

การศึกษาวิจัยเพื่อพัฒนาสร้างชุดข้อมูลในการฝึกสอนไฟร์วอลล์  
ปัญญาประดิษฐ์ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร์วอลล์

Researching for developing training sets  
with artificial neural network technology based on firewall rules

โดย

ฐิติโชติ ใจเมือง

Thitichote Chaimuang

พิพัฒน์บุญ พุทธคุณ

Pipatboon Buddhakul

อาจารย์ที่ปรึกษา

ผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 1 ปีการศึกษา 2563

การศึกษาวิจัยเพื่อพัฒนาสร้างชุดข้อมูลในการฝึกสอนไฟร์วอลล์

ปัญญาประดิษฐ์

ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร์วอลล์

Researching for developing training sets

with artificial neural network technology based on firewall rules

โดย

ฐิติโชติ ใจเมือง

พิพัฒน์บุญ พุทธคุณ

อาจารย์ที่ปรึกษา

ผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 2 ปีการศึกษา 2562

**RESEARCHING FOR DEVELOPING TRAINING SETS  
WITH ARTIFICIAL NEURAL NETWORK TECHNOLOGY  
BASED ON FIREWALL RULES**

**THITICHOTE CHAIMUANG  
PIPATBOON BUDDHAKUL**

**A PROJECT SUBMITTED IN PARTIAL FULFILLMENT  
OF THE REQUIREMENT FOR THE DEGREE OF  
BACHELOR OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY  
FACULTY OF INFORMATION TECHNOLOGY  
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG**

**2/2019**

**COPYRIGHT 2020**

**FACULTY OF INFORMATION TECHNOLOGY**

**KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG**

ใบรับรองปริญญาโท ประจำปีการศึกษา 2562

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การศึกษาวิจัยเพื่อพัฒนาสร้างชุดข้อมูลในการฝึกสอนไฟร์วอลล์  
ปัญญาประดิษฐ์ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร์วอลล์

RESEARCHING FOR DEVELOPING TRAINING SET WITH  
ARTIFICIAL NEURAL NETWORK TECHNOLOGY BASED ON  
FIREWALL RULES

ผู้จัดทำ

- |    |                       |                       |
|----|-----------------------|-----------------------|
| 1. | นายฐิติโชติ ใจเมือง   | รหัสประจำตัว 60070019 |
| 2. | นายพิพัฒน์บุญ พุทธคุณ | รหัสประจำตัว 60070065 |

..... อาจารย์ที่ปรึกษา  
(ผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ)

## ใบรับรองโครงการ (Project)

### เรื่อง

การศึกษาวิจัยเพื่อพัฒนาสร้างชุดข้อมูลในการฝึกสอนไฟร์วอลล์

ปัญญาประดิษฐ์

ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร์วอลล์

Researching for developing training sets

with artificial neural network technology based on firewall rules

นายฐิติโชติ ใจเมือง รหัสประจำตัว 60070019

นายพิพัฒน์บุญ พุทธคุณ รหัสประจำตัว 60070065

ขอรับรองว่ารายงานฉบับนี้ ข้าพเจ้าไม่ได้คัดลอกมาจากที่ใด  
รายงานฉบับนี้ได้รับการตรวจสอบและอนุมัติให้เป็นส่วนหนึ่งของ  
การศึกษาวิชาโครงการ หลักสูตรวิทยาศาสตรบัณฑิต (เทคโนโลยีสารสนเทศ)  
ภาคเรียนที่ 2 ปีการศึกษา 2562

.....  
(นายฐิติโชติ ใจเมือง)

.....  
(นายพิพัฒน์บุญ พุทธคุณ)

หัวข้อโครงการ	การศึกษาวิจัยเพื่อพัฒนาสร้างชุดข้อมูลในการฝึกสอนไฟร์วอลล์ปัญญาประดิษฐ์ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร์วอลล์		
นักศึกษา	ฐิติโชติ	ใจเมือง	รหัสนักศึกษา 60070019
	พิพัฒน์บุญ	พุทธรคุณ	รหัสนักศึกษา 60070065
ปริญญา	วิทยาศาสตร์บัณฑิต		
สาขาวิชา	เทคโนโลยีสารสนเทศ		
ปีการศึกษา	2563		
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ		

## บทคัดย่อ

ในโครงการวิจัยนี้เป็นการศึกษาการพัฒนาชุดข้อมูลฝึกสอนที่ออกแบบจากกฎของไฟร์วอลล์มีเป้าหมายเพื่อให้ชุดข้อมูลฝึกสอนที่ออกแบบขึ้นสามารถฝึกสอนโมเดลโครงข่ายประสาทเทียมเชิงลึกได้อย่างมีประสิทธิภาพมากที่สุดซึ่งปัจจัยสำคัญที่เรานำมาตัดสินใจในการพิจารณาเลือกใช้ชุดข้อมูลฝึกสอนที่สร้างขึ้น จะประกอบไปด้วยจำนวนชุดข้อมูลฝึกสอนเวลาที่ใช้ประมวลผล ความแม่นยำในการทำนายผล และวิธีการแบ่งจำนวนชุดข้อมูลฝึกสอนในแต่ละกฎไฟร์วอลล์ โดยเราได้ตั้งสมมติฐานและลงมือสร้างชุดข้อมูลฝึกสอนด้วยเงื่อนไขที่แตกต่างกันนำไปทดสอบและสรุปผลมุ่งเน้นไปที่การหาความสัมพันธ์ของตัวแปรต่างๆ เปรียบเทียบเป็นกราฟและเลือกจุดที่เหมาะสมที่สุดในการเลือกชุดข้อมูลฝึกสอนมาใช้ในโมเดล

จากการวิเคราะห์ผลลัพธ์ที่ได้จากการทดสอบสรุปได้ว่า การใช้โมเดลที่มีการแบ่งด้วยจำนวนชุดข้อมูลให้เท่าๆกันในแต่ละกฎไฟร์วอลล์มีประสิทธิภาพที่ดีกว่าการแบ่งจำนวนชุดข้อมูลฝึกสอนด้วยอัตราส่วนที่เท่ากัน เนื่องจากการใช้อัตราส่วนการเพิ่มจำนวนชุดข้อมูลฝึกสอนจะติดปัญหาเมื่อจำนวนความเป็นไปได้ของข้อมูลในเงื่อนไขต่างกันมากเกินไป จนทำให้โมเดลหาความสัมพันธ์ของข้อมูลได้ยาก

<b>Project Title</b>	Researching for developing training set with artificial neural network technology based on firewall rules		
<b>Student</b>	Thitichote	Chaimuang	Student ID 60070019
	Pipatboon	Buddhakul	Student ID 60070065
<b>Degree</b>	วิทยาศาสตรบัณฑิต		
<b>Program</b>	เทคโนโลยีสารสนเทศ		
<b>Academic Year</b>	2020		
<b>Advisor</b>	ผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ		

## ABSTRACT



## กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ทางผู้จัดทำขอขอบพระคุณเป็นอย่างสูงกับความกรุณาช่วยเหลือและการให้คำปรึกษาของ ผู้ช่วยศาสตราจารย์อัครินทร์ คุณกิตติ ที่ช่วยชี้แนะแนวทาง ตั้งแต่วันแรกถึงวันสุดท้าย และขอบพระคุณอาจารย์ คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่าน ที่ให้ความรู้อันเป็นประโยชน์ยิ่ง ต่อการพัฒนาต่อ ยอดองค์ความรู้

ขอขอบคุณครอบครัวที่ให้การสนับสนุนอย่างดีเสมอมา

ขอขอบคุณผู้ร่วมงานที่อดทนและร่วมแรงร่วมใจช่วยกันมาจนถึงทุกวันนี้

จิตติ ใจเมือง

พิพัฒน์บุญ พุทธคุณ

# สารบัญ

หน้า

บทคัดย่อภาษาไทย .....	I
บทคัดย่อภาษาอังกฤษ .....	II
กิตติกรรมประกาศ.....	III
สารบัญ .....	IV
สารบัญตาราง .....	VI
สารบัญรูป .....	VII

## บทที่

1. บทนำ.....	
1.1 ความเป็นมาของโครงการ.....	
1.2 วัตถุประสงค์.....	
1.3 วิธีการดำเนินงาน.....	
1.4 ขอบเขตของโครงการ.....	
1.5 ประโยชน์ที่คาดว่าจะได้รับ.....	
2. ทฤษฎีการนำโครงข่ายระบบประสาทเชิงลึกมาใช้ในการทำงานของไฟร่วอลต์.....	
2.1 เทคโนโลยีของไฟร่วอลต์และโครงข่ายระบบประสาทเชิงลึก.....	
2.2 ทบทวนวรรณกรรม.....	
3. วิธีการดำเนินการวิจัย.....	
3.1 การศึกษาค้นคว้าเทคโนโลยีและเครื่องมือที่ใช้ในการพัฒนาโมเดล.....	
3.2 การกำหนดเครื่องมือและสภาพแวดล้อมที่ใช้ในการทดลองวิจัย.....	
3.3 วัฏจักรการพัฒนางานวิจัยในการสร้างชุดข้อมูลฝึกสอน.....	
4. ผลการดำเนินงานวิจัย.....	
4.1 สมมติฐานการทดลองที่ 1.....	
4.2 สมมติฐานการทดลองที่ 2.....	
5. ผลการวิเคราะห์การทดลอง.....	
5.1 การวิเคราะห์กลไกการทำงานโดยรวมของโมเดล.....	
5.2 การวิเคราะห์ประสิทธิภาพการทำงานของโมเดล.....	

## สารบัญ (ต่อ)

หน้า

- 6. สรุปผลและข้อเสนอแนะ.....
- 6.1 สรุปผลการดำเนินงานวิจัย.....
- 6.2 ปัญหาและอุปสรรคที่พบในงานวิจัย.....
- 6.3 ข้อเสนอแนะและแนวทางการพัฒนางานวิจัยในอนาคต.....

# สารบัญตาราง

หน้า

## ตารางที่

- 3.1 ผลลัพธ์ความเป็นไปได้ที่เกิดขึ้นทั้งหมดจาก Data Field ที่กำหนด
- 3.2 ตัวอย่างการสร้างเงื่อนไขภายในชุดกฎของไฟร์วอลล์
- 4.1 ตารางการจำแนกความเป็นไปได้ของแต่ละ Data Field
- 4.2 ตารางการจำแนกความเป็นไปได้ของแต่ละกฎไฟร์วอลล์
- 4.3 ตารางผลการทดลองแบบ N Sample Rule set ที่ 1 (2 กฎ)
- 4.4 ตารางผลการทดลองแบบ N Sample Rule set ที่ 2 (4 กฎ)
- 4.5 ตารางผลการทดลองแบบ N Sample Rule set ที่ 3 (6 กฎ)
- 4.6 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 1 (2 กฎ)
- 4.7 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 2 (4 กฎ)
- 4.8 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 3 (6 กฎ)

# สารบัญรูป

หน้า

รูปที่

- 2.1 กระบวนการทำงานของกลไก Packet Filtering Firewall
- 2.2 กระบวนการทำงานของ Application Firewall
- 2.3 ส่วนประกอบที่สำคัญของ Packet Header Datagram
- 2.4 ขั้นตอนกระบวนการฝึกฝนปัญญาประดิษฐ์
- 2.5 ขั้นตอนการแยกหมวดหมู่และรูปแบบโมเดลที่จะศึกษา
- 2.6 ความแตกต่างระหว่าง Machine Learning และ Deep Learning
- 3.1 Block diagram วัฏจักรการพัฒนาสร้างชุดข้อมูลฝึกสอน
- 3.2 Block Diagram การกำหนดขอบเขตของข้อมูลทั้งหมดที่จะศึกษา
- 3.3 Block Diagram การสร้างชุดข้อมูลฝึกสอนสำหรับโมเดล
- 3.4 ตัวอย่างชุดข้อมูล Data set ที่ถูกสร้างขึ้นเมื่อแสดงผลออกมาเป็น Plain text
- 3.5 ตัวอย่างชุดข้อมูล Data set ที่ถูกสร้างขึ้นเมื่อแสดงผลออกมาเป็น Binary set
- 3.6 Block Diagram ขั้นตอนการนำโมเดลไปฝึกฝนด้วยชุดข้อมูลฝึกสอน
- 3.7 Block Diagram การสร้างชุดข้อมูลทดสอบโมเดล
- 3.8 Block Diagram การนำโมเดลไปประมวลผล
- 3.9 Reference Set ในการวิเคราะห์ความถูกต้องของโมเดล
- 3.10 Block Diagram ขั้นตอนการนำผลลัพธ์มาบันทึกผล
- 3.11 ตัวอย่างของตารางที่จะนำมาบันทึกผลลัพธ์การทดลอง
- 4.1 กราฟเวลาในการฝึกโมเดล: ชุดข้อมูลฝึกสอนต่อ 1 ญีโวลล์ (N Sample)
- 4.2 กราฟเวลาทำนายข้อมูลทดสอบ : จำนวนชุดฝึกสอนต่อ 1 ญี (N Sample)
- 4.3 กราฟความแม่นยำในการประมวลผล : จำนวนชุดฝึกสอนต่อ 1 ญี (N Sample)
- 4.4 กราฟเวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 ญี (Ratio)
- 4.5 กราฟเวลาในการทำชุดทดสอบ : อัตราส่วนข้อมูลฝึกสอนต่อ 1 ญี (Ratio)
- 4.6 กราฟเวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 ญี (Ratio)
- 5.1 กราฟเวลาที่ใช้ในการฝึกสอนโมเดล : จำนวนชุดข้อมูลฝึกสอนที่ใช้
- 5.2 เปรียบเทียบกราฟผลลัพธ์เวลาที่ใช้ในการประมวลของ N Sample และ Ratio
- 5.3 เปรียบเทียบกราฟผลลัพธ์ความแม่นยำของการแบ่งชุดข้อมูลฝึกสอนแต่ละแบบ
- 5.4 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของ N Sample
- 5.5 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของอัตราส่วน Ratio

## สารบัญรูป (ต่อ)

หน้า

รูปที่

- 5.6 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของ N Sample (2)
- 5.7 การเปรียบเทียบเมตริกซ์ Geometrical ก่อนและหลังยกกำลังสอง
- 5.8 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R1, Without Default)
- 5.9 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R1, With Default)
- 5.10 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R2, Without Default)
- 5.11 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R2, With Default)
- 5.12 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R3, Without Default)
- 5.13 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(N Sample, R3, With Default)
- 5.14 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R1, Without Default)
- 5.15 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R1, With Default)
- 5.16 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R2, Without Default)
- 5.17 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R2, With Default)
- 5.18 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R3, Without Default)
- 5.19 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้  
(Ratio, R3, With Default)

# บทที่ 1

## บทนำ

### 1.1 ความเป็นมาของโครงการ

Firewall ถูกสร้างขึ้นเพื่อจุดประสงค์ทางด้านความปลอดภัยทางเครือข่าย มีหน้าที่เปรียบเสมือนยามเฝ้าประตู โดยข้อมูลภายในเครือข่ายจะผ่านการคัดกรองข้อมูลด้วยหลักการของ Packet Filtering เมื่อเวลาผ่านไป การพัฒนาของเทคโนโลยีใหม่ๆ และรูปแบบการโจมตีทางเครือข่ายที่มีมากขึ้น Firewall แบบเก่าที่กำหนดโดยผู้ควบคุมระบบเพียงอย่างเดียว ไม่สามารถตอบโจทย์ทางด้านความปลอดภัยได้ ทำให้มีการนำปัญญาประดิษฐ์ หรือ AI มาประยุกต์ใช้งานกับ Firewall ให้มีความคิดและตัดสินใจเลือกคัดกรอง Packet ได้เอง ผู้จัดทำมีความคิดที่จะพัฒนา AI Firewall ให้มีประสิทธิภาพมากยิ่งขึ้น โดยนำมาประยุกต์ใช้กับกระบวนการเรียนรู้แบบ Deep Neural Network และมีชุดข้อมูล Packet ฝึกสอนที่สร้างขึ้นอ้างอิงตามนโยบายข้อกำหนดจาก Firewall Rules เพื่อแก้ปัญหาข้อมูลฝึกสอนที่ไม่ได้เป็นไปตามนโยบายข้อกำหนด ที่แต่เดิมต้องเอาข้อมูลการโจมตีที่เคยเกิดขึ้นมาก่อนเป็นข้อมูลฝึกสอน

### 1.2 วัตถุประสงค์

1. เพื่อให้เข้าใจหลักการทำงานของ Neural Network ที่จะใช้พัฒนาปัญญาประดิษฐ์
2. เพื่อพัฒนาสร้างชุดข้อมูลฝึกสอนให้เป็นไปตามนโยบายข้อกำหนดตาม Firewall Rules
3. เพื่อให้ชุดข้อมูล Network Packet ที่สร้างขึ้นสามารถฝึกสอนได้อย่างถูกต้องและแม่นยำ เมื่อนำไปใช้กับ AI ที่มีการเรียนรู้แบบ Deep Neural Network Model
4. เพื่อแก้ไขข้อจำกัดของข้อมูลฝึกสอน Firewall ให้ผ่านเงื่อนไขที่กำหนด เช่น เวลาที่ใช้ หรือ ปริมาณของข้อมูล Packet

### 1.3 วิธีการดำเนินงาน

พัฒนาสร้างชุดข้อมูลฝึกสอน Network Packet ที่สร้างขึ้นโดยมีการอ้างอิงจาก Firewall Rules ไปใช้กับ AI Firewall ที่มีการเรียนรู้แบบ Neural Network Model และทำการตรวจสอบความถูกต้อง ความผิดพลาดที่ได้ เปรียบเทียบกับ Firewall Rules ที่กำหนด โดยทำการทดลองหลายๆครั้ง เปลี่ยนตัวแปรและปัจจัยต่างๆ เพื่อหาวิธีการที่ทำให้ระบบสามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพมากที่สุด

### 1.4 ขอบเขตของโครงการ

พัฒนา Neural Network Model และชุดข้อมูลฝึกสอน Network Packet ที่สร้างขึ้นโดยอ้างอิงจาก Firewall Rules นำไปผ่านการเรียนรู้และทำการทดสอบ ลองเปลี่ยนปัจจัยและค่าตัวแปรต่างๆ เปรียบเทียบผลลัพธ์ในแต่ละรูปแบบ ใช้ความถูกต้อง ความผิดพลาดที่อิงจากกฎของ Firewall Rules เป็นเกณฑ์ในการวัดผล ศึกษาหาวิธีการและผลลัพธ์ที่ดีที่สุดภายใต้การทำงานของโปรแกรมคอมพิวเตอร์

### 1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. พัฒนาทักษะการเขียนโปรแกรมที่เขียนด้วยภาษา Python
2. เรียนรู้วิธีการสร้างชุดข้อมูลฝึกสอน สามารถประยุกต์ใช้กับปัญญาประดิษฐ์ได้
3. เรียนรู้วิธีการพัฒนาอัลกอริทึมที่ช่วยลดเวลาเพิ่มประสิทธิภาพในการคำนวณข้อมูลได้
4. สามารถประยุกต์ learning model ไปใช้กับปัญญาประดิษฐ์รูปแบบอื่น เช่น การทำแชทบอท โปรแกรมวิเคราะห์ข้อมูล หรือ ระบบปฏิบัติการตอบโต้อัตโนมัติ



## บทที่ 2

# ทฤษฎีการนำโครงข่ายระบบประสาทเชิงลึก มาใช้ในการทำงานของไฟร์วอลล์

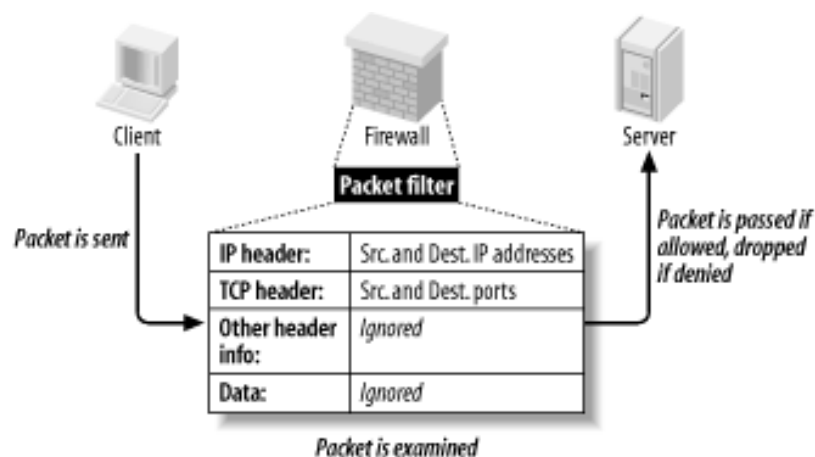
### 2.1 เทคโนโลยีของไฟร์วอลล์และโครงข่ายระบบประสาทเชิงลึก

#### 2.1.1 Firewall

Firewall เป็นระบบควบคุมและรักษาความปลอดภัยของระบบเครือข่าย คัดกรองข้อมูลเข้าออกในช่องทางอินเทอร์เน็ต เปรียบเสมือนยามเฝ้าประตู คอยป้องกันการโจมตี สแปม ผู้บุกรุกต่างๆ ที่ไม่หวังดีต่อระบบ และยังสามารถใช้ควบคุมการใช้งานของโปรแกรมที่ต้องการ ในปัจจุบันมีการใช้งานได้ทั้งระบบ Hardware และ Software ขึ้นอยู่กับความเหมาะสม ผลลัพธ์ที่ออกมาจาก Firewall จะพิจารณาการกระทำของ Packet ออกมาเป็น Allow หรือ Deny

