำมา สอนแบบจำลอง จะมีค่า Accuracy สูงกว่าครั้งที่ 2 เนื่องจากข้อมูลส่วนมากเป็น Record ที่ปกติระบบจึงไม่มีข้อมูลเพียงพอในการทำนาย Record ที่ผิดปกติ ประกอบกับข้อมูลที่ใช้ Test มีปริมาณ Record ปกติมากจึงมีคะแนนที่สูง ซึ่งเมื่อดูที่ค่า Recall, Precision และ F1 Score จะเห็นว่าการทดสอบครั้งที่ 2 ด้วยชุดข้อมูลที่ได้ปรับปรุงให้อัตราส่วนระหว่าง Record ปกติ และ ผิดปกติ มีปริมาณที่ใกล้เคียงกัน มีคะแนนที่ดีกว่าบ่งบอกว่าข้อมูลที่ใช้ในการทดสอบครั้งที่ 2 มีคุณภาพมากกว่าที่ใช้ในการทดสอบครั้งที่ 1 และเมื่อพิจารณาค่า F1 Score จะพบว่า AveragedPerceptron มีคะแนนสูงสุดอยู่ที่ 0.931 ซึ่งมีความแม่นยำเพียงพอต่อการนำไปใช้งานจริง ผู้วิจัยจึงเลือกวิธี AveragedPerceptron ในการใช้งานวิเคราะห์พฤติกรรมของการเข้าสู่ระบบที่ผิดปกติ

บทที่ 5
สรุปผลการวิจัย ความท้าทาย และข้อเสนอแนะ

จากการผลการพัฒนาระบบการตรวจสอบสิทธิ์แบบสองปัจจัยและการประเมินประสิทธิภาพของแบบจำลองการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ สามารถแบ่งเป็นประเด็นในด้านต่าง ๆ ได้ดังนี้

5.1 สรุปผลการวิจัย

การพัฒนาระบบตรวจสอบพฤติกรรมการเข้าใช้งานที่ผิดปกติ พบว่าการใช้ Binary Classification สามารถจำแนกการเข้าสู่ที่ผิดปกติได้ โดยจำเป็นต้องมีชุดข้อมูล log การเข้าสู่ระบบที่เพียงพอ ซึ่งจากการทดสอบด้วย Classification ทั้ง 8 วิธีพบว่า AveragedPerceptron มีความเหมาะสมกับที่สุด ทั้งนี้ผลของการทดสอบอาจแตกต่างกันตามชุดข้อมูลที่ใช้งาน จึงสรุปได้ว่าการเพิ่มความปลอดภัยโดยใช้การยืนยันตัวตน 2 ปัจจัย ที่ไม่กระทบประสบการณ์ของผู้ใช้งานสามารถทำได้จริง ซึ่งการนำไปใช้งานควรเริ่มจากประยุกต์ใช้ในระบบที่ไม่ใช่บริการหลักของหน่วยงานก่อน เนื่องจากระบบที่พัฒนาขึ้น เน้นไปที่การเพิ่มความปลอดภัยให้ระบบและความสะดวกในการใช้งาน จึงมีความความปลอดภัยที่น้อยกว่าการบังคับใช้การยืนยันตัวตน 2 ปัจจัยในทุกครั้งที่เข้าสู่ระบบ

ทั้งนี้การพัฒนาระบบในการวิจัยนี้เป็นเพียงแนวทางเบื้องต้นมีความจำเป็นต้องปรับปรุงเพิ่มเติมให้เข้ากับบริบทของแต่ละหน่วยงาน เนื่องจากหน่วยงานมีการใช้เทคโนโลยีในการพัฒนาระบบและมีภารกิจการให้บริการที่ต่างกัน ผู้วิจัยจึงเลือกใช้การพัฒนาระบบในรูปแบบ API ซึ่งสามารถใช้งานจากหลายแพลตฟอร์ม เพื่อให้สามารถเพิ่มความปลอดภัยของระบบ โดยไม่กระทบต่อประสบการณ์ของผู้ใช้งานต่อไป

5.2 ความท้าทาย

5.2.1 เทคโนโลยีและภัยคุกคามมีการเปลี่ยนแปลงที่รวดเร็วจึงควรพิจารณาปรับเปลี่ยนความเหมาะสม

5.2.2 การนำระบบตรวจสอบพฤติกรรมการเข้าใช้งานที่ผิดปกติไปปรับใช้กับระบบของหน่วยงานจำเป็นต้องปรับปรุง ระบบเดิมของหน่วยงาน ในขั้นตอนการเข้าสู่ระบบ ซึ่งในระบบที่ไม่มีผู้ให้บริการบำรุงรักษาแล้ว อาจมีความลำบากในการนำไปใช้งาน

5.2.3 API สำหรับตรวจสอบ IP Address ที่ผิดปกติมีข้อจำกัดเรื่องปริมาณการส่ง หากนำไปประยุกต์ใช้ในระบบที่มีการเข้าใช้งานจำนวนมาก อาจส่งผลให้ระบบไม่สามารถตรวจสอบ IP Address เมื่อมีปริมาณการส่งเกินที่ผู้ให้บริการกำหนด