##### 2.1.1.1 Packet Filtering

ระบบการทำงานของ Firewall ทำงานในระบบ Internet Layer และ Transport Layer ตรวจสอบและคัดกรอง Packet ที่เข้ามาในเครือข่าย โดยพิจารณาจาก Packet Header ตัดสินใจว่าจะทำการ Allow หรือ Deny โดยใช้กฎของ Firewall ในการอ้างอิง ซึ่ง Firewall แบ่งประเภทตามลักษณะการทำงาน ได้แก่



รูปที่ 2.1 กระบวนการทำงานของกลไก Packet Filtering Firewall

### 2.1.1.2 Stateful Filtering

Stateful Filtering จะมีเก็บสถานะ Packet ใดที่เคยถูกปล่อยผ่านและเก็บบันทึกไว้ใน State Table ทำให้การทำงานของ Firewall นี้จะถูกรวบรวมเริ่มจากที่ State Table ก่อน ถ้าหาก Packet ที่กำลังถูกรวบรวมอยู่ยังไม่เคยถูกปล่อยผ่านยังไม่มีการเก็บสถานะเอาไว้ถึงจะไปพิจารณากฎของไฟร์วอลล์เป็นอันดับถัดไป กลไกนี้จะช่วยไฟร์วอลล์ทำงานได้เร็วขึ้น เพราะช่วยลดระยะเวลาในการทำงานไม่ต้องเสียเวลาพิจารณาทุก Packet Header ในกลไก Packet Filtering

### 2.1.1.3 Application Firewall

มีชื่อเรียกได้อีกอย่างหนึ่งว่า “Application-level Firewall” หรือ “Application Gateway” เป็น Firewall ชนิดที่ติดตั้งบนเครื่องคอมพิวเตอร์แยกต่างหาก ทำให้คอมพิวเตอร์เครื่องดังกล่าวทำหน้าที่เป็น Firewall โดยเฉพาะ อย่างไรก็ตาม Application Firewall สามารถกรอง Packet ที่จะผ่านเข้ามาในเครือข่ายอีกทั้งยังตรวจสอบเนื้อหาใน Packet ได้เช่นเดียวกับ Stateful Filtering Firewall นอกจากนี้ Application Firewall ยังทำหน้าที่คล้ายกับ Proxy Server ในการให้บริการคำร้องขอของผู้ใช้ได้อีกด้วย โดยความสามารถของ Application Firewall สามารถแบ่งทำได้ดังนี้

#### Security

การยืนยันตัวตนด้วย AAA คือ Authentication, Authorization และ Audit โดยการสร้าง Token ไปให้ทั้งผู้รับและผู้ส่ง มีการกำหนด Policy เพื่อการเข้าถึงข้อมูล และยังทำการเก็บข้อมูลการเข้าออกของ Policy นั้นๆ อีกทั้งยังมีการป้องกันด้วยการตรวจสอบข้อมูลที่ได้รับก่อนว่าถูกต้องตามโครงสร้างที่ได้กำหนดไว้หรือไม่

#### Integration

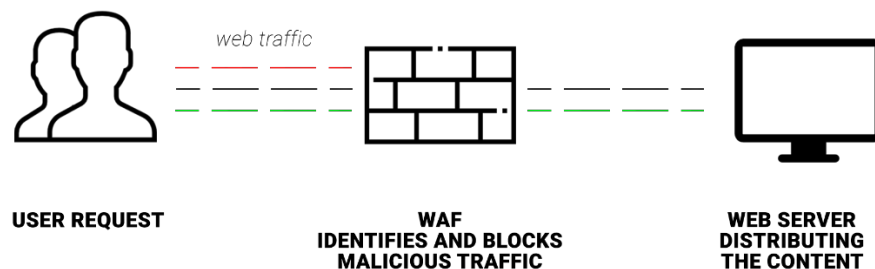
การสร้างการเชื่อมต่อเข้ากับระบบต่างๆ ให้สามารถทำงานร่วมกันได้ เช่น ถ้าหากระบบที่ใช้มีโปรโตคอลที่แตกต่างกัน มันจะทำการแปลงโครงสร้างข้อความโดยการจับคู่ข้อมูล

## Control and Managing

การควบคุมปริมาณของข้อความที่จะวิ่งเข้าไปหา Server โดยการกำหนด Policy แยกตามประเภทของ API และประเภทของข้อมูล สำหรับการควบคุมปริมาณข้อความนี้จะเป็นการป้องกันการถูกผู้ไม่หวังดีโจมตีจากช่องโหว่ของระบบได้ เช่น เรามี API ที่เปิดให้ลูกค้าหรือบุคคลอื่นๆเข้ามาใช้งานได้ ถ้าหากไม่มีการกำหนดปริมาณการเรียกใช้ API หรือเส้นทางของข้อมูล ก็จะเกิดช่องโหว่ของระบบที่ผู้ไม่หวังดีสามารถทำการ DOS ได้

## Optimizing

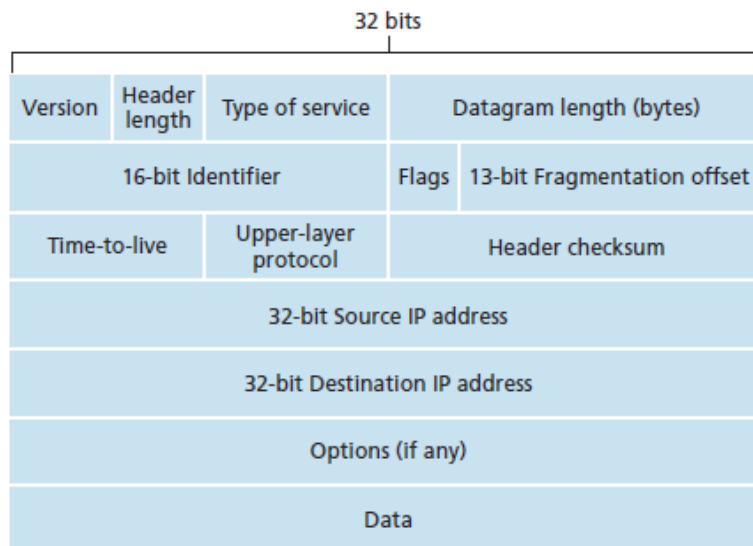
การลดภาระการทำงานของ Server โดยการทำ SSL และนำภาระงานจากการถอดรหัสที่ Server ไปให้ไฟร์วอลล์ทำงานแทน จะทำให้ Server มีทรัพยากรเหลือพอที่จะรองรับการทำงานมากขึ้น



รูปที่ 2.2 กระบวนการทำงานของ Application Firewall

### 2.1.2 Packet Header

**Packet Header** เป็นโปรโตคอลอินเทอร์เน็ต มาตรฐานที่ทำให้อินเทอร์เน็ตสามารถเชื่อมต่อเข้าหากัน ติดต่อสื่อสารข้อมูลได้ด้วยการกำหนดวิธีการติดต่อสื่อสารร่วมกัน ในส่วนของ **Packet Header** จะเป็นลำดับชั้นโครงสร้างประกอบไปด้วย **Field** ข้อมูลที่บ่งบอกถึงวัตถุประสงค์และลักษณะการทำงานของ **Packet** โดยองค์ประกอบของ **Packet Header** มีดังนี้

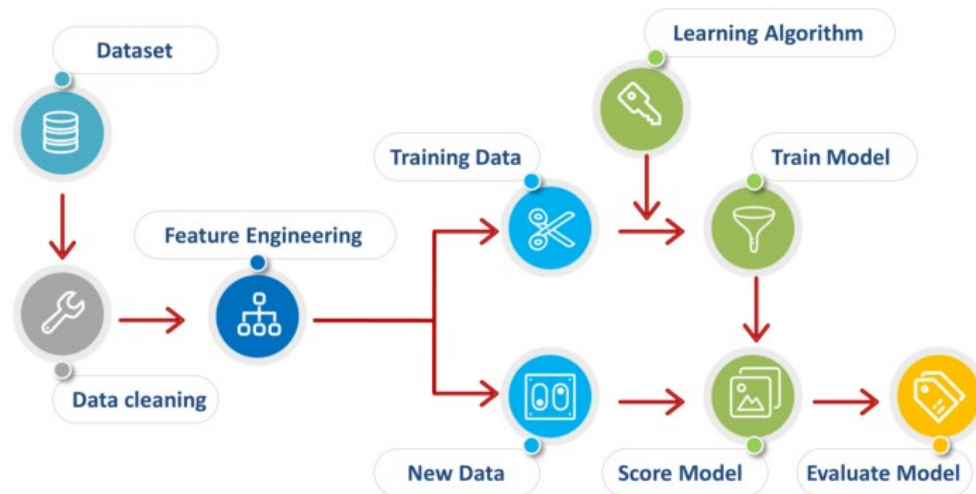


รูปที่ 2.3 ส่วนประกอบที่สำคัญของ Packet Header Datagram

- Version ส่วนที่ระบุเวอร์ชันโปรโตคอลของ Datagram
- Header length ส่วนที่ระบุขนาดของ Datagram Header
- Type of service ส่วนที่ระบุประเภทของ Datagram
- Datagram length ส่วนที่ระบุขนาดของ Datagram ทั้งหมดรวมถึง Datagram Header
- Identifier ส่วนที่มีไว้เพื่อยืนยันตัว หากมีการทำ Fragmentation
- Flags ส่วนที่ระบุว่า Datagram นี้จะทำการ Fragmentation หรือไม่
- Fragmentation offset ส่วนที่แสดงให้เห็นถึงจำนวนของข้อมูลก่อนทำการ Fragmentation
- Time-to-live ส่วนที่กำหนดวงจรชีวิตของ Datagram เพื่อป้องกันไม่เกิด Loop ในเครือข่าย
- Protocol ส่วนที่ระบุโปรโตคอลที่ใช้ใน Datagram นี้
- Header checksum ส่วนที่ใช้สำหรับตรวจสอบความถูกต้อง Datagram Header
- Source and destination IP addresses ส่วนที่ระบุที่อยู่ของ IP ต้นทางกับ IP ปลายทาง
- Options ส่วนเพิ่มเติมที่คอยเก็บข้อมูลเช่น เส้นทางที่ใช้โดยเก็บไว้เพื่อตรวจสอบการทำงาน

### 2.1.3 Artificial Intelligent

Artificial Intelligence คือ เครื่องจักรอัจฉริยะที่มีความสามารถในการทำความเข้าใจ เรียนรู้ องค์ความรู้ต่างๆ เช่น การรับรู้ การให้เหตุผล ในการแก้ไขปัญหาต่างๆเพื่อปฏิบัติงานตามความต้องการของมนุษย์ เครื่องจักรที่มีความสามารถนี้ถูกเรียกอีกชื่อหนึ่งว่า “ปัญญาประดิษฐ์”



รูปที่ 2.4 ขั้นตอนกระบวนการฝึกฝนปัญญาประดิษฐ์

AI ถูกจำแนกเป็น 3 ระดับตามความสามารถดังนี้

**Narrow Artificial Intelligent** ปัญญาประดิษฐ์เชิงแคบ คือ AI ที่มีความสามารถเฉพาะทาง ได้ดีกว่ามนุษย์ เช่น เครื่องจักรที่ใช้ในการผ่าตัด

**General Artificial Intelligent** ปัญญาประดิษฐ์ทั่วไป คือ AI ที่มีความสามารถระดับเดียวกับมนุษย์สามารถทำทุกอย่างในประสิทธิภาพที่ใกล้เคียงกับมนุษย์

**Strong Artificial Intelligent** ปัญญาประดิษฐ์แบบเข้ม คือ AI ที่มีความสามารถมากกว่ามนุษย์ในหลายๆด้าน

และจากการนำปัญญาประดิษฐ์มาประยุกต์ใช้ในการแก้ไขปัญหา มุมมองต่อ AI ที่แต่ละคน มีอาจไม่เหมือนกัน ขึ้นอยู่กับว่า เราต้องการความฉลาดโดย คำนึงถึงพฤติกรรมที่มีต่อสิ่งแวดล้อม หรือคำนึงการคิดได้ของผลผลิต AI ดังนั้นจึงมีคำนิยาม AI ตามความสามารถที่มนุษย์ต้องการ ให้ มันแบ่งได้ 4 กลุ่ม ดังนี้

### **Thinking humanly (การคิดคล้ายมนุษย์)**

natural language processing สื่อสารกับ มนุษย์ได้ด้วยภาษาที่มนุษย์ใช้ เช่น ภาษาอังกฤษ เป็นการประมวลผลภาษาธรรมชาติ

computer vision มีประสิทธิภาพสัมผัสคล้ายมนุษย์ เช่นคอมพิวเตอร์วิทัศน์ รับภาพได้โดยใช้อุปกรณ์รับสัญญาณภาพ

machine learning เพื่อปรับให้เข้ากับสถานการณ์ใหม่และ ตรวจจับและคาดการณ์รูปแบบ

### **Thinking rationally (คิดอย่างมีเหตุผล)**

คิดอย่างมี เหตุผล หรือคิดถูกต้อง โดยใช้หลักตรรกศาสตร์ในการคิดหาคำตอบอย่างมีเหตุผล เช่น ระบบผู้เชี่ยวชาญ

### **Acting humanly (การกระทำคล้ายมนุษย์)**

การคิดคล้าย มนุษย์ ก่อนที่จะทำให้เครื่องคิดอย่างมนุษย์ได้ ต้องรู้ก่อนว่ามนุษย์มีกระบวนการคิดอย่างไร ซึ่งการวิเคราะห์ลักษณะการคิดของมนุษย์เป็นศาสตร์ด้าน cognitive science เช่น ศึกษาโครงสร้างสามมิติของเซลล์สมอง การแลกเปลี่ยนประจุไฟฟ้าระหว่างเซลล์สมอง วิเคราะห์การเปลี่ยนแปลงทางเคมีไฟฟ้าในร่างกายระหว่างการคิด ซึ่งจนถึงปัจจุบันเรายังไม่รู้แน่ชัดว่า มนุษย์เรา คิดได้อย่างไร

### **Acting rationally (การกระทำอย่างมีเหตุผล)**

กระทำอย่างมีเหตุผล เช่น agent (agent เป็น โปรแกรมที่มีความสามารถในการกระทำ หรือเป็นตัวแทนในระบบอัตโนมัติต่าง ๆ ) สามารถกระทำอย่างมีเหตุผลคือ agent ที่กระทำการเพื่อบรรลุเป้าหมายที่ได้ตั้งไว้ เช่น agent ใน ระบบขับรถอัตโนมัติที่มีเป้าหมายว่าต้องไปถึงเป้าหมายในระยะทางที่สั้นที่สุด ต้องเลือกเส้นทางที่ไปยังเป้าหมายที่สั้นที่สุดที่เป็นไปได้จึงจะเรียกได้ ว่า agent กระทำอย่างมีเหตุผล อีกตัวอย่างเช่น agent ใน เกมหมากรุกมีเป้าหมายว่าต้องเอาชนะคู่ต่อสู้ ต้องเลือกเดินหมากที่จะทำให้คู่ต่อสู้แพ้ให้ได้ เป็นต้น

## 2.1.4 Machine Learning

Machine Learning คือ ส่วนการเรียนรู้ของเครื่อง ถูกใช้งานเสมือนเป็นสมองของปัญญาประดิษฐ์ในการสร้างความฉลาด มักจะใช้เรียกโมเดลที่เกิดจากการเรียนรู้ของปัญญาประดิษฐ์ โดยมนุษย์มีหน้าที่เขียนโปรแกรมให้เรียนรู้จากชุดข้อมูลฝึกสอนหรือ Training set และอาศัยกลไกที่เป็นโปรแกรม หรือเรียกว่า Algorithm ที่มีหลากหลายแบบ โดยมี Data Scientist เป็นผู้ออกแบบ หนึ่งใน Algorithm ที่ได้รับความนิยมสูง คือ Deep Learning ซึ่งถูกออกแบบมาให้ใช้งานได้ง่าย และประยุกต์ใช้ได้หลายลักษณะงาน อย่างไรก็ดีตาม ในการทำงานจริง Data Scientist จำเป็นต้องออกแบบตัวแปรต่างๆ ทั้งในตัวของ Deep Learning เอง และต้องหา Algorithm อื่นๆ มาเป็นคู่เปรียบเทียบกับ เพื่อมองหา Algorithm ที่เหมาะสมที่สุดในการใช้งานจริง โดยตามหลักแล้วจะแบ่งประเภทของ Machine Learning ได้ดังนี้

### 2.1.4.1 Supervised

การทำให้เครื่องคอมพิวเตอร์สามารถเรียนรู้ได้จากชุดข้อมูลฝึกสอนหรือ Training set ก่อนที่จะประมวลผล โดยมนุษย์จะเป็นผู้กำหนดคุณลักษณะความสัมพันธ์เฉพาะของข้อมูลที่ต้องการให้เครื่องคอมพิวเตอร์เรียนรู้ หรือที่เรียกว่า Label และเมื่อโมเดลผ่านการเรียนรู้แล้ว จะสามารถแยกแยะประเภทมีวิธีการคิดที่เริ่มมีเหตุผล เมื่อข้อมูลที่ต้องการวิเคราะห์มีจำนวนที่มากขึ้นจำเป็นต้องมีข้อมูลที่เป็น Training set มากขึ้นเช่นเดียวกัน โดยการเรียนรู้แบบ Supervised Learning นี้จะประกอบไปด้วยดังนี้

#### 2.1.4.1.1 Classification

คือการสอนโมเดลให้สามารถแบ่งหรือแยกประเภทกลุ่มข้อมูลได้ โดยอ้างอิงจากความสัมพันธ์และน้ำหนักของข้อมูลแต่ละ Label ตัวอย่างเช่น การแยกกลุ่มผู้ป่วยว่าเป็นเนื้องอกในสมอง ซึ่งจะมีปัจจัยต่างๆมากมายไม่ว่าจะเป็น ขนาด, รูปร่าง, ตำแหน่ง หรือ สีผิว ซึ่งถ้าหากมีข้อมูลเพียงแค่ Label เดียว ไม่สามารถพิสูจน์หรือแบ่งกลุ่มได้

#### 2.1.4.1.2 Regression

การสอนโมเดลโดยอิงจากผลลัพธ์ที่ผ่านมา โดยผลลัพธ์จะเป็นการประมาณค่าความเป็นไปได้ที่จะเกิดขึ้นต่อ ทำให้เหมาะแก่การวิเคราะห์ความสัมพันธ์ของตัวแปรที่อยู่ในรูปกราฟ เช่น การหาความสัมพันธ์ระหว่างขนาดของบ้านและราคา การประเมินราคาหุ้น

#### 2.1.4.2 Unsupervised

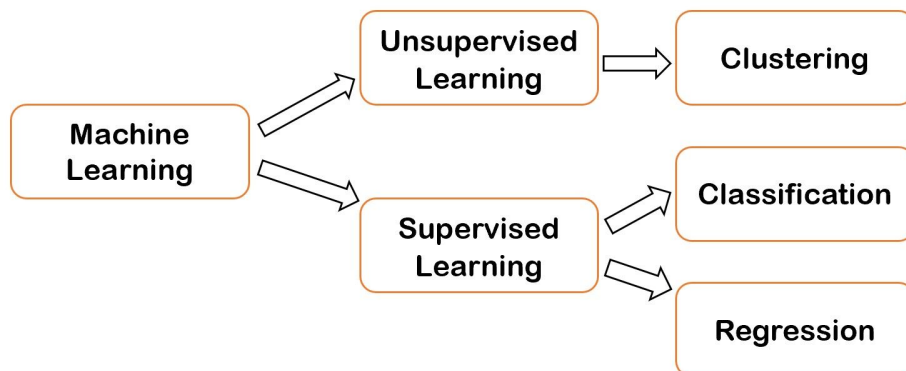
รูปแบบการเรียนรู้ที่ไม่จำเป็นต้องใช้ชุดข้อมูลฝึกสอน แต่เป็นการป้อนข้อมูล Test set ไปประมวลผลเพียงอย่างเดียว ทำให้ผลลัพธ์ที่ออกมาไม่รู้ผลลัพธ์แน่ชัด ซึ่งอัลกอริทึมจะวิเคราะห์และหาโครงสร้างของข้อมูลเอง

##### 2.1.4.2.1 Clustering

เป็นการกำหนดให้เครื่องคอมพิวเตอร์หาวิธีแบ่งกลุ่มหรือจัดกลุ่มข้อมูลเอง เปรียบเสมือนการลด Label ของข้อมูลที่มีปริมาณมาก จัดกลุ่มหาข้อมูลที่มีความสัมพันธ์ใกล้เคียงกัน ผลลัพธ์ที่ได้ออกมาจะมีปริมาณ Label ที่น้อยลงเป็นอย่างมาก

##### 2.1.4.2.2 Dimensionality Reduction

เป็นการลดการบีบอัดและลดมิติข้อมูลจำนวนมากให้มีจำนวนลดลง โดยที่ข้อมูลยังครบถ้วน และยังสามารถนำไปจำแนกข้อมูลได้เหมือนเดิม



รูปที่ 2.5 ขั้นตอนการแยกหมวดหมู่และรูปแบบโมเดลที่จะศึกษา

#### 2.1.4.3 Reinforcement Learning

เป็นการเรียนรู้ด้านหนึ่งของ Machine Learning มักใช้พัฒนาหุ่นยนต์หรือการเรียนรู้ที่อยู่ในเกมคอมพิวเตอร์ เช่น การลองผิดลองถูกไปเรื่อยเพื่อหาผลลัพธ์ที่ดีที่สุด ประเมินออกมาเป็นคะแนน โดยชุดข้อมูลทดสอบจะเป็นสภาพแวดล้อมโดยรอบขึ้นอยู่กับการต้องการของผู้พัฒนา



### 2.1.5 Deep Learning

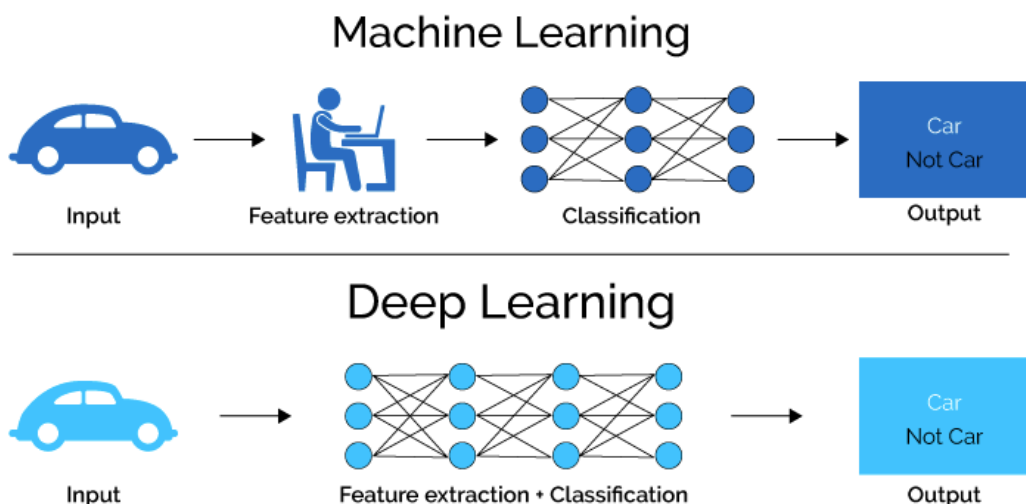
Deep learning คือ อัลกอริทึมการเรียนรู้เชิงลึกที่ใช้หลักการ Artificial Neural Networks ที่มีรูปแบบการทำงานคล้ายคลึงกับเซลล์ประสาทที่เชื่อมต่อกันเป็นโครงข่ายประสาทในสมองมนุษย์ เหมาะกับการวิเคราะห์ข้อมูลขนาดใหญ่ที่มีความซับซ้อน เช่น การจำแนกรูปภาพ การจำแนกใบหน้า ประกอบไปด้วย โครงสร้างของหน่วยประมวลผลจำนวนมากคือเซลล์ประสาท หรือ Neuron โดยอัลกอริทึมนี้จะประกอบไปด้วยชั้นต่างๆ ดังนี้

Input Layer มีหน้าที่รับข้อมูลเข้ามาประมวลผลและส่งต่อไปให้ Hidden Layer

Hidden Layer มีหน้าที่คำนวณและประมวลผลข้อมูลโดยสามารถมีได้หลายชั้น หลายขนาด ขึ้นอยู่กับความซับซ้อนของข้อมูล

Output Layer มีหน้าที่ส่งผลลัพธ์ข้อมูลที่ผ่านการประมวลผลแล้วออกมา

เมื่อเริ่มการฝึกฝนจะเริ่มจากการสุ่มค่าถ่วงน้ำหนัก (Weight) และจะเริ่มปรับผลลัพธ์เอามาคูณกับค่าถ่วงน้ำหนักแล้วบวกด้วยค่าความเอนเอียงของข้อมูล (Bias) หลังจากนั้นจะนำผลลัพธ์ที่ได้มาในแต่ละขาของ Neural Network มารวมกันแล้วมาผ่านฟังก์ชันส่งต่อไปให้ลำดับชั้นถัดไปประมวลผลมีการใช้วิธีการประมวลผลทางคณิตศาสตร์ (Activation Function) โดยทุกวันนี้มีการประยุกต์ใช้อย่างแพร่หลาย แบ่งชนิดโครงข่ายประสาทออกเป็นดังนี้



รูปที่ 2.6 ความแตกต่างระหว่าง Machine Learning และ Deep Learning

#### 2.1.5.1 โครงข่ายประสาทแบบป้อนไปข้างหน้า (Forward Propagation)

Feed-forward neural networks ถือเป็น โมเดลที่มีโครงสร้างที่เรียบง่ายที่สุด เพราะว่า การดำเนินการของข้อมูลจะเป็นไปในทิศทางเดียว ก็คือ รับข้อมูลจาก input layer แล้วส่งไปต่อไปยัง hidden layer เลื่อนๆ จนกระทั่งถึง output layer ก็จะหยุด สังเกตได้ว่าจะไม่มีวงวน หรือ loop เกิดขึ้นเลย

#### 2.1.5.2 โครงข่ายแบบวนซ้ำ (Recurrent neural networks : RNN)

Recurrent neural networks คือ neural networks หลายเลเยอร์ที่สามารถเก็บข้อมูล information ไว้ที่ node จึงทำให้มันสามารถรับข้อมูลเป็นแบบลำดับ (data sequences) และ ให้ผลลัพธ์ออกเป็นลำดับของข้อมูลได้ อธิบายอย่างง่ายๆ RNN ก็คือ neural network เชื่อมต่อกันหลายๆอันและยังสามารถต่อกันเป็นวงวนหรือ loop ได้นั่นเอง เพราะฉะนั้น RNN จึงเหมาะสมในการประมวลผลข้อมูลที่เป็นลำดับอย่างมาก

## 2.2 ทบทวนวรรณกรรม

### 2.2.1 การนำเอาความสามารถของ GPU มาใช้ในการคำนวณ

การที่เราเลือกใช้ GPU ในการทำ Machine Learning เนื่องจากตัว GPU นั้นมีหน่วยความจำที่ให้ค่าแบนด์วิดท์ที่สูง และตัว GPU เองยังออกแบบให้สามารถแก้สมการทางคณิตศาสตร์ได้อย่างรวดเร็ว นอกจากนี้ยังมีจำนวนหน่วยประมวลผลที่มีมากกว่า CPU หลายเท่าตัว จึงทำให้มีอัตราการประมวลผลที่สูงกว่า CPU และยังมีแพลตฟอร์มของ Nvidia ที่รองรับอย่าง CUDA ซึ่งเป็น Parallel Computing แพลตฟอร์มเพื่อช่วยให้นักพัฒนาสร้าง Tools ในการเรียกใช้การประมวลผลของ GPU และยังมี library อย่าง NVIDIA cuDNN ซึ่งรองรับการทำ Deep Neural Network โดยตัว cuDNN ได้อำนวยความสะดวกปรับแต่งขั้นสูงสำหรับการทำงานของ DNN เช่น forward และ backward convolution pooling normalization activation layers เป็นต้น

### 2.2.2 ทฤษฎี Rule of Thumb ในการหาจำนวนของ Hidden Layer

การตัดสินใจเลือกจำนวน Neurons ใน Hidden Layers ถือเป็นส่วนสำคัญในการตัดสินใจภาพรวมของสถาปัตยกรรมโครงข่ายประสาทเทียม โดย Hidden Layers นั้นจะไม่ค่อยมีผลกับองค์ประกอบภายนอกแต่จะมีผลอย่างมากกับผลลัพธ์ที่จะออกมา จึงทำให้การกำหนดจำนวน Hidden Layers และ จำนวน Neurons ต้องพิจารณาอย่างระมัดระวัง หากเราใช้จำนวน Neurons น้อยเกินไปก็จะเกิดปัญหาที่เรียกว่า Underfitting โดยจะเกิดขึ้นเมื่อมีจำนวน Neurons ใน Hidden Layers น้อยเกินไปจนไม่สามารถตรวจจับสัญญาณในข้อมูลที่ซับซ้อนได้อย่างเพียงพอ แต่ในทางกลับกันหากเราใช้จำนวน Neurons มากเกินไปก็จะเกิดปัญหา Overfitting โดยจะเกิดขึ้นเมื่อความจุของข้อมูลที่จะประมวลผลมีมากเกินไป ซึ่งจะไปจำกัดข้อมูลที่จะอยู่ในชุดฝึกสอนทำให้ไม่เพียงพอต่อการเรียนรู้ของ Neurons ดังนั้นทำให้ต้องการกำหนดจำนวน Neurons ที่ไม่น้อยเกินไปหรือมากเกินไป โดยมีหลักการอย่างง่ายในการกำหนดจำนวน Neurons ตามนี้

- จำนวน Neurons ควรอยู่ในช่วงขนาดของ Input Layer และ Output Layer
- จำนวน Neurons ควรมีขนาดเป็น 2 : 3 ของขนาด Input layer รวมกับ Output layer
- จำนวน Neurons ควรมีขนาดน้อยกว่า 2 เท่าของขนาด Input Layer

โดยกฎทั้งสามที่ยกมานั้นเป็นเพียงส่วนหนึ่งในตัวเลือกให้สามารถนำไปใช้เพื่อให้ไม่ต้องมาสุ่มจำนวน Neurons ใหม่ซึ่งเท่าทำให้ไม่เสียเวลาที่ต้องนำไปทดลองกับจำนวน Neurons ที่สุ่มขึ้นมาใหม่

## บทที่ 3

### วิธีการดำเนินการวิจัย

การดำเนินการวิจัยการสร้างชุดข้อมูลในการฝึกสอนไฟร่วลล์ปัญญาประดิษฐ์ด้วยเทคโนโลยีโครงข่ายประสาทเทียมจากกฎของไฟร่วลล์ มีเป้าหมายเพื่อพัฒนาชุดข้อมูลฝึกสอนที่สร้างจากกฎของไฟร่วลล์ เพื่อให้ชุดข้อมูลฝึกสอนสามารถสอนโมเดลได้ถูกต้องและแม่นยำอย่างมีประสิทธิภาพ

#### 3.1. การศึกษาค้นคว้าเทคโนโลยีและเครื่องมือที่ใช้ในการพัฒนาโมเดล

ในการดำเนินการวิจัย เราเลือกใช้ Python เป็นภาษาหลักในการพัฒนาโปรแกรมสร้างชุดข้อมูลฝึกสอนและโมเดล DNN ดังนั้นเพื่อให้การทำงานและการใช้งานเป็นไปตามที่งานวิจัยต้องการ จึงจำเป็นต้องศึกษาความเข้ากันได้ของเครื่องมือและไลบรารีที่เกี่ยวข้องในการพัฒนา

- Anaconda3 โปรแกรมจัดการแพ็คเกจและสร้าง Environment ที่จำเป็นในการเขียนซอฟต์แวร์ภาษา Python เหมาะแก่งาน Data Visualization, Machine Learning, Neural Network และยังสามารถใช้งานร่วมกับ IDE ได้หลากหลาย

Version: Anaconda 3.8 64-Bit

- Spyder โปรแกรมพัฒนาซอฟต์แวร์ด้วยภาษา Python สามารถตรวจสอบตัวแปรได้ง่าย

Version: Spyder 4.1.4

- TensorFlow ไลบรารีพื้นฐานในการพัฒนา Neural Network Model

Version: TensorFlow 2.3.0 สามารถใช้ได้กับ Python 64-Bit เท่านั้น

- Sklearn เป็นเครื่องมือสำคัญในการทำ Model Selection และ Data Preprocessing ทำงานโดยพื้นฐานของ Numpy

Version: Scikit-learn 0.23.2

- Keras เป็น Deep Learning Framework ที่สำคัญ อีกทั้งสามารถประมวลผลได้ทั้ง CPU และ GPU

Version: Keras 2.4.3

- Pandas เป็นไลบรารีช่วยในการจัดกลุ่ม แยกประเภทข้อมูลกลุ่มโครงสร้าง เช่น ไฟล์นามสกุล CSV

Version: Pandas 1.1.2

- Pip เครื่องมือที่ช่วยในการติดตั้งแพ็คเกจในภาษา Python  
Version: pip 20.2.3
- Tkinter ไลบรารีพัฒนาการสร้าง GUI ด้วยภาษา Python  
Version: Tk 8.6.10
- NVIDIA CUDA เครื่องมือช่วยให้คอมพิวเตอร์สามารถประมวลผลผ่าน GPU ได้  
Version: CUDA 11.1.0
- NVIDIA cuDNN เครื่องมือช่วยในการประมวลผล DNN ผ่าน GPU  
Version: cuDNN 8.0

### 3.2. การกำหนดเครื่องมือและสภาพแวดล้อมที่ใช้ในการทดลองวิจัย

#### 3.2.1 ประสิทธิภาพของเครื่องคอมพิวเตอร์ที่ใช้ในงานวิจัย

ผลลัพธ์ที่ได้จากการทดลองมีเวลามาเกี่ยวข้องด้วย ดังนั้นประสิทธิภาพในการทดลองแต่ละครั้งจะจำเป็นต้องใช้เครื่องคอมพิวเตอร์เดียวกันในการประมวลผล

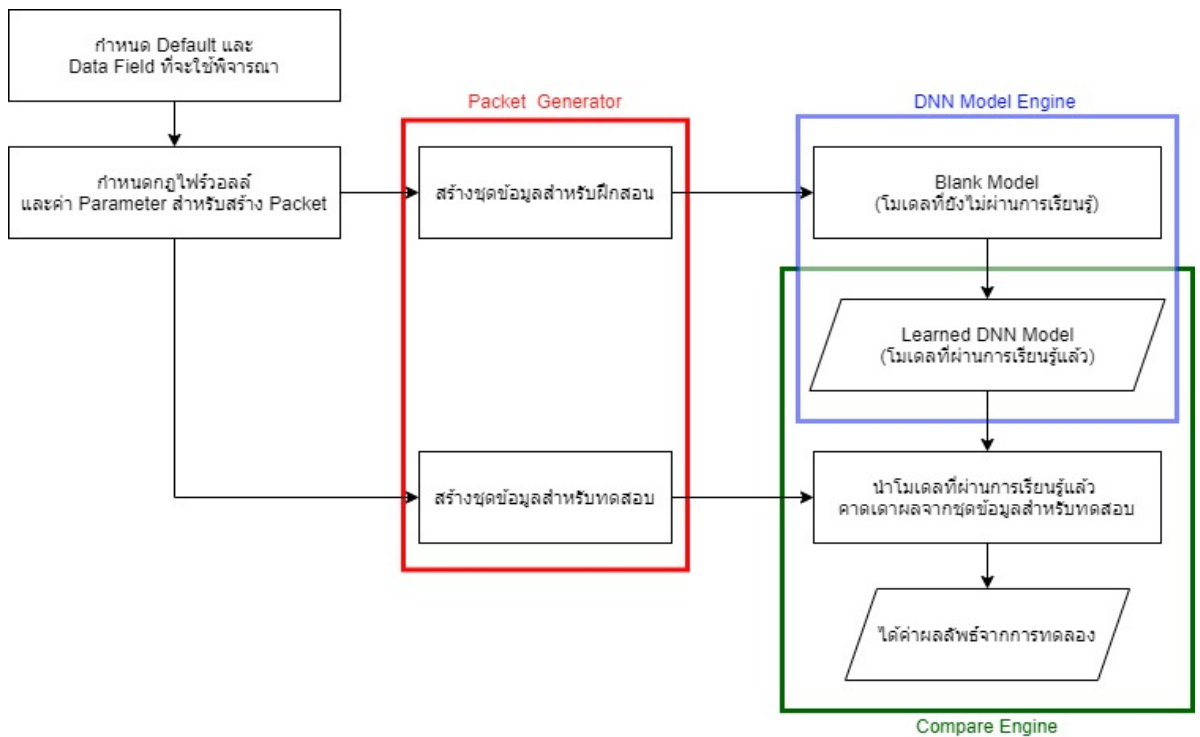
- Computer Specification (Hardware)
  - OS: Windows 10 Enterprise x64 bit operating system
  - CPU: Intel(R) Core(TM) i7-3770K CPU @ 3.50GHz
  - RAM: DDR3(1600) 16GB (8GB x 2)
  - Mainboard: Gigabyte H61M-DS2
  - VGA: Gigabyte Geforce GTX1060 6GB

#### 3.2.2 โปรแกรมที่ต้องพัฒนาขึ้นเองเพื่อใช้ในงานวิจัย

- Packet Generator  
โปรแกรมสำหรับสร้างชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบภายใต้เงื่อนไขที่กำหนด
- Deep Learning Model Engine  
โปรแกรมสำหรับฝึกสอนและสร้างโมเดล DNN จากข้อมูลที่กำหนดไว้
- Evaluate / Comparing Program  
โปรแกรมสำหรับสรุปผลประสิทธิภาพการทำงานและความแม่นยำของโมเดล

### 3.3. วัฏจักรการพัฒนางานวิจัยในการสร้างชุดข้อมูลฝึกสอน

ในการวิจัยจะมุ่งเน้นไปที่การพัฒนาชุดข้อมูลฝึกสอนที่ทำให้โมเดลสามารถประมวลผลและคาดเดาผลลัพธ์ได้อย่างมีประสิทธิภาพ เพื่อให้การทดลองสามารถชี้ประเด็นปัจจัยต่างๆ ที่ส่งผลให้ความแม่นยำเปลี่ยนแปลงได้ จึงต้องมีการเปรียบเทียบผลลัพธ์ที่มาจากการสร้างชุดข้อมูลฝึกสอนด้วยค่า Parameter ที่แตกต่างกัน ทดลองหลายครั้งในหลายแง่มุมเพื่อให้สามารถวิเคราะห์และเปรียบเทียบผลลัพธ์หาข้อสรุปได้ ซึ่งการทดลองในแต่ละสมมติฐานจะมีการดำเนินงานที่คล้ายคลึงกัน ดังนี้

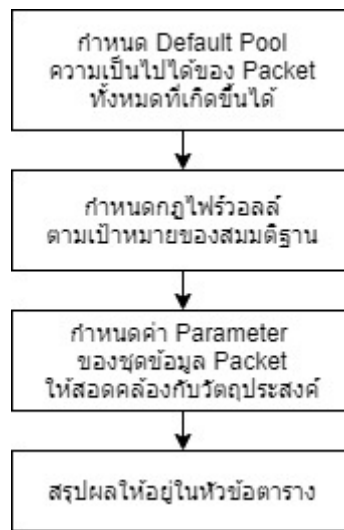


รูปที่ 3.1 Block diagram วัฏจักรการพัฒนาสร้างชุดข้อมูลฝึกสอน

จากรูปภาพ Block Diagram ข้างต้น สามารถแบ่งกระบวนการทำงานออกเป็นขั้นตอนได้ 6 ขั้นตอน ดังนี้

- การกำหนดขอบเขตของข้อมูล Data Field ที่จะพิจารณา และการกำหนดกฎของไฟร่วลล์
- การสร้างชุดข้อมูลสำหรับการฝึกสอนโมเดล
- การนำโมเดลไปผ่านการเรียนรู้ด้วยชุดข้อมูลสำหรับฝึกสอน
- การสร้างชุดข้อมูลสำหรับการทดสอบโมเดล
- การนำโมเดลไปประมวลผล ทำนายผลลัพธ์จากชุดข้อมูลสำหรับทดสอบ
- บันทึกผลลัพธ์จากการทดสอบโมเดล

### 3.3.1. ขั้นตอนที่ 1 การกำหนดขอบเขตของ Data Field ที่จะพิจารณา และการกำหนดกฎไฟร์วอลล์



รูปที่ 3.2 Block Diagram การกำหนดขอบเขตของข้อมูลทั้งหมดที่จะศึกษา

เป็นขั้นตอนที่สำคัญสุดของงานวิจัย เป็นการชี้ประเด็นที่จะศึกษาและแนวทางของผลลัพธ์ที่จะเป็น โดยเริ่มจากการทำการทดลองอิงจากงานวิจัยเก่า ทดลองตั้งสมมติฐาน นำไปต่อยอดและสรุปเป็นประเด็นใหม่ที่สามารถพิสูจน์ได้

เงื่อนไขหลักของการวิจัยคือการสร้างชุดข้อมูลฝึกสอนจากกฎของไฟร์วอลล์ เพื่อให้ได้ระบบการทำงานคัดกรองข้อมูล Packet ที่ได้มาตรฐานและเรียนรู้ได้เองอย่างมีประสิทธิภาพ มีความแม่นยำสูง สิ่งที่ต้องทำในส่วนแรกคือการกำหนดขอบเขตความเป็นไปได้ที่ข้อมูลจะสามารถเกิดขึ้นในเครือข่าย และการกำหนดกฎของไฟร์วอลล์เพื่อให้สามารถสร้างชุดข้อมูล Packet ที่จะนำไปฝึกสอนให้กับโมเดล สร้างชุดข้อมูลทดสอบโมเดลที่สามารถเปรียบเทียบความถูกต้องของผลลัพธ์ที่ได้จากโมเดลหลังการเรียนรู้แล้ว