5.3 ข้อเสนอแนะ

5.3.1 การ สอนแบบจำลอง หากหน่วยงานยังมีข้อมูล log การเข้าสู่ระบบไม่เพียงพอ อาจพิจารณาใช้ข้อมูลจากแหล่งอื่นหรือ Dataset ที่เผยแพร่บนอินเทอร์เน็ตก่อน ซึ่งมีจะข้อจำกัดเรื่องข้อมูลซึ่งไม่ตรงกับพฤติกรรมการการใช้งานจริงของระบบ และเริ่มจัดเก็บข้อมูล log การเข้าสู่ระบบเมื่อใช้งานจนมีข้อมูล log การเข้าสู่ระบบมากพอ จึงนำข้อมูลที่ได้มา สอนแบบจำลอง ซึ่งเป็นข้อมูลที่เกิดจากพฤติกรรมการใช้งานของระบบจริง ซึ่งจะช่วยให้ระบบสามารถตรวจจับพฤติกรรมได้แม่นยำมากยิ่งขึ้น

5.3.2 หากมีการใช้งานร่วมกับหลายระบบซึ่งมีกลุ่มผู้ใช้งานที่แตกต่างกัน พิจารณาแยกการจัดเก็บ log การเข้าใช้งานไว้ที่ Table แยกจากกันเพื่อนำข้อมูลไป สอนแบบจำลอง ให้ตรงตามพฤติกรรมการใช้งานจริงมากที่สุด

5.3.3 การยืนยันตัวตนอาจพิจารณาเพิ่มช่องทางในการส่ง เช่น SMS , Line Offical หรือช่องทางอื่น ๆ ตามความสะดวก หรืออาจพิจารณาเปลี่ยนวิธีการยืนยันตัวตนโดยใช้ Third Party ตามความสำคัญ ของแต่ละระบบงาน ซึ่งอาจต้องคำนึงถึงค่าใช้จ่ายในการใช้งานด้วย

5.3.4 การแยกข้อมูลจาก User Agent Header ผู้วิจัยดำเนินการเพียงเบื้องต้น ซึ่งยังมีข้อมูลอื่น ๆ ที่สามารถจัดเก็บได้ สามารถใช้งาน API UserAgentParser ซึงเพื่อได้รับข้อมูลที่ละเอียดมากขึ้นได้

บรรณานุกรม

[1] ทิภาพร, สุวิทย์, ชนะธิป, และ อิฐติศักดิ์, "การวิเคราะห์แนวทางการพัฒนาการตรวจสอบและแจ้งเตือน การรักษาความมั่นคงปลอดภัยของระบบปฏิบัติการด้วยการประยุกต์ใช้โมเดลออโตมาตา", คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษมบัณฑิต

[2] วรกฤษฎ์, "ปัจจัยที่อิทธิพลต่อความพึงพอใจในการใช้งานการยืนยันตัวตน แบบหลายปัจจัยในโมไบล์แบงก์กิ้งแอปพลิเคชัน", คณะพาณิชยศาสตร์และการบัญชี มหาวิทยาลัยธรรมศาสตร์

[3] วุฒิพงษ์, และ ชัยนันท์, "การยืนยันตัวตนสำหรับเว็บแอปพลิเคชันแบบพหุปัจจัย", คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏสกลนคร

[4] AbuseIPDB, "Introduction The AbuseIPDB"
Available: https://docs.abuseipdb.com/#introduction

[5] IsMalicious, "API Reference" ,Available: https://docs.ismalicious.com/technical-docs/api-reference

[6] Mathews, "Multifactor Authentication Using Zero Trust ",Department of Electrical Engineering & Computing Rochester Institute of Technology Dubai Campus

[7] Microsoft, "ML.NET documentation"
Available: https://learn.microsoft.com/en-us/dotnet/machine-learning

[8] OWASP, "OWASP Top 10:2021", Available: https://owasp.org/Top10

[9] PyTorch, "PyTorch documentation"
Available: https://docs.pytorch.org/docs/stable/index.html

[10] VirusTotal, "VirusTotal API v3 Overview",
Available: https://docs.virustotal.com/reference/ip-info

[11] World Economic Forum, "The Globla Risks Report 2024 19th Edition"
Available: https://www.weforum.org/publications/global-risks-report-2024/digest

ประวัติผู้เขียน

ชื่อ นายณัฐวีร์ พูลสมบัติภิญโญ

ชื่อการค้นคว้าอิสระ การตรวจสอบสิทธิ์แบบสองปัจจัยพร้อมการกรองพฤติกรรมการเข้าสู่ระบบที่ผิดปกติ

สาขาวิชา การบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ

ประวัติ สำเร็จการศึกษาระดับปริญญาตรี วิทยาศาสตรบัณฑิต คณะศิลปศาสตร์และวิทยาศาสตร์ สาขาวิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์ ปีการศึกษา 2562

ประสบการณ์ทำงาน ปฏิบัติงานในตำแหน่ง นักวิชาการคอมพิวเตอร์ปฏิบัติการ กองยุทธศาสตร์และแผนงาน หน่วยงานด้านนโยบายระดับกรมแห่งหนึ่ง พ.ศ. 2565 – ปัจจุบัน

ปฏิบัติงานในตำแหน่ง Programmer Analyst บริษัท ไมโม่เทค จำกัด พ.ศ. 2562 – 2565