#### 3.3.1.1. การกำหนด Default Pool และ Data Field ที่จะใช้พิจารณา

การกำหนดขอบเขตของ Packet ที่สามารถเกิดขึ้นหรือการกำหนด Default เองเป็นอีกหนึ่งขั้นตอนที่สำคัญ เพื่อลดปัญหาในการใช้ Workload และลดเวลาที่ใช้ในการทดลองของคอมพิวเตอร์ที่มากเกินไปในการคำนวณหา Sample Space เพราะ Packet ที่เกิดขึ้นจริงมีจำนวนมหาศาล แม้มีข้อมูลภายใน Field เพียงชุดเดียวที่แตกต่างกัน ชุดข้อมูลนั้นจะถูกสรุปเหมือนเป็นชุดข้อมูลใหม่ แต่ถึงกระนั้นการลดจำนวน Default จะต้องไม่น้อยเกินไปและยังสามารถสร้างกฎไฟร์วอลล์ที่ใช้ในการทดลองได้

Data Field	ขนาดใน Packet Header (Bit)	ความเป็นไปได้ (N Possible)
Source Address	32	$2^{32}$
Source Mask	32	32
Destination Address	32	$2^{32}$
Destination Mask	32	32
Port	16	$2^{16}$
Protocol	8	$2^8$

ตารางที่ 3.1 ผลลัพธ์ความเป็นไปได้ที่เกิดขึ้นทั้งหมดจาก Data Field ที่กำหนด

Data Field ที่จะใช้พิจารณาแจกแจง Sample Space ของ Possible Packet

- Source Address (32 bits)  
ความเป็นไปได้ทั้งหมดจะขึ้นอยู่กับ Mask ของ Source Address
- Source Mask (32 bits)
- Destination Address (32 bits)  
ความเป็นไปได้ทั้งหมดจะขึ้นอยู่กับ Mask ของ Destination Address
- Destination Mask (32 bits)
- Port (16 bits)  
ความเป็นไปได้ขึ้นอยู่กับจำนวน port ใน pull ที่กำหนดไว้
- Protocol (8 bits)  
ประกอบไปด้วย TCP และ UDP

เมื่อนำมาลองวิเคราะห์หา Packet Possible แม้จะมี Data Field เพียงแค่ 6 Field ก็ยังมีจำนวน  
มากเกินไปที่จะสามารถคำนวณได้ หมายความว่า Sample Space ของชุดข้อมูลจะเท่ากับ

$$2^{32} \times 32 \times 2^{32} \times 32 \times 2^{16} \times 2^8 = 5.7089907708 \times 10^{45}$$

ตัวแปรที่สำคัญคือจำนวน Source Address, Destination Address และจำนวน Port ที่มีมาก  
เกินไป ซึ่งเมื่อลองลดจำนวนลงแล้วค่าจะเปลี่ยนไปอย่างมาก

- IP อยู่ในวง Subnet Mask /16, มีปลายทางเดียว, จำกัด 4 Ports, จำกัด 2 Protocols

$$2^{16} \times 16 \times 1 \times 1 \times 4 \times 2 = 8,388,608$$



จะเห็นได้ว่าจำนวนของ Possible Packet ของ Default เริ่มสามารถคำนวณได้ เห็นภาพรวมของข้อมูลได้ง่ายขึ้นเนื่องจากลดค่าความคลาดเคลื่อนของชุดข้อมูล Packet ลง

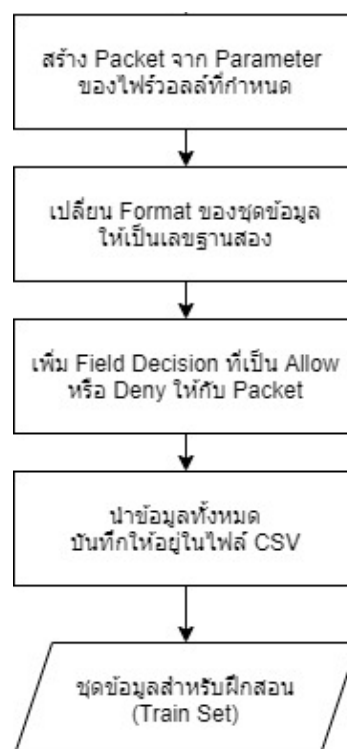
### 3.3.1.2. การกำหนดกฎไฟร์วอลล์สำหรับใช้สร้างชุดข้อมูล

ขั้นตอนต่อมาคือการสร้างกฎของไฟร์วอลล์ ในขั้นตอนนี้จะเป็นการกำหนดกระบวนการทำ Packet Filtering ที่จะเป็นการตัดสินใจว่า ข้อมูล Packet ชุดดังกล่าวจะสามารถถูกตัดสินใจให้ผ่านหรือไม่ ซึ่ง Packet ทุกชุดจะถูกตรวจสอบในทุกกฎของไฟร์วอลล์โดยมี 2 คำสั่งหลัก ได้แก่ “Allow” ปล่อยให้ข้อมูลชุดนั้นเข้าสู่ระบบหรือ “Deny” ไม่ปล่อยให้ข้อมูลชุดนั้นผ่านเข้าสู่ระบบ ค่าในตารางจะเป็น Parameter ที่จำเป็นในการสร้างชุดข้อมูลใน Packet Generator ในขั้นตอนต่อไป

Action	Source Address/Mask	Destination Address/Mask	Port	Protocol
Allow	192.168.0.0/16	201.223.16.1/24	21	TCP
Deny	192.168.0.0/16	201.223.16.1/24	80	TCP
Deny	192.168.0.0/16	201.223.16.1/24	21	UDP
Deny	192.168.0.0/16	201.223.16.1/24	80	UDP

ตารางที่ 3.2 ตัวอย่างการสร้างเงื่อนไขภายในชุดกฎของไฟร์วอลล์

### 3.3.2. ขั้นตอนที่ 2 การสร้างชุดข้อมูลสำหรับการฝึกสอนโมเดล



รูปที่ 3.3 Block Diagram การสร้างชุดข้อมูลฝึกสอนสำหรับโมเดล

ชุดข้อมูลฝึกสอนชุดหนึ่งจะประกอบไปด้วยตัวอย่างข้อมูล Packet ที่ตรงตามเงื่อนไขในแต่ละกฎไฟร์วอลล์ มีวิธีการแบ่งจำนวนตามสมมติฐานที่วางเอาไว้ และจะเพิ่มจำนวนขึ้นไปเรื่อยๆตามการทดลอง

เพื่อให้ชุดข้อมูลฝึกสอนอยู่ในรูปแบบที่โมเดลสามารถใช้งานได้และอยู่ในขอบเขตของงานวิจัย จึงตัดสินใจสร้างชุดข้อมูลฝึกสอนโดยใช้โปรแกรม Packet Generator ที่สร้างขึ้นเอง ชุดข้อมูลฝึกสอนที่ถูกสร้างขึ้นจะถูกจัดระเบียบอยู่ใน Cell ของไฟล์นามสกุล CSV ทำให้ง่ายแก่การดึงข้อมูลกลับมาใช้ต่อในขั้นตอนถัดไป

แต่ก่อนที่จะสร้างชุดข้อมูล Packet นั้นจะต้องทราบความต้องการและจุดประสงค์ของโมเดล ว่าโมเดลดังกล่าวต้องการชุดข้อมูลที่มีความสัมพันธ์และมีจำนวน Input และ Output อย่่างไร การสร้างชุดข้อมูล Packet จะเป็นการสุ่มเลือกจากความเป็นไปได้ทั้งหมดของชุดข้อมูล Packet ทั้งหมด และหลังจากนั้นจะเป็นการเพิ่ม Decision Field เข้าไปในชุดข้อมูล Packet แต่ละชุด เพื่อให้โมเดลนำไปเข้ากระบวนการเรียนรู้ และเปรียบเทียบผลลัพธ์ในขั้นตอนหลังการทดสอบ (Evaluate) ตัดสินจากความแม่นยำในการทำนาย Decision Field ซึ่งจะถูกสร้างอ้างอิงกับกฎของไฟร์วอลล์ในขั้นตอนแรก

### กลไกในการออกแบบชุดข้อมูลฝึกสอน

ชุดข้อมูลที่เราได้ทำการจำลองมาจาก Packet Header และเพื่อแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสมแก่การนำมาประมวลผลได้ จึงมีการเปลี่ยนแปลงรูปแบบและแทนค่าข้อมูลดังกล่าว ดังนี้

- การแทนค่าเป็นเลขฐานสองใน Decision Field
  - Allow แทนค่า เป็น 1
  - Deny แทนค่า เป็น 0
- ข้อมูลอื่นใน Packet Header จะถูกแปลงเป็นเลขฐานสองตามขนาดของ Label นั้นๆ

ชุดข้อมูล Packet ที่สร้างขึ้นเป็นการประยุกต์ใช้วิธีเรียนรู้แบบ Supervised Learning หรือการจับกลุ่มเรียนรู้จากข้อมูลที่มีโครงสร้าง ดังนั้นเพื่อให้ชุดข้อมูลฝึกสอนสามารถใช้งานได้เต็มประสิทธิภาพ ชุดข้อมูลฝึกสอนจะต้องออกแบบให้มีความครอบคลุม ไม่เกิดปัญหา Underfitting หรือ Overfitting

- **Underfitting** คือ การที่โมเดลของเราไม่สามารถทำงานได้ จากการที่ไม่สามารถจัดแนวโน้มของข้อมูลได้ อันเนื่องมาจากโมเดลเราไม่เหมาะสมหรือข้อมูลมีจำนวนน้อยไป กรณีนี้โมเดลมีค่าความเอนเอียงสูง (high bias) ยกตัวอย่างเช่น หากเรานำข้อมูลที่ Train มาลองแล้วได้ความแม่นยำต่ำ เมื่อนำชุดข้อมูลทดสอบมาลองก็จะได้ความแม่นยำต่ำเช่นกัน

- **Overfitting** คือ การที่โมเดลตอบสนองต่อการรบกวน (noise) จำนวนมาก จนเริ่มเรียนจากการรบกวนและรายละเอียดของข้อมูลที่ไม่ถูกต้อง แล้วโมเดลของเราจะไม่เหมาะสมสำหรับการสามารถทำนายข้อมูล เช่น ทำนายข้อมูลที่ไม่เคยมีอย่างผิดพลาดกว่าที่คาดจะเป็นมาก (ล้มเหลวที่จะทำนายข้อมูลได้ถูกต้อง) เพราะมีรายละเอียดและการรบกวนมากเกินไป กรณีนี้โมเดลมีค่าความแปรปรวนของข้อมูลสูง (high variance) ยกตัวอย่างเช่น โมเดลที่พัฒนาขึ้นมีความแม่นยำจากชุดข้อมูลทดสอบมากถึง 99% แต่เมื่อนำชุดข้อมูลทดสอบซึ่งไม่เคยปรากฏเคยในชุดข้อมูลฝึกสอนมาทดสอบ ทำให้ความแม่นยำเหลืออยู่เพียง 40% ปัญหานี้คือ Overfitting

46	deny	192.168.116.116	255.255.0.0	161.246.34.11	255.255.255.0	22	17
47	deny	192.168.180.108	255.255.0.0	161.246.34.11	255.255.255.0	22	17
48	allow	192.168.90.28	255.255.0.0	161.246.34.11	255.255.255.0	22	6
49	allow	192.168.138.145	255.255.0.0	161.246.34.11	255.255.255.0	22	6
50	deny	192.168.16.146	255.255.0.0	161.246.34.11	255.255.255.0	80	6
51	deny	192.168.30.41	255.255.0.0	161.246.34.11	255.255.255.0	80	6
52	deny	192.168.215.79	255.255.0.0	161.246.34.11	255.255.255.0	80	17
53	allow	192.168.242.239	255.255.0.0	161.246.34.11	255.255.255.0	22	6
54	deny	192.168.230.104	255.255.0.0	161.246.34.11	255.255.255.0	80	6
55	allow	192.168.121.255	255.255.0.0	161.246.34.11	255.255.255.0	22	6
56	deny	192.168.224.185	255.255.0.0	161.246.34.11	255.255.255.0	80	6
57	allow	192.168.174.122	255.255.0.0	161.246.34.11	255.255.255.0	22	6
58	allow	192.168.204.76	255.255.0.0	161.246.34.11	255.255.255.0	22	6
59	deny	192.168.181.143	255.255.0.0	161.246.34.11	255.255.255.0	80	17
60	deny	192.168.9.78	255.255.0.0	161.246.34.11	255.255.255.0	80	17
61	allow	192.168.75.191	255.255.0.0	161.246.34.11	255.255.255.0	22	6
62	deny	192.168.140.0	255.255.0.0	161.246.34.11	255.255.255.0	80	17

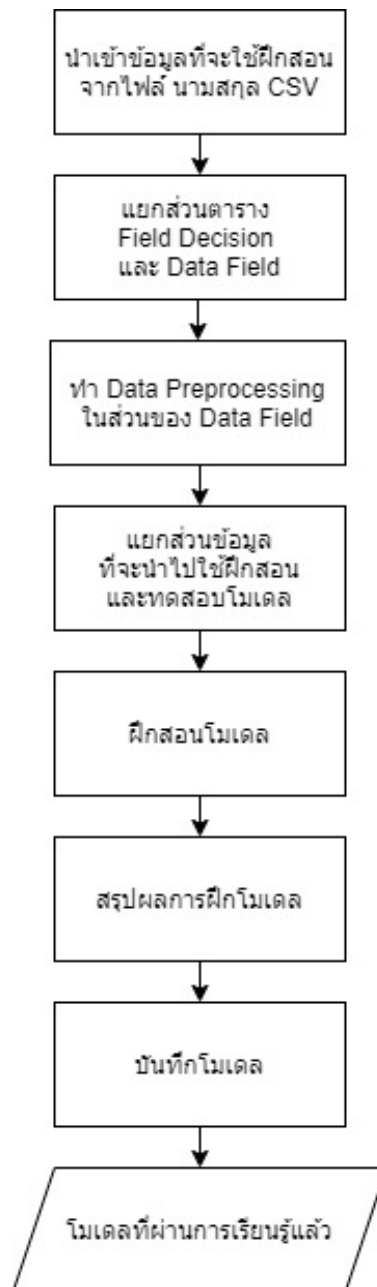
รูปที่ 3.4 ตัวอย่างชุดข้อมูล Data set ที่ถูกสร้างขึ้นเมื่อแสดงผลออกมาเป็น Plain text

```
["Act","src_a1","src_a2","src_a3","src_a4","src_m1","src_m2","src_m3","src_m4","dst_a1","dst_a2","dst_a3","dst_a4","dst_m1","dst_m2","dst_m3","dst_m4",
"1","11000000","10101000","00100011","11110000","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00111111","01011010","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","00001110","11011000","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","01100111","00011001","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","01011001","11110111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","01011001","11110111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","11110000","00010001","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","01011001","11110111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","10100001","10101011","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","11110110","11101111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","10100001","10101011","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","10100110","10101110","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00011110","10001011","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00011111","11100001","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","00101001","01110011","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","01011001","00010000","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","10110101","11001111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00010000","10010101","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00110100","10010111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","10010000","01111101","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","00100110","01110111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","11101001","11010000","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","00010000","11101111","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"0","11000000","10101000","01100110","11011011","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","01000101","11001110","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:
"1","11000000","10101000","01101000","00111110","11111111","11111111","00000000","00000000","10100001","11110110","00100010","00001011","11111111","11:

```

รูปที่ 3.5 ตัวอย่างชุดข้อมูล Data set ที่ถูกสร้างขึ้นเมื่อแสดงผลออกมาเป็น Binary set

### 3.3.3. ขั้นตอนที่ 3 การนำโมเดลไปผ่านการเรียนรู้ด้วยชุดข้อมูลสำหรับฝึกสอน



รูปที่ 3.6 Block Diagram ขั้นตอนการนำโมเดลไปฝึกฝนด้วยชุดข้อมูลฝึกสอน

เป็นขั้นตอนการนำชุดข้อมูลฝึกสอนที่สร้างขึ้นไปประมวลผลผ่านโมเดลให้เกิดการเรียนรู้ โดยขั้นตอนการฝึกโมเดลจะต้องมีการกำหนดค่าพารามิเตอร์และปรับปรุงแก้ไขการประมวลผลหาคำตอบที่ขึ้นอยู่กับขอบเขตของงานหรือข้อมูลที่จะพิจารณา ซึ่งในส่วนนี้เราสามารถหาหลักการได้จากคำแนะนำของผู้พัฒนาโมเดล หรืองานวิจัยที่มีการใช้งานใกล้เคียงกัน โดยมีจุดประสงค์เพื่อพัฒนาให้โมเดลสามารถเรียนรู้ผ่านชุดข้อมูลฝึกสอนได้อย่างมีประสิทธิภาพขึ้นได้

เราได้ตัดสินใจเลือกโมเดลที่มีการเรียนรู้แบบ Sequential Logistic Regression มีฟังก์ชันการประมวลผลแบบ Sigmoid สมการถดถอยที่มีการเรียนรู้ในเชิงคุณภาพหรือเชิงกลุ่ม โดยที่ตัวแปรที่ออกมาเมื่ออยู่ 2 ค่า คือมีค่าเป็น 0 กับ 1 ทำให้รูปแบบการเรียนรู้นี้เหมาะกับการแก้ปัญหาตามโจทย์ Binary Classification Problem ที่คำตอบจะถูกตัดสินใจแบบ Two-Class-Label แบ่งออกเป็น 2 ตัวเลือก ได้แก่ Allow หรือ Deny ตามที่เรากำหนดไว้ตั้งแต่แรกภายในการทดสอบ

### ข้อมูลการตั้งค่าที่สำคัญภายในโมเดล

- รูปแบบการเรียนรู้: Sequential Logistic Regression
- ฟังก์ชันการประมวลผล: Sigmoid  $f(x) = 1/(1 + \exp(-x))$
- เครื่องมือเสริมประสิทธิภาพในการประมวลผล: Adam Optimizer

กระบวนการทำงานในขั้นตอนนี้ จะเป็นการแยกส่วนข้อมูลที่จะใช้พิจารณาแยกกันในไฟล์นามสกุล CSV ที่สร้างจากขั้นตอนที่แล้ว โดยแบ่งออกเป็นชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบ สำหรับการสรุปผลการเรียนรู้ในอัตราส่วนที่ได้จาก Rule of Thumb คือ 8:2 และแบ่งชุดข้อมูลดังกล่าวออกอีก ได้แก่

- ชุดข้อมูลฝึกสอน ที่ประกอบไปด้วย Data Field ภายใน Packet ทั้งหมด
- ชุดข้อมูลฝึกสอน ที่ประกอบไปด้วย Decision ที่เป็นผลลัพธ์ตัดสินใจว่าจะปล่อยผ่าน
- ชุดข้อมูลทดสอบ ที่ประกอบไปด้วย Data Field ภายใน Packet ทั้งหมด
- ชุดข้อมูลทดสอบ ที่ประกอบไปด้วย Decision ที่เป็นผลลัพธ์ตัดสินใจว่าจะปล่อยผ่าน

นำข้อมูลข้างต้นมาทำ Data Preprocessing หรือการจัดข้อมูลชุดให้อยู่ในรูป Matrix เปลี่ยนค่าภายในในกลายเป็นค่าถ่วงน้ำหนัก เป็นค่าที่โมเดลจะนำไปเรียนรู้ต่อและหาค่าความสัมพันธ์ว่าชุดข้อมูลดังกล่าวจะถูกตัดสินใจว่าเป็น Allow หรือ Deny โดยชุดข้อมูลที่จะต้องนำไปทำ Data Preprocessing ได้แก่

- ชุดข้อมูลฝึกสอน ที่ประกอบไปด้วย Data Field ภายใน Packet ทั้งหมด
- ชุดข้อมูลทดสอบ ที่ประกอบไปด้วย Data Field ภายใน Packet ทั้งหมด

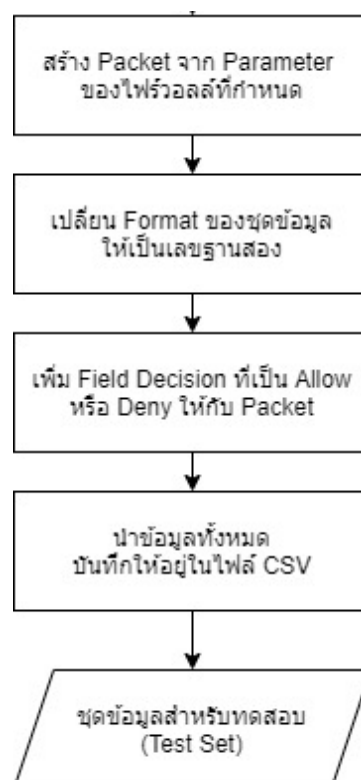
## การออกแบบ MLP Architecture ในงานวิจัย

โครงสร้างของชุดข้อมูลฝึกสอนมีผลอย่างมากในการเลือกโมเดลที่จะนำมาใช้ เนื่องจากข้อมูล Packet ของเราทั้งหมดจะอยู่ในรูปแบบเลขฐานสอง ทำให้มีหน่วยตั้งเป็นค่า Bit ซึ่งเมื่อถ้าหากนำไปอ้างอิงกับบทประพันธ์ที่ผ่านมาข้างต้น จะได้จำนวน Neuron กับจำนวน Hidden Layer ที่ต้องการได้

- **Input:** Source Address + Mask + Destination Address + Mask + Port + Protocol  
 $= 32+32+32+32+16+8 = 152$  Neurons
- **Output:** 2 Neurons (Allow, Deny)
- **Hidden Layer:** 3 Layers

กระบวนการเรียนรู้ในขั้นตอนนี้จะหยุดลงเมื่อข้อผิดพลาดในชุดการตรวจสอบความถูกต้องคงที่ {เมื่อค่าความคลาดเคลื่อนระหว่างข้อผิดพลาดก่อนหน้าและปัจจุบันหารด้วยข้อผิดพลาดปัจจุบันต่ำกว่าค่าคงที่เล็กน้อย ในกรณีของเราค่าคงที่นี้ถูกตั้งค่าเป็น 0.1%

### 3.3.4. ขั้นตอนที่ 4 การสร้างชุดข้อมูลสำหรับการทดสอบโมเดล



รูปที่ 3.7 Block Diagram การสร้างชุดข้อมูลทดสอบ โมเดล

### หลักการออกแบบชุดข้อมูลทดสอบ

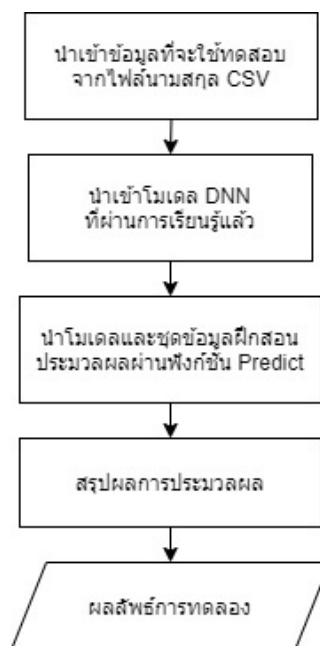
ในการสร้างชุดข้อมูลทดสอบที่สามารถวัดผลความแม่นยำของโมเดลจากการทดลองได้ ในการออกแบบนั้นถือว่ามีความท้าทายในระดับหนึ่ง เพราะมีประเด็นสำคัญที่จำเป็นต้องพิจารณาดังต่อไปนี้

- จะทราบได้อย่างไรว่า โมเดลสามารถทำนายผลลัพธ์ได้ดีในทุกกฎไฟร่วอลล์
- จะทราบได้อย่างไรว่า โมเดลติดปัญหา Underfitting หรือ Overfitting

เราได้ทำการสร้างชุดข้อมูลทดสอบ และแบ่งจำนวนชุดข้อมูลออกเป็นจำนวนที่เท่าๆกัน ในแต่ละเงื่อนไขกฎของไฟร่วอลล์ เพื่อให้สามารถทราบได้ว่าภาพรวมที่โมเดลทำนายผลมานั้นให้ ความถูกต้องแม่นยำเป็นอย่างไร ซึ่งถ้าหากไฟร่วอลล์นั้นสามารถทำนายผลได้เพียงบางเงื่อนไข ความแม่นยำที่ได้จากชุดข้อมูลทดสอบเดียวกันแต่โมเดลต่างกันจะต้องเห็นผลลัพธ์ที่สามารถสังเกต ได้อย่างแน่นอน

ในความเป็นจริงแล้ว เพื่อให้มีการทดสอบและวิเคราะห์ได้ดียิ่งขึ้น อาจต้องสร้างชุดข้อมูล ทดสอบหลายๆแบบที่มีความแตกต่างกัน เพื่อให้สามารถจับประเด็นสำคัญหรือปัญหาที่เกิดขึ้นจาก โมเดลได้ เช่น การทดสอบว่าโมเดลมีปัญหา Overfitting หรือมีวิธีการตรวจสอบที่ดีหรือไม่

#### 3.3.5. ขั้นตอนที่ 5 การนำโมเดลไปประมวลผล ทำนายผลลัพธ์จากชุดข้อมูลสำหรับทดสอบ



รูปที่ 3.8 Block Diagram การนำโมเดลไปประมวลผลหรือ Evaluate

เป็นขั้นตอนทดสอบ (Evaluate) เพื่อทำนายความแม่นยำของโมเดลที่ผ่านการเรียนรู้แล้ว โดยใช้ข้อมูลทดสอบอีกชุดหนึ่ง ในส่วนนี้จะใช้โปรแกรม Compare Engine ที่เขียนขึ้นเอง เริ่มจากการนำเข้าโมเดลที่ผ่านการเรียนรู้แล้วจากขั้นตอนที่ 3 นำไปคาดเดาชุดข้อมูลทดสอบจากขั้นตอนที่ 4 ตัวโปรแกรมจะทำการแยกส่วนชุดข้อมูล CSV เป็นส่วนของข้อมูลและผลลัพธ์เช่นเดียวกันกับตอนฝึกโมเดล ด้วยฟังก์ชัน model.predict ของ Keras จะสามารถทำนายผลด้วยโมเดลได้ทันทีว่าจากชุดข้อมูล Packet ทดสอบนั้น ให้ผลลัพธ์ Allow หรือ Deny ซึ่งผลลัพธ์สุดท้ายจะเป็นสรุปในการหาความแม่นยำของโมเดลนั้นตาม Reference Variant Set ดังนี้

<p>True Positive (TP) Correct variant allele or position call.</p>	<p>False Positive (FP) Incorrect variant allele or position call.</p>
<p>False Negative (FN) Incorrect reference genotype or no call.</p>	<p>True Negative (TN) Correct reference genotype or no call.</p>

รูปที่ 3.9 Reference Set ในการวิเคราะห์ความถูกต้องของโมเดล

Reference Variant Set เป็น Matrix ที่ใช้ในการอ้างอิงในการหาข้อสรุปของโมเดลว่ามีความแม่นยำหรือไม่ อย่างไร ซึ่งมักถูกใช้กับโมเดลที่มีการเรียนรู้และแก้ปัญหาในการแบ่งกลุ่ม โดยผลลัพธ์ที่ได้จะประกอบไปทั้งหมด 4 รูปแบบ ได้แก่

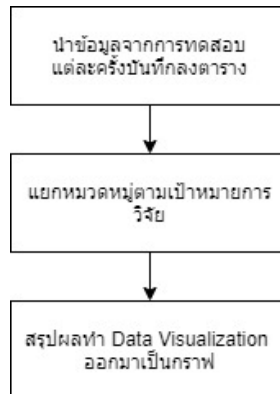
- True Positive  
โมเดลอนุญาตให้ข้อมูลผ่านตรงตามกฎของไฟร์วอลล์ หรือให้ Allow ถูกต้อง
- True Negative  
โมเดลไม่อนุญาตให้ข้อมูลผ่านตรงตามกฎของไฟร์วอลล์ หรือให้ Deny ถูกต้อง
- False Positive  
โมเดลอนุญาตให้ข้อมูลผ่านไม่ตรงตามกฎของไฟร์วอลล์ หรือให้ Allow ผิดพลาด
- False Negative  
โมเดลไม่อนุญาตให้ข้อมูลผ่านไม่ตรงตามกฎของไฟร์วอลล์ หรือให้ Deny ผิดพลาด



ซึ่งผลลัพธ์ที่ได้จะเป็นไปตามสูตร

$$\text{ความแม่นยำ (Accuracy)} = \text{SUM}(\text{TP}, \text{TN}) / \text{SUM}(\text{TP}, \text{TN}, \text{FP}, \text{FN})$$

### 3.3.6. ขั้นตอนที่ 6 บันทึกผลลัพธ์จากการทดสอบโมเดล



รูปที่ 3.10 Block Diagram ขั้นตอนการนำผลลัพธ์มาบันทึกผล

การหาวิธีการที่สามารถทำให้ชุดข้อมูลฝึกสอนสามารถสอนโมเดลได้อย่างมีประสิทธิภาพ เราจำเป็นต้องนำผลลัพธ์ของการทดสอบในแต่ละครั้งของการทดลองมาบันทึกผล แล้วสรุปให้อยู่ในรูปกราฟเปรียบเทียบที่ประกอบไปด้วยผลลัพธ์จากการทดลองภายใต้สภาพแวดล้อมเดียวกัน เพื่อหาว่าผลลัพธ์ออกมาตรงตามสมมติฐานหรือมีความสัมพันธ์กันในแต่ละตัวแปรอย่างไรบ้าง

	Sample per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default											
With Default											

รูปที่ 3.11 ตัวอย่างของตารางที่จะนำมาบันทึกผลลัพธ์การทดลอง

## บทที่ 4

### ผลการดำเนินงานวิจัย

การทดลองจะเป็นไปตามวัฏจักรการดำเนินงานวิจัยข้างต้น โดยชุดข้อมูลฝึกสอนที่ทำการพัฒนาขึ้นมีรูปแบบโครงสร้างจำลองมาจาก Packet Header และสร้างขึ้นผ่าน โปรแกรม Packet Generator ที่ออกแบบขึ้นเอง ชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบจะมีการออกแบบให้มีความแตกต่างกันตามสมมติฐานที่กำหนด สังเกตกระบวนการทำงานของโมเดล และรูปแบบความสัมพันธ์ของตัวแปรที่ได้หลังโมเดลทำการเรียนรู้และประมวลผล และทำการสรุปผลลัพธ์ที่ได้หลังเสร็จสิ้นการทดลอง

#### การออกแบบเงื่อนไขของชุดข้อมูล

ค่าความเป็นไปได้ทั้งหมดที่สามารถเกิดขึ้นได้ในแต่ละส่วนของข้อมูล มีดังนี้

Data Field	ตัวแปรที่ใช้	จำนวนความเป็นไปได้ทั้งหมด
Source Address	Subnet 192.168.0.0/16	65534
Source Mask	ขึ้นอยู่กับ Source Address	1
Destination Address	161.246.34.11	1
Destination Mask	/32	1
Port	22, 80	2
Protocol	TCP, UDP	2

ตารางที่ 4.1 ตารางการจำแนกความเป็นไปได้ของแต่ละ Data Field

ดังนั้น ข้อมูล Packet ที่เกิดขึ้นได้ทั้งหมด จะเท่ากับ  $65534 * 1 * 1 * 1 * 2 * 2 = 262,016$

#### การจำแนกชุดกฎไฟร์วอลล์ที่จะทำการทดสอบ

ชุดเงื่อนไขทั้งหมดที่สร้างขึ้นจากกฎไฟร์วอลล์	จำนวนข้อมูลที่สามารถเกิดขึ้นตรงตามเงื่อนไขของไฟร์วอลล์ที่กำหนด
Rule set ที่ 1 <ul style="list-style-type: none"><li>allow 192.168.0.0/16 to 161.246.34.11/24 port 80 tcp</li><li>deny 192.168.128.0/18 to 161.246.34.11/24 port 22 udp</li></ul>	$65,534 + 16,382$ $= 81,916$

<p>Rule set ที่ 2</p> <ul style="list-style-type: none"> <li>● allow 192.168.0.0/16 to 161.246.34.11/24 port 80 tcp</li> <li>● deny 192.168.128.0/18 to 161.246.34.11/24 port 22 udp</li> <li>● allow 192.168.64.0/24 to 161.246.34.11/24 port 22 tcp</li> <li>● deny 192.168.64.0/24 to 161.246.34.11/24 port 80 udp</li> </ul>	$65,534 + 16,382 + 254 + 254 = 82,424$
<p>Rule set ที่ 3</p> <ul style="list-style-type: none"> <li>● allow 192.168.0.0/16 to 161.246.34.11/24 port 80 tcp</li> <li>● deny 192.168.128.0/18 to 161.246.34.11/24 port 22 udp</li> <li>● allow 192.168.64.0/24 to 161.246.34.11/24 port 22 tcp</li> <li>● deny 192.168.64.0/24 to 161.246.34.11/24 port 80 udp</li> <li>● allow 192.168.192.0/18 to 161.246.34.11/24 port 22 udp</li> <li>● allow 192.168.128.0/18 to 161.246.34.11/24 port 22 tcp</li> </ul>	$65,534 + 16,382 + 254 + 254 + 16,382 + 16,382 = 115,188$

ตารางที่ 4.2 ตารางการจำแนกความเป็นไปได้ของแต่ละกฎไฟร์วอลล์

**การนำ Default Rule มาเป็นส่วนหนึ่งของกฎไฟร์วอลล์เพื่อสร้างชุดข้อมูลฝึกสอน**

นอกจากกฎไฟร์วอลล์ที่กำหนดขึ้นทั่วไป ยังมีกฎของ Default Rule ซึ่งจำเป็นต้องพิจารณาแยกเป็นกรณีพิเศษ เนื่องจากจำนวนความเป็นไปได้ของข้อมูลของกฎไฟร์วอลล์ที่มีการกำหนด ทำให้การทดสอบแบ่งออกเป็น 2 แบบ ได้แก่ With Default Rule และ Without Default Rule ซึ่งเราได้ตั้ง Default Rule เป็น Deny any หรือ Deny ทุกข้อมูลทีนอกเหนือจากไฟร์วอลล์ที่เรากำหนดไว้

**การออกแบบชุดข้อมูลฝึกสอนในสมมติฐาน**

ประกอบไปด้วย 2 ชุดข้อมูลฝึกสอนใน 1 เซต โดยประกอบไปด้วยชุดข้อมูลฝึกสอนที่มีประเด็นการนำ Default Rule มาใช้ และชุดข้อมูลฝึกสอนที่ไม่มีการนำ Default Rule มาใช้ในการสร้าง โดยมีตัวแปรสำคัญในการสร้างชุดข้อมูลฝึกสอนเพื่ออิงตามประเด็นศึกษาในสมมติฐาน ดังนี้

- จำนวนและเงื่อนไขของแต่ละกฎไฟร์วอลล์ที่ใช้ภายใน Rule set
- จำนวนของ Packet ของแต่ละกฎไฟร์วอลล์ที่จะนำเข้าระบบ
- การนำประเด็น Default Rule มาใช้ด้วย ประกอบด้วย With Default และ Without Default

## การออกแบบชุดข้อมูลทดสอบ

ประกอบไปด้วย 2 ชุดข้อมูลเช่นเดียวกับชุดฝึกสอน โดยประกอบไปด้วยชุดข้อมูลฝึกสอนที่มีประเด็นการนำ Default Rule มาใช้ และ ไม่มีการนำ Default Rule มาใช้ในการสร้าง

ผลลัพธ์ที่คาดว่าจะต้องเปลี่ยนแปลงไปตามการทดสอบแต่ละครั้ง

- เวลาที่โมเดลใช้ในการเรียนรู้จากชุดข้อมูลฝึกสอน หรือ Training
- เวลาที่โมเดลใช้ในการตัดสินใจจากชุดข้อมูลทดสอบ หรือ Predict
- ค่าความแม่นยำโดยรวม หรือ Accuracy
- อัตราความผิดพลาดที่อ้างอิงจาก Reference Variant Set

### 4.1 สมมติฐานการทดลองที่ 1

ในสมมติฐานการทดลองที่ 1 เป็นการทดลองใช้ชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบที่สร้างขึ้น และเพื่อเป็นการพิสูจน์ว่าโมเดลสามารถประยุกต์ใช้ในงานวิจัยได้จริง มีหลักการทำงาน และผลลัพธ์ที่คล้ายคลึงกับปัญญาประดิษฐ์ที่พบได้ทั่วไป โดยวางสมมติฐานเบื้องต้นไว้ ดังนี้

- โมเดลจะสามารถเรียนรู้จากชุดข้อมูลฝึกสอนที่สร้างจากกฎของไฟร์วอลล์และสามารถทำนายผลลัพธ์ได้
- เมื่อโมเดลเรียนรู้จากชุดข้อมูลฝึกสอนที่มีจำนวนมากขึ้นในแต่ละกฎไฟร์วอลล์ โมเดลจะสามารถทำนายผลลัพธ์ได้แม่นยำมากขึ้น
- โมเดลเมื่อมีการเรียนรู้ถึงจุดๆหนึ่งจะไม่สามารถเพิ่มความแม่นยำในการทำนายผลลัพธ์ได้อีก
- โมเดลจะใช้เวลาในการทดสอบประมวลผลข้อมูลเท่าเดิม แม้จะผ่านการเรียนรู้จากชุดข้อมูลฝึกสอนที่มีจำนวนต่างกัน

#### 4.1.1. หลักการออกแบบชุดข้อมูลฝึกสอน

ชุดข้อมูลฝึกสอนในแต่ละกฎไฟร์วอลล์ จะมีจำนวน N Sample เท่ากันทั้งหมด โดยจำนวนที่ของชุดข้อมูลฝึกสอนที่ใช้ทดสอบ ประกอบไปด้วย 10, 100, 300, 600, 1,000, 3,000, 6,000, 10,000 ในแต่ละกฎไฟร์วอลล์

#### 4.1.2. ผลลัพธ์ที่ได้จากการทดลอง

##### 4.1.2.1. ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 1 (2 กฎไฟร์วอลล์)

	Sample per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	10	20	5.7363	0.8397	100.00%	1.86736	73.32%	20000	9327	10673	0
	100	200	5.7315	1.9852	100.00%	1.64584	78.64%	20000	11455	8545	0
	300	600	5.6906	4.7394	100.00%	1.67258	76.77%	20000	10707	9293	0
	600	1,200	6.5325	8.1221	100.00%	1.65713	85.16%	20000	14064	5936	0
	1,000	2,000	6.5526	12.8895	100.00%	1.95779	76.80%	20000	10718	9282	0
	3,000	6,000	5.9349	37.3558	100.00%	1.61391	80.23%	20000	12090	7910	0
	6,000	12,000	6.1347	55.0237	100.00%	1.65309	79.88%	20000	11950	8050	0
	10,000	20,000	6.6183	95.47293	100.00%	1.62671	80.07%	20000	12029	7971	0
With Default	10	30	6.3698	0.9225	90.00%	1.72755	70.82%	15048	13278	6722	4952
	100	300	6.8951	2.9970	88.00%	1.70799	86.87%	20000	14747	5253	0
	300	900	7.8927	6.6133	87.44%	1.69338	86.67%	20000	14669	5331	0
	600	1,800	9.2233	11.8236	91.22%	1.81215	90.66%	19689	16574	3426	311
	1,000	3,000	12.1462	18.6338	98.50%	1.70495	98.38%	20000	19352	648	0
	3,000	9,000	23.5967	55.4014	100.00%	1.72629	100.00%	20000	20000	0	0
	6,000	18,000	40.0096	109.0460	100.00%	1.69623	100.00%	20000	20000	0	0
	10,000	30,000	68.4731	177.2943	100.00%	1.85090	100.00%	20000	20000	0	0

ตารางที่ 4.3 ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 1 (2 กฎ)

##### 4.1.2.2. ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 2 (4 กฎไฟร์วอลล์)

	Sample per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	10	40	5.2958	1.1594	70.00%	1.853798	63.30%	15178	10140	9860	4822
	100	400	5.7217	4.5659	76.33%	2.014582	73.23%	20000	9293	10707	0
	300	1,200	5.4175	11.8386	84.83%	1.715588	72.82%	20000	9128	10872	0
	600	2,400	5.1841	22.3334	79.11%	2.027679	70.82%	18642	9684	10316	1358
	1,000	4,000	5.6130	35.9608	66.67%	2.080458	70.65%	17483	10775	9225	2517
	3,000	12,000	5.7068	106.9028	66.67%	1.771398	69.70%	16371	11507	8493	3629
	6,000	24,000	6.6372	163.6855	66.67%	1.800605	68.36%	9982	17362	2638	10018
	10,000	40,000	6.6303	271.9477	66.67%	1.72514	76.94%	20000	10775	9225	0
With Default	10	50	5.3566	1.1049	68.00%	1.795341	66.62%	14972	11676	8324	5028
	100	500	6.0489	3.8215	63.60%	1.876988888	66.98%	16195	10598	9402	3805
	300	1,500	7.6283	9.7706	77.47%	1.852022648	76.52%	20000	10607	9393	0
	600	3,000	9.1046	18.7983	95.03%	1.685086727	89.12%	17497	18152	1848	2503
	1,000	5,000	12.1534	31.5166	67.84%	1.718641996	78.20%	20000	11279	8721	0
	3,000	15,000	24.1351	92.0530	69.82%	1.737370491	74.96%	9982	20000	0	10018
	6,000	30,000	43.9188	185.1592	65.53%	1.889707565	73.23%	20000	9293	10707	0
	10,000	50,000	58.0737	296.8121	67.21%	1.688281775	76.53%	20000	10613	9387	0

ตารางที่ 4.4 ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 2 (4 กฎ)

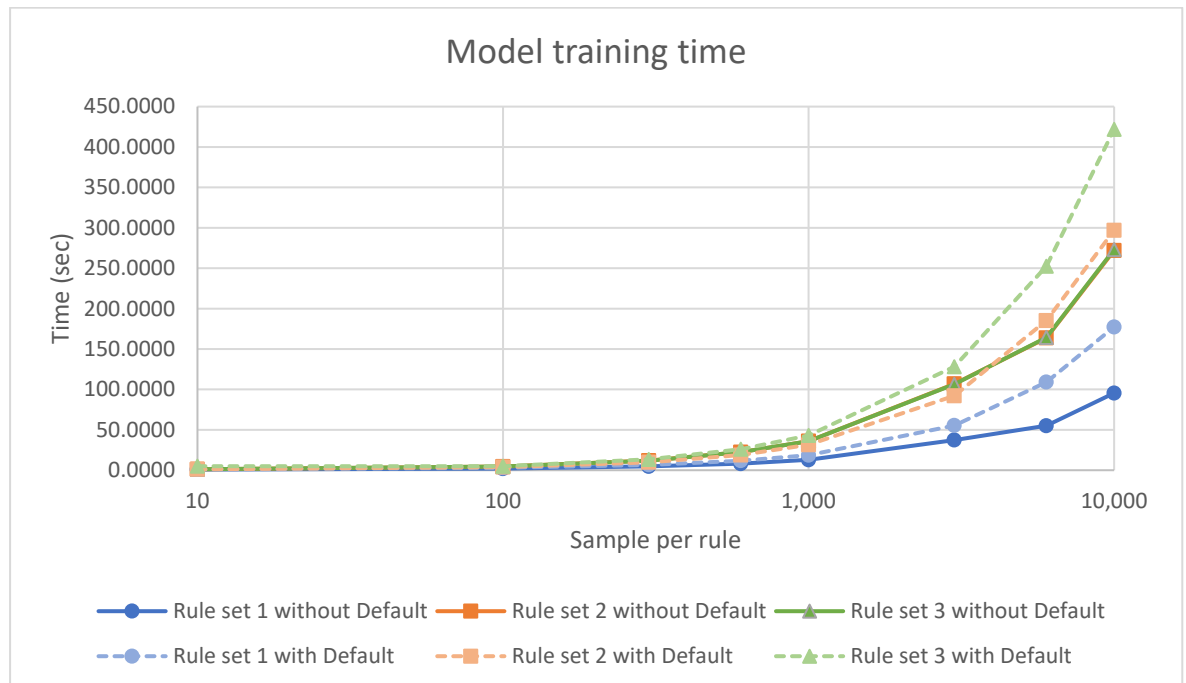
#### 4.1.2.3. ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 3 (6 กฎไฟร์วอลล์)

	Sample per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	10	60	5.8085	1.2054	68.33%	1.66634798	57.59%	12535	10501	9499	7465
	100	600	5.7885	4.5129	78.17%	1.746937037	55.55%	12039	10181	9819	7961
	300	1,800	6.5156	11.7211	88.50%	1.727326393	63.14%	17640	7614	12386	2360
	600	3,600	6.6043	22.4303	96.36%	2.085625	71.35%	18282	10256	9744	1718
	1,000	6,000	7.0093	35.9000	100.00%	1.712445974	74.56%	20000	9823	10177	0
	3,000	18,000	7.2327	106.3793	100.00%	1.824865818	75.69%	20000	10276	9724	0
	6,000	36,000	7.8331	164.2635	100.00%	1.695466518	73.98%	20000	9591	10409	0
	10,000	60,000	8.9511	273.1400	70.85%	1.712440729	50.96%	20000	382	19618	0
With Default	10	70	5.3128	4.9957	71.43%	1.743295193	53.50%	10961	10440	9560	9039
	100	700	6.4049	4.9825	59.86%	1.729964256	63.52%	11082	14326	5674	8918
	300	2,100	9.6961	13.4444	70.00%	1.721653461	71.00%	10740	17659	2341	9260
	600	4,200	12.1775	25.9940	69.86%	1.837330818	74.77%	12814	17092	2908	7186
	1,000	7,000	18.4346	43.0301	58.70%	1.714560986	53.29%	20000	1315	18685	0
	3,000	21,000	39.6620	128.1276	57.86%	1.699449778	51.63%	20000	650	19350	0
	6,000	42,000	77.2631	252.4501	59.21%	1.683763266	53.99%	20000	1595	18405	0
	10,000	70,000	116.6793	421.8212	60.30%	1.688794374	56.47%	20000	2587	17413	0

ตารางที่ 4.5 ตารางผลการทดลองแบบ N Sample เงื่อนไข Rule set ที่ 3 (6 กฎ)

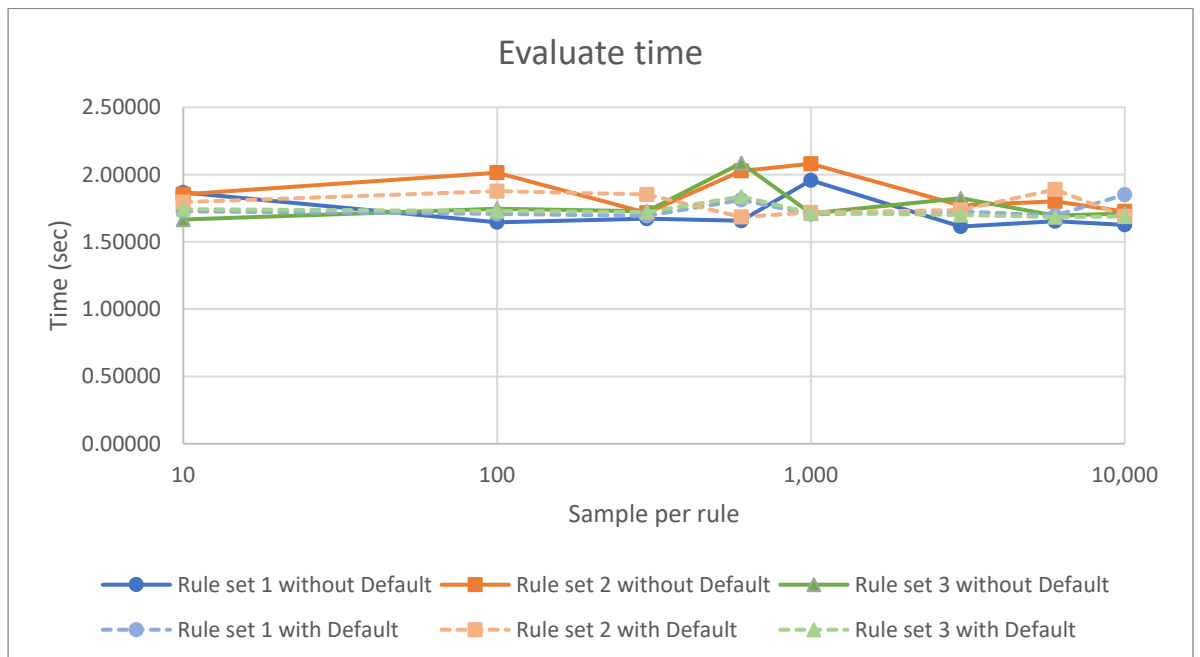
#### 4.1.3. กราฟผลลัพธ์การทดลองแบบ N Sample

##### 4.1.3.1. กราฟผลลัพธ์ เวลาในการฝึกสอนโมเดล : ชุดข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



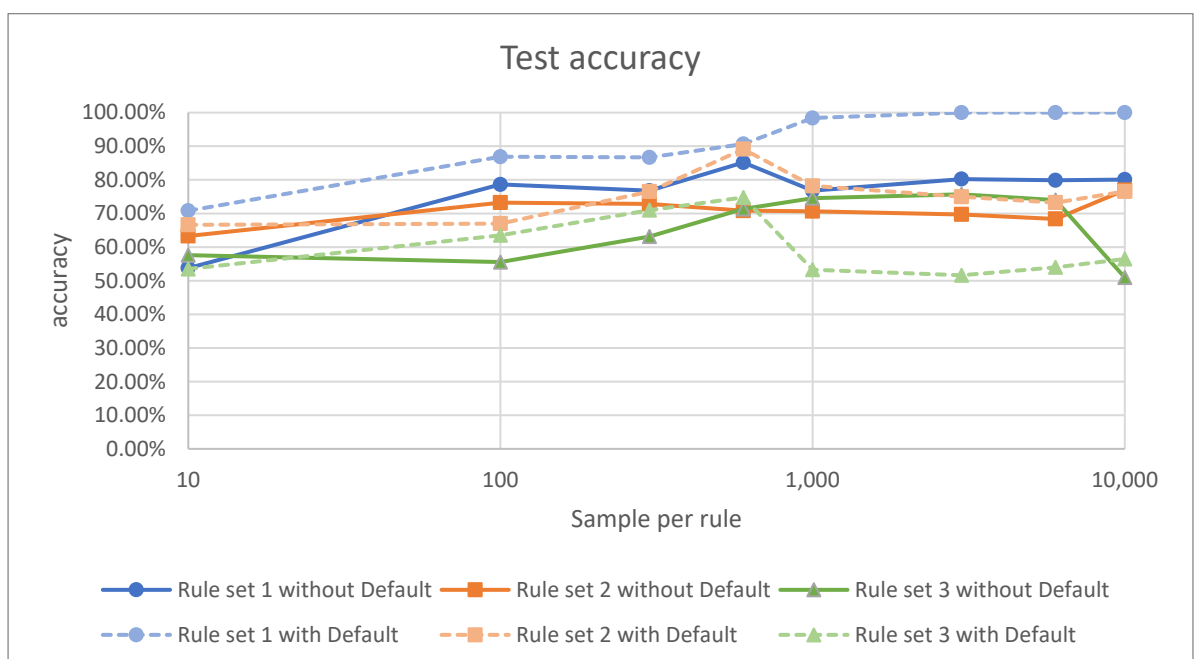
รูปที่ 4.1 กราฟเวลาในการฝึกโมเดล : ชุดข้อมูลฝึกสอนต่อ 1 กฎ (N Sample)

#### 4.1.3.2. กราฟผลลัพธ์ เวลาในการทำนายชุดทดสอบ : จำนวนชุดข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



รูปที่ 4.2 กราฟเวลาทำนายข้อมูลทดสอบ : จำนวนชุดฝึกสอนต่อ 1 กฎ (N Sample)

#### 4.1.3.3. กราฟผลลัพธ์ ความแม่นยำในการประมวลผล : จำนวนชุดข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



รูปที่ 4.3 กราฟความแม่นยำในการประมวลผล : จำนวนชุดฝึกสอนต่อ 1 กฎ (N Sample)

## 4.2 สมมติฐานการทดลองที่ 2

จากสมมติฐานแรกจะเห็นได้ว่า ความเป็นไปได้ของชุดข้อมูลฝึกสอนในแต่ละกฎไฟร์วอลล์มีจำนวนไม่เท่ากัน ดังนั้นในสมมติฐานนี้จึงเป็นการตั้งข้อสันนิษฐานว่า ถ้าหากตั้งเงื่อนไขให้กฎไฟร์วอลล์แต่ละกฎได้รับจำนวนชุดข้อมูลฝึกสอนไม่เท่ากัน จะส่งผลต่อความแม่นยำของโมเดลอย่างไร มีการพัฒนาโมเดลในทางที่ดีขึ้นหรือแย่ลงอย่างไร

### 4.2.1. หลักการออกแบบชุดข้อมูลฝึกสอน

สมมติฐานนี้จะเป็นการใช้อัตราส่วนมาเป็นหลักเกณฑ์ในการแบ่งจำนวนชุดข้อมูลฝึกสอนที่แต่ละกฎจะได้รับ โดยจำนวนของข้อมูลฝึกสอนที่ใช้ทดสอบ เพิ่มขึ้นด้วยอัตราส่วน Ratio ที่เท่าๆกัน โดยอัตราส่วนที่ใช้ประกอบไปด้วย 0.01, 0.03, 0.05, 0.07, 0.09, 0.11, 0.13, 0.15 โดยมีหน่วยเป็นจำนวนชุดข้อมูลฝึกสอนที่ใช้ในแต่ละกฎไฟร์วอลล์ต่อจำนวนความเป็นไปได้ทั้งหมดของชุดฝึกสอนในกฎไฟร์วอลล์นั้นๆ ยกตัวอย่างเช่น เงื่อนไขของกฎของไฟล์วอลล์หนึ่งมีจำนวนความเป็นไปได้คือ 16,382 ความเป็นไปได้ ชุดข้อมูลฝึกสอนที่กฎไฟร์วอลล์นั้นจะได้รับหากมีอัตราส่วน Ratio ที่ 0.01 คือ 163 แพ็คเกต

### 4.2.2. ผลลัพธ์ที่ได้จากการทดลอง

#### 4.2.2.1. ตารางผลการทดลองแบบอัตราส่วน Ratio เงื่อนไข Rule set ที่ 1 (2 กฎไฟร์วอลล์)

	Ratio per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	0.01	818	5.0650	5.4139	100.00%	2.23602	76.61%	20000	10642	9358	0
	0.03	2,457	5.1566	14.9142	100.00%	2.13728	76.80%	20000	10718	9282	0
	0.05	4,095	5.1948	24.6296	100.00%	1.89453	76.80%	20000	9282	9282	0
	0.07	5,733	4.9379	32.5826	100.00%	1.66133	75.01%	20000	10005	9995	0
	0.09	7,372	5.2300	41.6751	100.00%	1.97772	73.32%	20000	9327	10673	0
	0.11	9,010	5.1073	50.6292	100.00%	1.96472	83.95%	20000	13580	6420	0
	0.13	10,648	5.4003	59.7547	100.00%	2.14677	80.07%	20000	12029	7971	0
	0.15	12,287	5.5851	69.13035	100.00%	1.91364	73.32%	20000	9327	10673	0
With Default	0.01	3,439	33.1170	19.7218	100.00%	2.05126	100.00%	20000	20000	0	0
	0.03	10,321	69.2406	57.1022	100.00%	2.06812	100.00%	20000	20000	0	0
	0.05	17,201	140.7002	96.0926	100.00%	2.02230	100.00%	20000	20000	0	0
	0.07	24,082	139.7293	133.8839	100.00%	1.90665	100.00%	20000	20000	0	0
	0.09	30,964	222.5676	172.3245	81.28%	2.04455	50.71%	285	20000	0	19715
	0.11	37,844	221.8128	209.6226	100.00%	2.01761	100.00%	20000	20000	0	0
	0.13	44,725	282.9598	246.3675	100.00%	1.96987	100.00%	20000	20000	0	0
	0.15	51,607	299.4318	286.4131	100.00%	1.83328	100.00%	20000	20000	0	0

ตารางที่ 4.6 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 1 (2 กฎ)



#### 4.2.2.2. ตารางผลการทดลองแบบอัตราส่วน Ratio เงื่อนไข Rule set ที่ 2 (4 กฎไฟร์วอลล์)

	Ratio per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	0.01	822	6.8078	7.0741	99.76%	1.637415648	79.46%	19884	11899	8101	116
	0.03	2,471	6.7609	21.1054	94.98%	1.69797492	77.24%	18610	12285	7715	1390
	0.05	4,119	7.4316	30.3708	99.71%	1.638393402	74.88%	20000	9952	10048	0
	0.07	5,767	6.8587	39.2830	99.71%	1.633202076	75.64%	20000	10254	9746	0
	0.09	7,416	7.1668	47.3943	99.70%	1.65933919	73.23%	20000	9293	10707	0
	0.11	9,064	6.4777	60.5878	99.70%	1.978661299	75.55%	20000	10219	9781	0
	0.13	10,714	7.2456	64.6513	99.69%	1.650335073	73.23%	20000	9293	10707	0
	0.15	12,363	6.9624	72.1588	99.69%	1.637886763	76.10%	20000	10439	9561	0
With Default	0.01	3,443	29.9887	20.3163	92.71%	1.79810524	89.87%	16654	19294	706	3346
	0.03	10,335	63.0633	59.1839	80.91%	1.735456944	50.00%	0	20000	0	20000
	0.05	17,225	116.7934	98.4386	81.50%	1.714803696	51.53%	612	20000	0	19388
	0.07	24,116	140.1712	133.7606	85.46%	1.686157227	62.10%	4839	20000	0	15161
	0.09	31,008	188.0968	132.4159	80.91%	1.716438055	50.00%	0	20000	0	20000
	0.11	37,898	217.8810	213.0778	80.91%	1.898934364	50.00%	0	20000	0	20000
	0.13	44,791	273.9160	248.2306	80.91%	1.734778404	50.00%	0	20000	0	20000
	0.15	51,683	294.2466	289.4282	80.91%	1.747563124	50.00%	0	20000	0	20000

ตารางที่ 4.7 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 2 (4 กฎ)

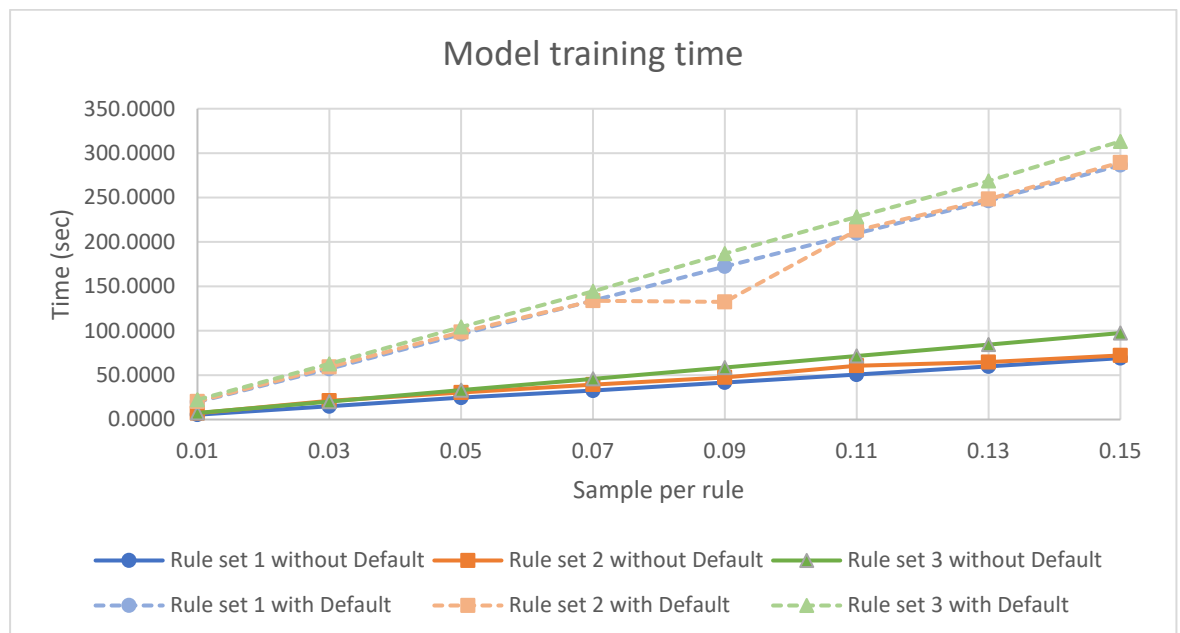
#### 4.2.2.3. ตารางผลการทดลองแบบอัตราส่วน Ratio เงื่อนไข Rule set ที่ 3 (6 กฎ)

	Ratio per rule	Total packet	Create packet time	Model training time	Train accuracy	Evaluate time	Test accuracy	True positive	True negative	False positive	False negative
Without Default	0.01	1,148	4.9646	7.4414	77.35%	1.687930107	62.09%	14022	10812	9188	5978
	0.03	3,453	5.0026	20.1497	92.41%	1.696537256	67.51%	16963	10042	9958	3037
	0.05	5,757	5.2669	33.0984	85.57%	1.664369583	53.29%	20000	1315	18685	0
	0.07	8,059	4.9777	45.8461	85.57%	1.582185745	53.15%	20000	1259	18741	0
	0.09	10,364	5.2534	58.4993	85.57%	2.296858788	53.18%	20000	1272	18728	0
	0.11	12,668	5.4734	71.6172	85.56%	1.924854279	56.47%	20000	2587	17413	0
	0.13	14,972	5.2141	84.4273	85.56%	1.99267149	53.18%	20000	1272	18728	0
	0.15	17,277	5.7127	97.3888	85.56%	2.053668261	52.34%	20000	936	19064	0
With Default	0.01	3,769	26.4840	22.2643	62.09%	1.900811434	64.92%	18287	7680	12320	1713
	0.03	11,317	65.1995	62.6677	73.89%	1.670017242	50.00%	0	20000	0	20000
	0.05	18,863	112.5015	104.1826	73.89%	1.679123163	50.00%	0	20000	0	20000
	0.07	26,408	140.5160	144.4166	73.89%	1.664469957	50.00%	0	20000	0	20000
	0.09	33,956	182.8826	186.8493	73.88%	1.754523516	50.00%	0	20000	0	20000
	0.11	41,502	215.6940	228.2477	73.88%	1.672006369	50.00%	0	20000	0	20000
	0.13	49,049	257.9227	268.6818	73.88%	1.775111437	50.00%	0	20000	0	20000
	0.15	56,597	292.7492	313.1831	73.88%	1.84391284	50.00%	0	20000	0	20000

ตารางที่ 4.8 ตารางผลการทดลองแบบอัตราส่วน Ratio Rule set ที่ 3 (6 กฎ)

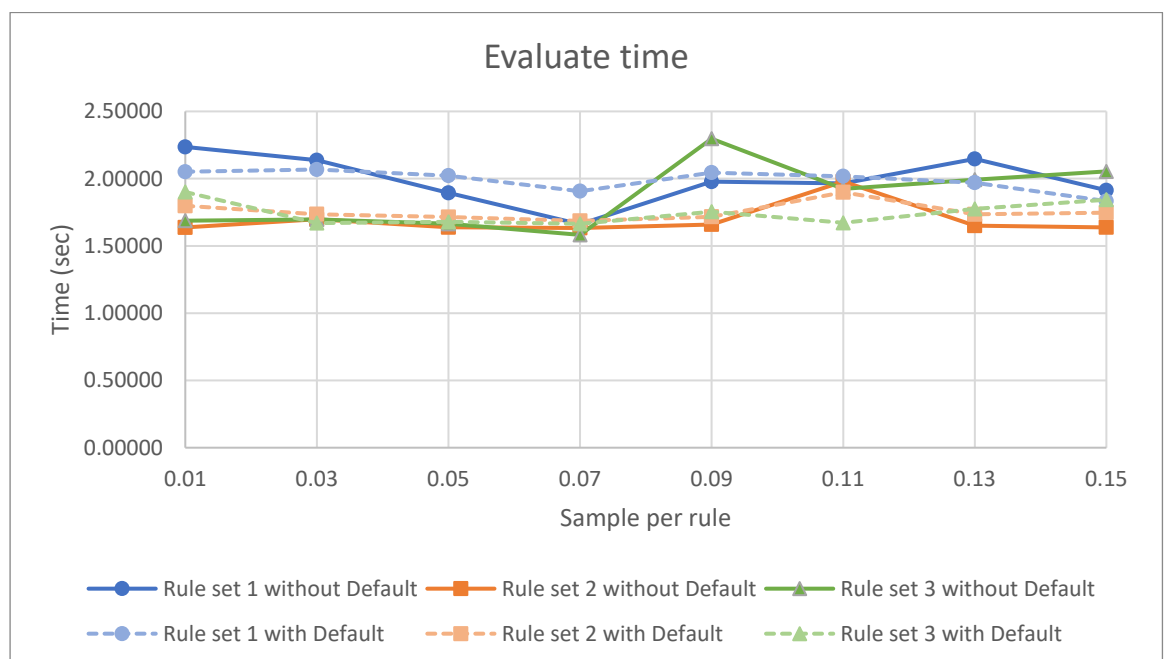
#### 4.2.3. กราฟผลลัพธ์การทดลองแบบอัตราส่วน Ratio

##### 4.2.3.1. กราฟผลลัพธ์ เวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



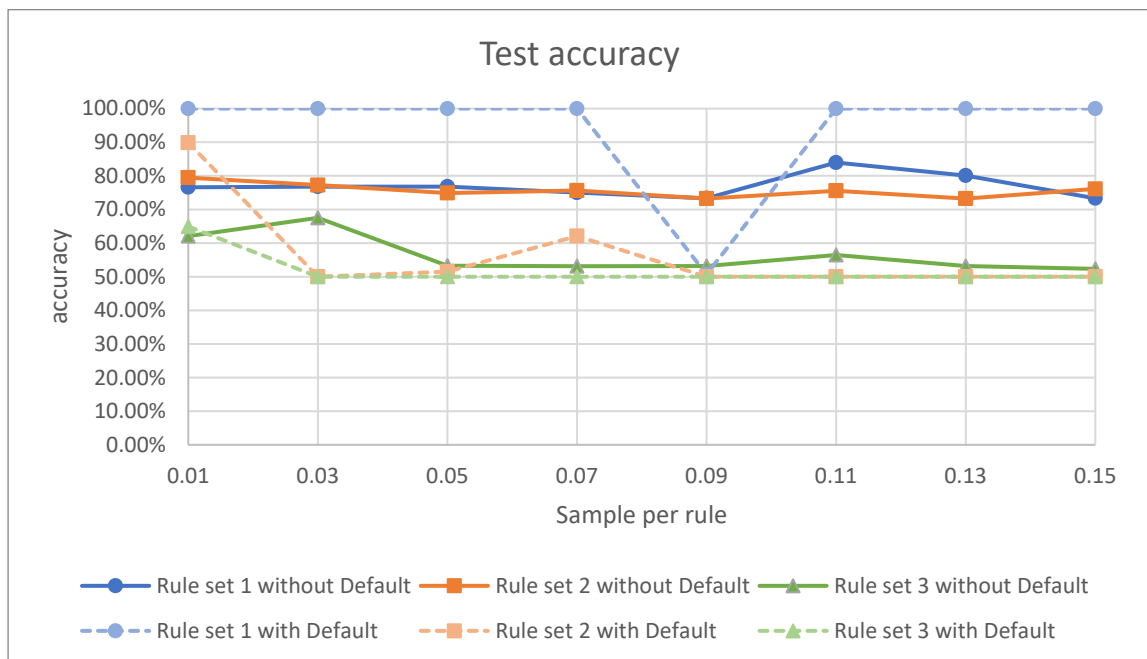
รูปที่ 4.4 กราฟเวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎ (Ratio)

##### 4.2.3.2. กราฟผลลัพธ์ เวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



รูปที่ 4.5 กราฟเวลาในการทำนายชุดทดสอบ : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎ (Ratio)

#### 4.2.3.3. กราฟผลลัพธ์ เวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎไฟร์วอลล์



รูปที่ 4.6 กราฟเวลาในการฝึกสอนโมเดล : อัตราส่วนข้อมูลฝึกสอนต่อ 1 กฎ (Ratio)

## บทที่ 5

### ผลการวิเคราะห์การทดลอง

เป้าหมายหลักของบทนี้คือการวิเคราะห์ผลการทดลองจากการนำชุดข้อมูลฝึกสอนที่สร้างจากกฎของไฟร์วอลล์ที่ออกแบบให้ตรงตามจุดประสงค์ของสมมติฐาน เพื่อหาชุดข้อมูลฝึกสอนที่สามารถทำให้โมเดลมีประสิทธิภาพในด้านความแม่นยำในการทำนายและเวลาที่ใช้ได้ดีที่สุด จึงจำเป็นต้องมีการวิเคราะห์ในเชิงเปรียบเทียบ ปรับรูปแบบกราฟเพื่อหาความสัมพันธ์ของตัวแปรต่างๆ

#### 5.1. การวิเคราะห์กลไกการทำงานโดยรวมของโมเดล

##### 5.1.1 วิเคราะห์ความสัมพันธ์ เวลาที่ใช้ในการฝึกสอนโมเดลและจำนวนข้อมูลฝึกสอน

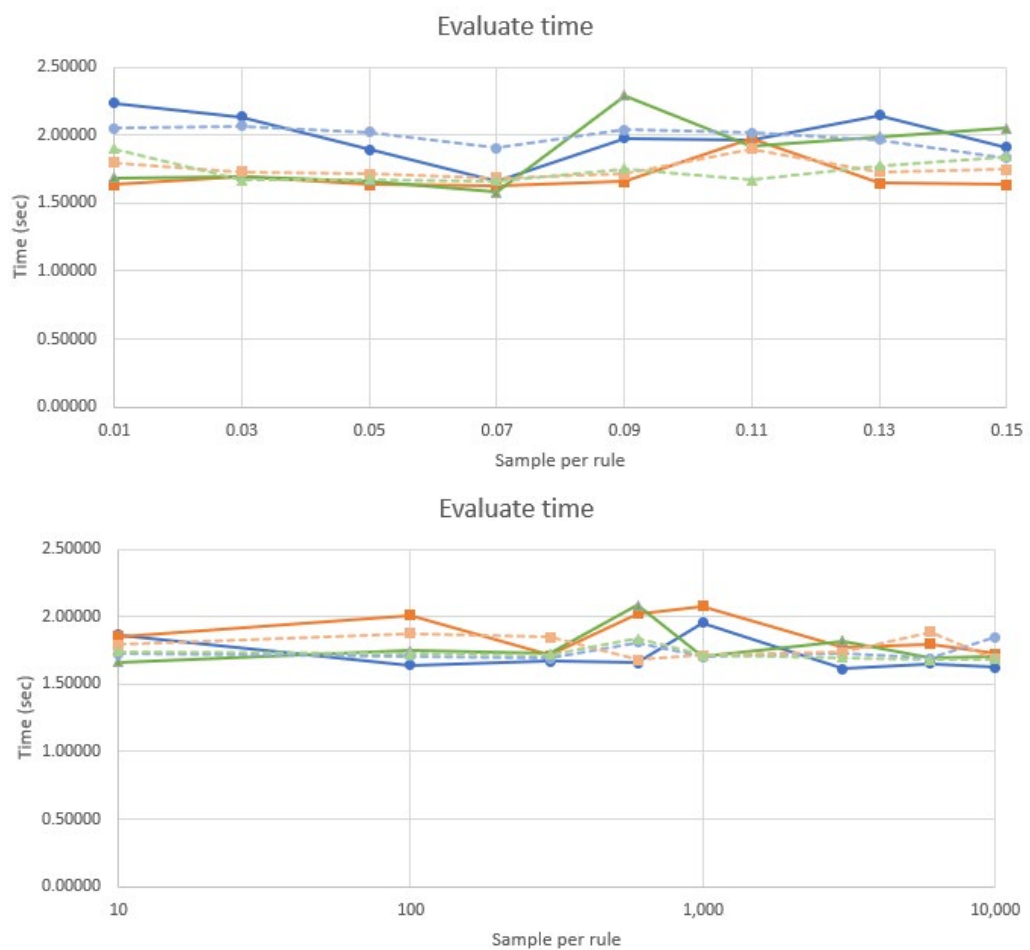
จากผลการทดลองในบทก่อนหน้าพบว่า ทั้ง 2 ผลการทดลอง เมื่อมีจำนวนชุดข้อมูลฝึกสอนในระบบมากขึ้น โมเดลจะใช้เวลาในการเรียนรู้ชุดข้อมูลฝึกสอนในอัตราคงที่ สังเกตได้จากกราฟที่ออกมามีลักษณะใกล้เคียงกับกราฟเส้นตรงมาก



รูปที่ 5.1 กราฟผลลัพธ์ เวลาที่ใช้ในการฝึกสอนโมเดล : จำนวนชุดข้อมูลฝึกสอนที่ใช้

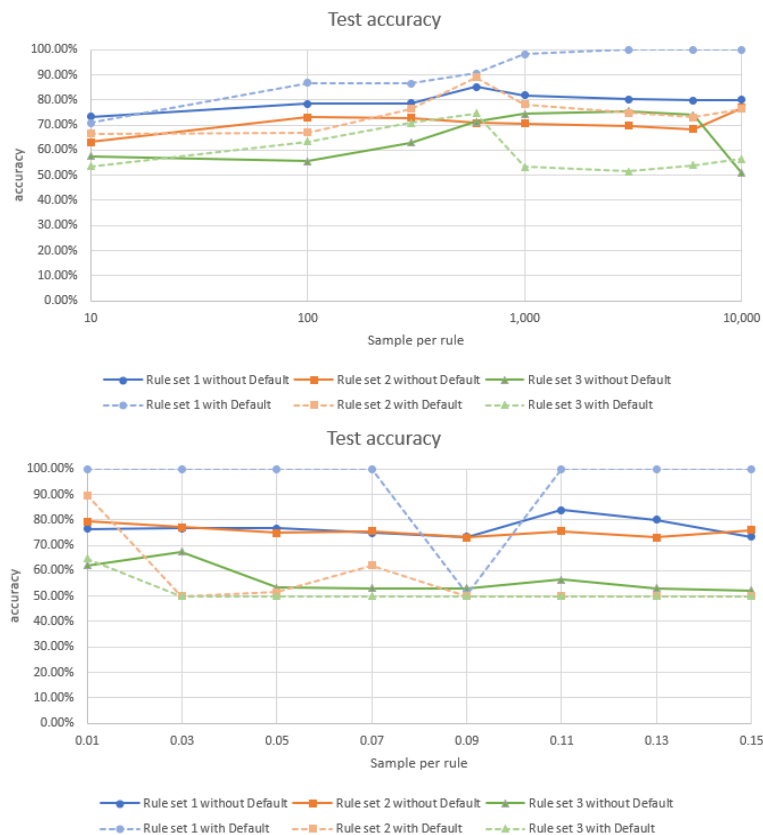
### 5.1.2. วิเคราะห์ความสัมพันธ์ เวลาที่ใช้ในการประมวลผลและจำนวนข้อมูลที่ใช้ฝึกสอน

จากผลการทดลองในบทก่อนหน้าพบว่า ทั้ง 2 ผลการทดลอง จะเห็นได้ว่าจำนวนชุดข้อมูล ฝึกสอนแทบไม่ส่งผลกับเวลาที่ใช้ในการประมวลผล นั่นหมายความว่าถ้าหากเราใช้ชุดกฎไฟร์ วอลล์ที่มีเงื่อนไขมากขึ้นก็ยังใช้เวลาในการประมวลผลเท่าเดิม จากภาพ 5.2 จะเห็นได้ว่าทั้ง 2 รูป ถึงจะใช้เวลามากขึ้นหรือน้อยลงบ้าง แต่ค่าความแตกต่างจะอยู่ในเลืยวินาทีเท่านั้น



รูปภาพที่ 5.2 เปรียบเทียบกราฟผลลัพธ์เวลาที่ใช้ในการประมวลของ N Sample และ Ratio

### 5.1.3. วิเคราะห์ความสัมพันธ์ ความแม่นยำการทำนายต่อจำนวนของข้อมูลฝึกสอน



รูปภาพที่ 5.3 เปรียบเทียบกราฟผลลัพธ์ความแม่นยำของการแบ่งชุดข้อมูลฝึกสอนแต่ละแบบ

จะเห็นได้ว่า ถ้าหากเป็น N Sample ค่าความแม่นยำที่ได้จะค่อยๆเพิ่มขึ้น และขึ้นเกือบจุดสูงสุดที่จุดหนึ่ง ซึ่งตัวแปรที่ใช้ในการทดลองจุดนั้นคือ 600 ชุดข้อมูลฝึกสอนต่อหนึ่งกฎไฟร์วอลล์ แต่เมื่อให้จำนวนข้อมูลฝึกสอนที่มากกว่านั้น ความแม่นยำในทุกชุดกฎไฟร์วอลล์จะเริ่มตกลงเล็กน้อย โดยเฉพาะชุดกฎไฟร์วอลล์ที่มีเงื่อนไขที่มากกว่า ยกเว้นชุด 2 กฎไฟร์วอลล์ที่มี Default Rule ด้วย

จึงสรุปได้ว่า ความซับซ้อนและเงื่อนไขของชุดข้อมูลฝึกสอนส่งผลต่อ การเรียนรู้ของโมเดล หมายความว่า ถ้าหากเงื่อนไขกฎไฟร์วอลล์ที่ใช้มีจำนวนเงื่อนไขและกฎการประเมินที่มากขึ้น จำเป็นต้องหาจำนวนของข้อมูลที่จะใช้ฝึกสอนที่เหมาะสมที่ทำให้โมเดลมีประสิทธิภาพมากที่สุด

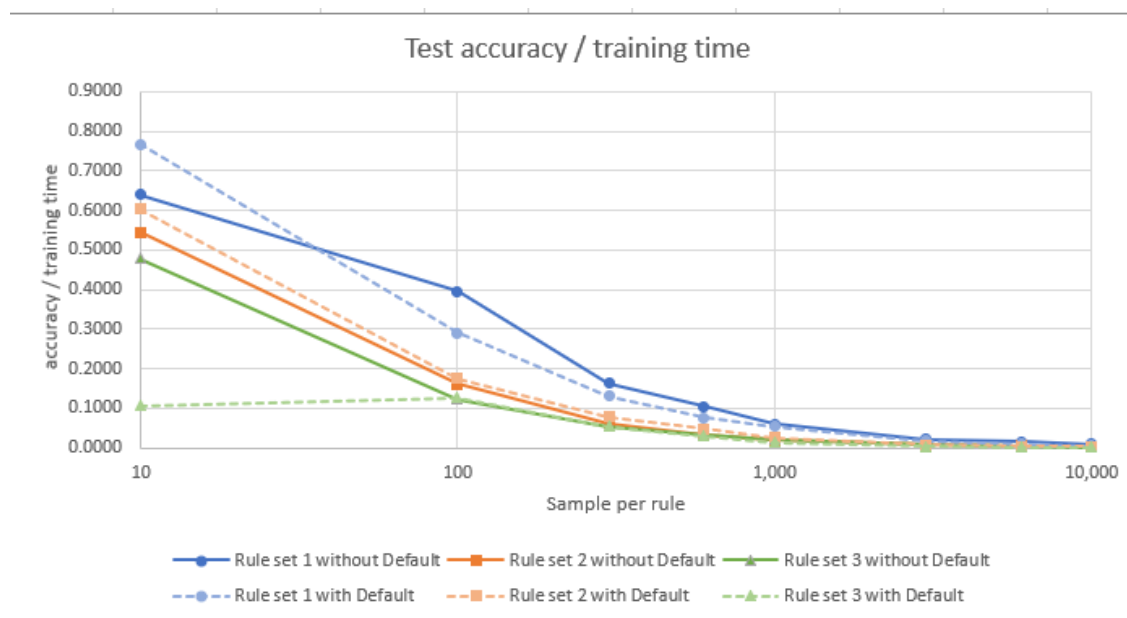
## 5.2. การวิเคราะห์ประสิทธิภาพการทำงานของโมเดล

### 5.2.1. อัตราการเรียนรู้ของโมเดล

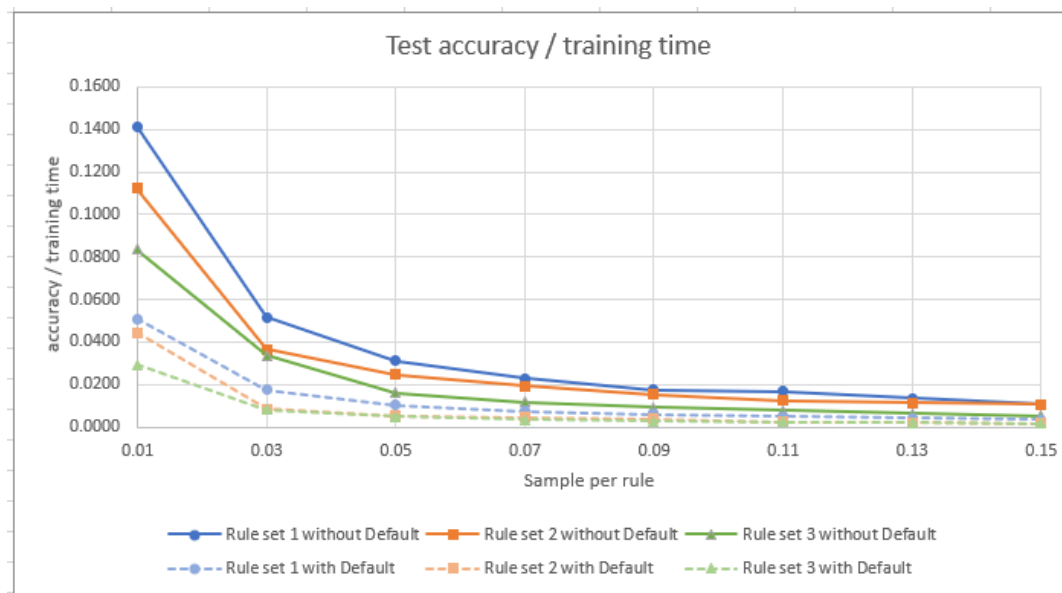
ตัวแปรสำคัญในการวัดผลในเชิงประสิทธิภาพ ได้แก่ ความแม่นยำในการทำนายผล เวลาที่ใช้ในการฝึกโมเดล และ จำนวนชุดข้อมูลทดสอบ ซึ่งทั้ง 3 ค่านี้ให้ความหมายในเชิงประสิทธิภาพได้ดังนี้

- โมเดลที่มีความแม่นยำสูงกว่า เป็นโมเดลที่มีประสิทธิภาพมากกว่า
- โมเดลที่ใช้เวลาในการเรียนรู้น้อยกว่าย่อมดีกว่าโมเดลที่ใช้เวลาในการเรียนรู้มากกว่า ถ้าหากโมเดลทั้งสองให้ผลลัพธ์ความถูกต้องในการทำนายผลเท่ากัน
- โมเดลที่ใช้จำนวนชุดข้อมูลฝึกสอนน้อยกว่าจะดีกว่าโมเดลที่ใช้จำนวนชุดข้อมูลฝึกสอนมากกว่า ถ้าหากโมเดลทั้งสองให้ผลลัพธ์ความถูกต้องในการทำนายผลเท่ากัน ซึ่งจำนวนชุดข้อมูลฝึกสอนจะมีผลโดยตรงกับเวลาที่ใช้ นั่นหมายความว่า เราจะต้องใช้เวลาในการสร้างชุดข้อมูลและฝึกฝนนานขึ้น

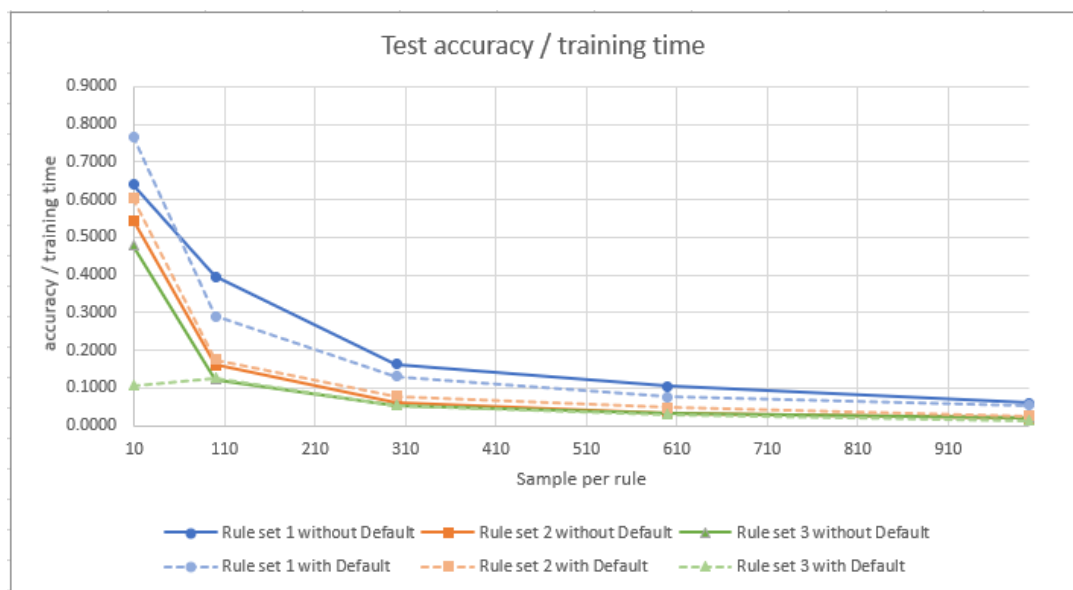
ซึ่ง 3 ตัวแปรนี้ ทำให้ได้กราฟอีกตัวหนึ่งคือกราฟ ความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอน



รูปที่ 5.4 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของ N Sample



รูปที่ 5.5 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของอัตราส่วน Ratio



รูปที่ 5.6 กราฟความแม่นยำ / เวลาฝึกโมเดล : จำนวนชุดข้อมูลฝึกสอนของ N Sample (2)

จะเห็นว่ากราฟทั้งสองรูปแบบ ทั้ง N Sample และอัตราส่วน Ratio มีอัตราการเรียนรู้ที่มีลักษณะคล้ายกันคือ โมเดลที่มีการเรียนรู้จากชุดข้อมูลฝึกสอนน้อยยังสามารถคำนวณหาผลลัพธ์ได้ถูกต้องบ้างอยู่ แต่การที่จะเพิ่มความแม่นยำได้นั้นจะต้องเพิ่มจำนวนชุดข้อมูลฝึกสอนไปอีกเกือบเท่าตัวหรือหลายเท่า นั่นหมายความว่าอัตราการเรียนรู้จะเริ่มน้อยลงไปเรื่อยๆแปรผกผันกับจำนวนชุดข้อมูลฝึกสอนที่ป้อนเข้าไป



### 5.2.2. การเลือกหาจุดจำนวนชุดข้อมูลฝึกสอนที่เหมาะสมที่สุดในการพัฒนาโมเดล

ในการเลือกจำนวนชุดข้อมูลฝึกสอนมาใช้ในการพัฒนาโมเดล ผลลัพธ์สุดท้ายที่สามารถตอบโจทย์ประสิทธิภาพทั้ง 3 ค่าได้ โดยประกอบไปด้วย จำนวนชุดข้อมูลฝึกสอน เวลาที่ใช้ในการฝึกสอน และความแม่นยำในการทำนายผล จะต้องให้ผลลัพธ์ได้ดีที่สุด ซึ่งในหัวข้อการวิเคราะห์นี้ เราได้ใช้ผลลัพธ์การทดลองและหลักการมาอ้างอิง ดังนี้

#### 5.2.2.1. การอ้างอิงความสัมพันธ์ของเวลาฝึกสอนและจำนวนชุดข้อมูลฝึกสอน

จากการนำกราฟที่ 5.1 มาวิเคราะห์ พบว่าจำนวนชุดข้อมูลฝึกสอนโมเดลและเวลาที่ใช้ในการฝึกสอนโมเดลมีความสัมพันธ์แปรผันตรงด้วยอัตราการเพิ่มค่าที่คงที่ หมายความว่าเราสามารถหาเวลาที่ใช้ในการฝึกสอนโมเดลได้ถ้าหากเราทราบจำนวนชุดข้อมูลฝึกสอนที่ใช้ หมายความว่า การเลือกจุดจำนวนชุดข้อมูลฝึกสอนที่ให้ค่าความถูกต้องได้เยอะที่สุด เหมาะสมที่จะเป็นจำนวนที่ควรหยิบมาใช้มากที่สุดนั่นเอง ซึ่งเป็นไปตามสูตร

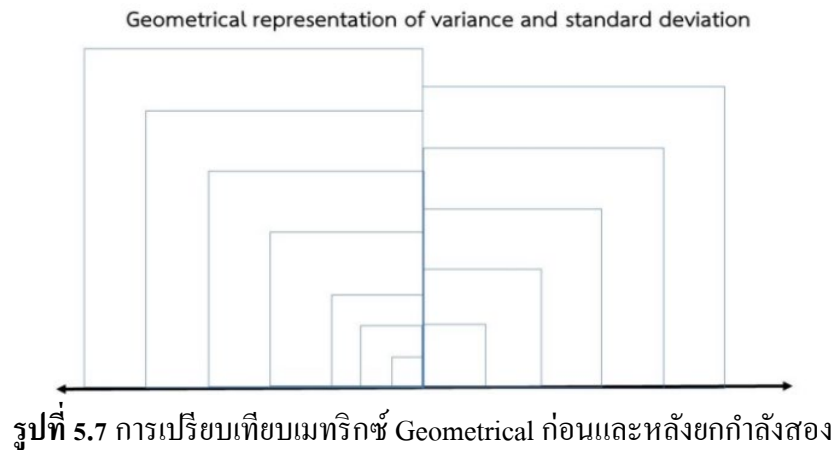
$$nt = T$$

- $n$  เป็นจำนวนชุดข้อมูลฝึกสอนที่ต้องการ
- $t$  เป็นเวลาที่โมเดลใช้ในการเรียนรู้ต่อ 1 ข้อมูล
- $T$  เป็นเวลาทั้งหมดที่โมเดลใช้ในการเรียนรู้จนเสร็จ

#### 5.2.2.2. การใช้หลักของการหาค่าความแปรปรวนในการเปลี่ยนมิติการเปรียบเทียบ

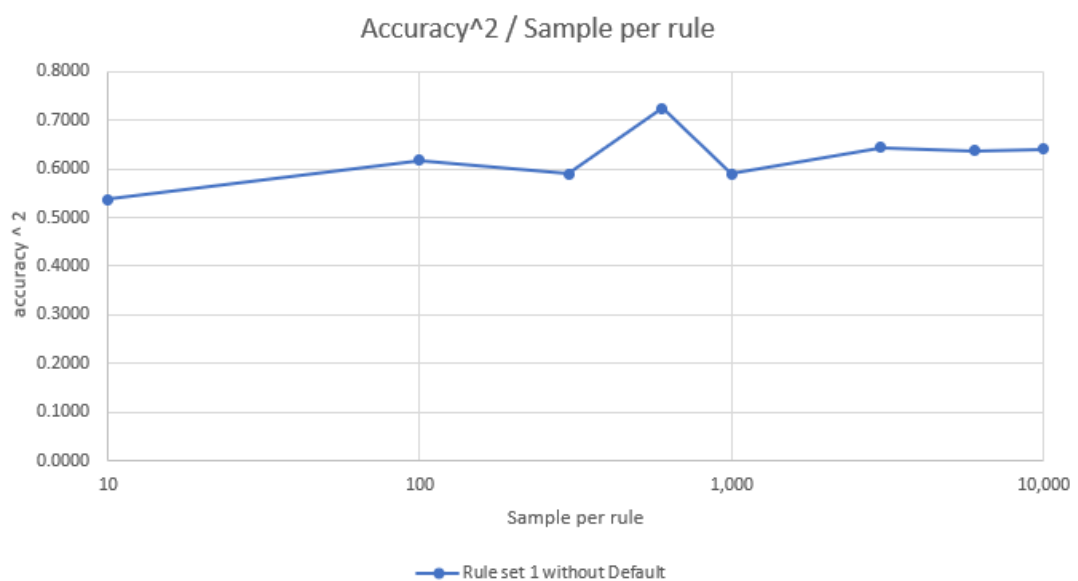
ความแปรปรวนคือความกว้างของข้อมูลหรือความห่างของการกระจายตัวของชุดข้อมูล โดยในทางสถิติหรือการเปรียบเทียบเชิงคณิตศาสตร์มักมีการนำความแปรปรวนและส่วนเบี่ยงเบนมาใช้ในการหาค่าความแตกต่างของกราฟ ซึ่งหลักการหนึ่งที่ทำให้เห็นผลชัดคือการนำค่ามายกกำลังสองตามสูตรความแปรปรวน เราจะสามารถหาจุดแตกต่างหรือมีความแปรปรวนได้ชัดเจนขึ้นโดยสมการมีรูปแบบสูตร ดังนี้

$$s^2 = \sum \frac{(x - \bar{x})}{n - 1}$$

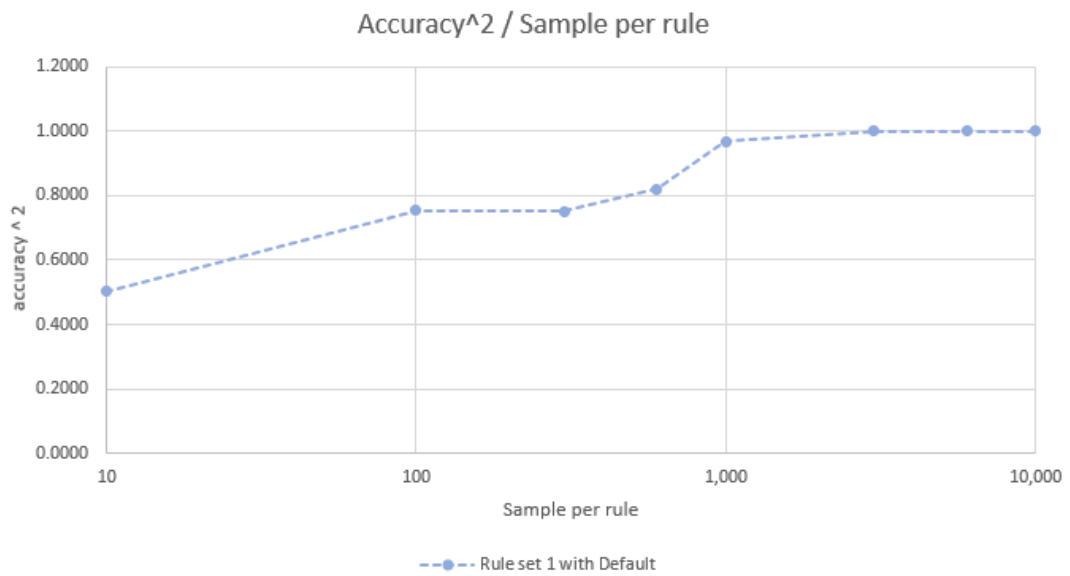


จากประเด็นก่อนหน้านี้ที่นำมาพิจารณาคือจำนวนชุดข้อมูลฝึกสอนกับเวลาในการเรียนรู้มีทิศทางความสัมพันธ์แบบแปรผันตรงแบบคงที่ จึงสามารถนำชุดข้อมูลฝึกสอนมาเป็นเกณฑ์ในการวัดเรื่องเวลาได้เลย และตัวแปรอีกตัวหนึ่งคือความแม่นยำ โดยเราจะอ้างอิงหลักการหาความแปรปรวนที่มีการยกกำลังสอง มาเพิ่มมิติให้กับค่าความแม่นยำเพื่อให้เห็นความแตกต่างของความแม่นยำได้ดีขึ้น สาเหตุมาจากความแม่นยำมีค่าระยะหรือ Range ที่ต่ำคืออยู่ระหว่าง 0 ถึง 1 แน่นนอน (หรือ 0% - 100% ) ทำให้สรุปได้เป็นกราฟใหม่ คือค่าความแม่นยำกำลังสองต่อจำนวนข้อมูลฝึกสอนที่ใช้

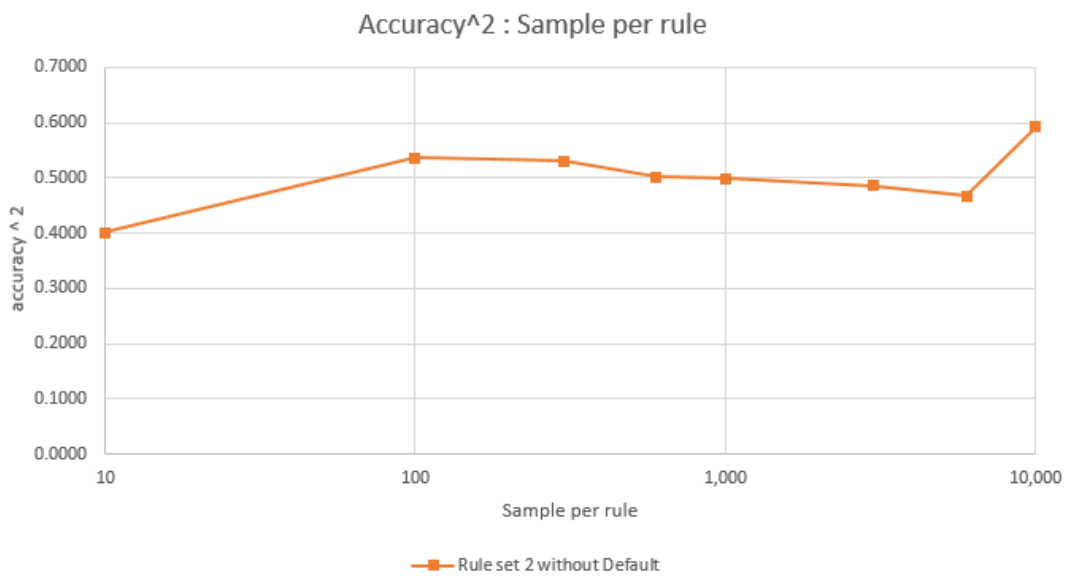
กราฟแสดงผลลัพธ์ที่ได้จากการคำนวณใหม่แบบ N Sample



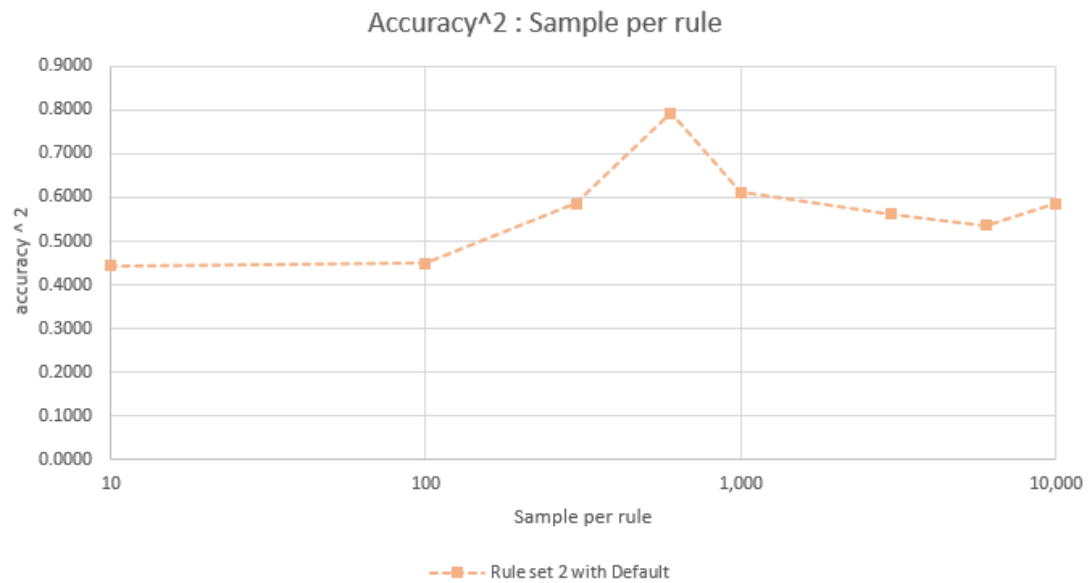
รูปที่ 5.8 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R1, Without Default)



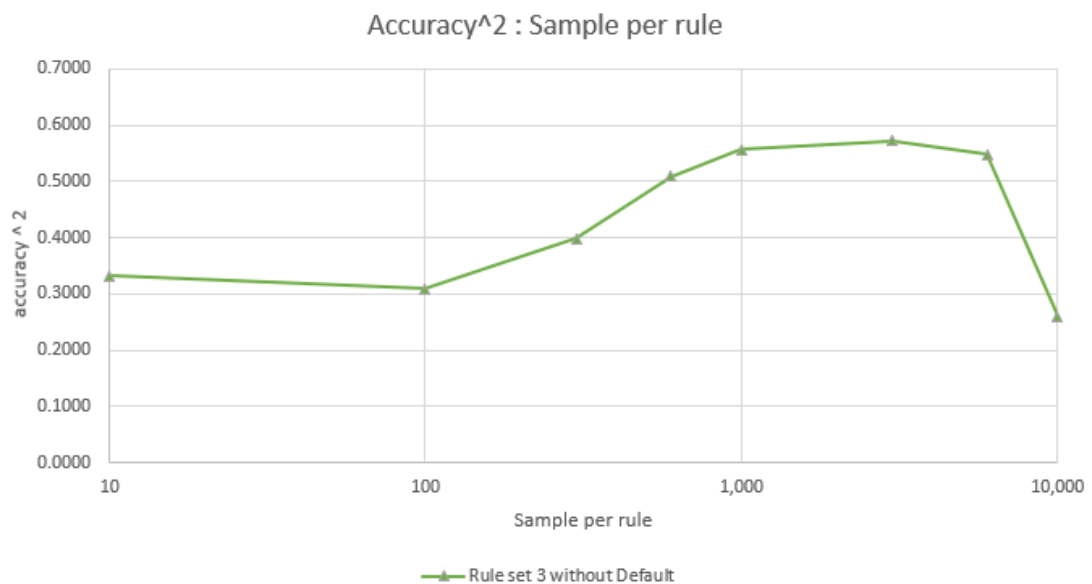
รูปที่ 5.9 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R1, With Default)



รูปที่ 5.10 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R2, Without Default)



รูปที่ 5.11 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R2, With Default)

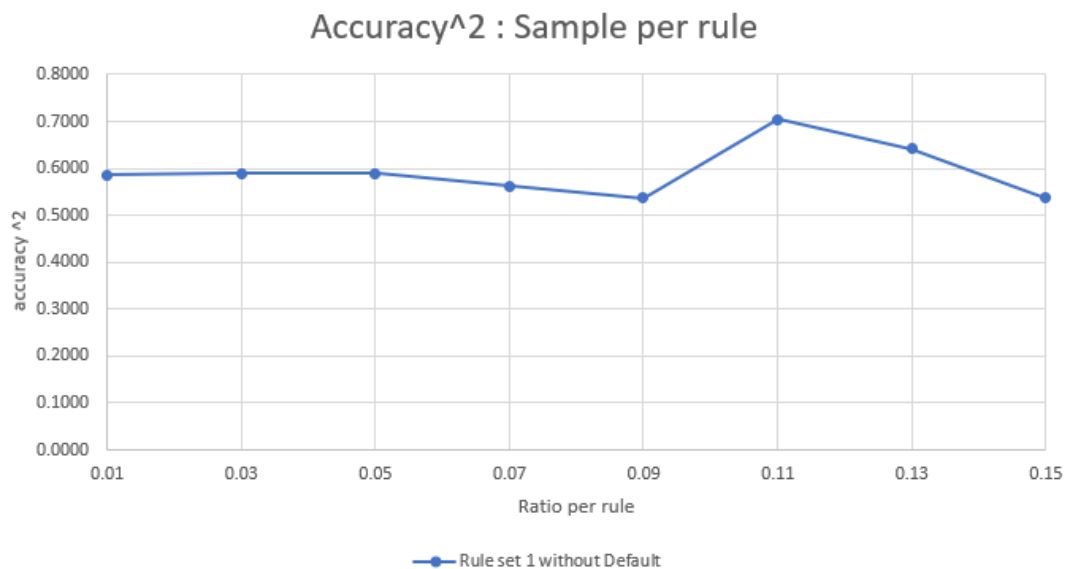


รูปที่ 5.12 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R3, Without Default)

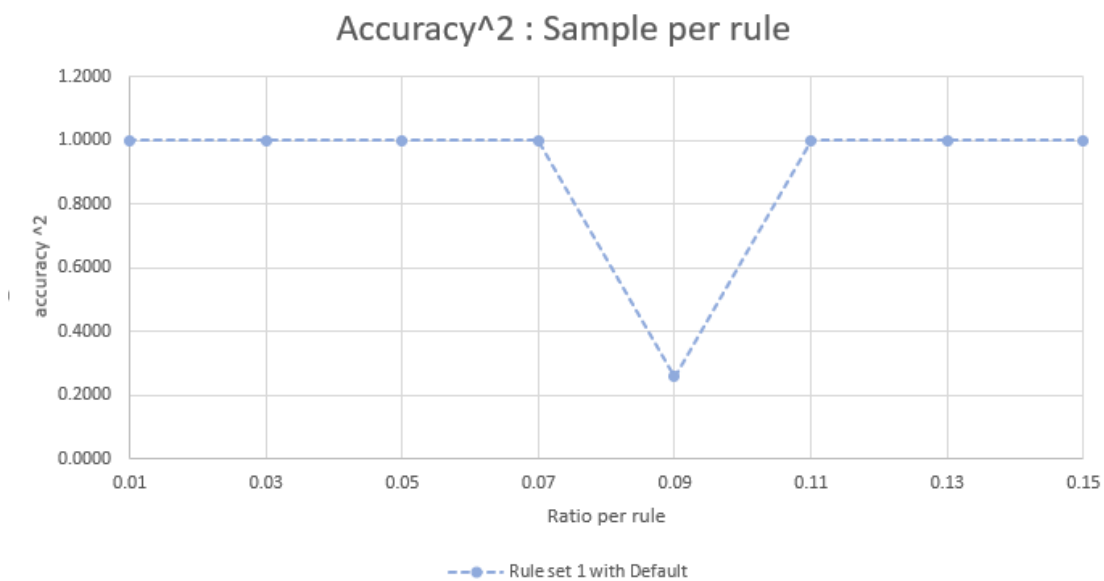


รูปที่ 5.13 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (N Sample, R3, With Default)

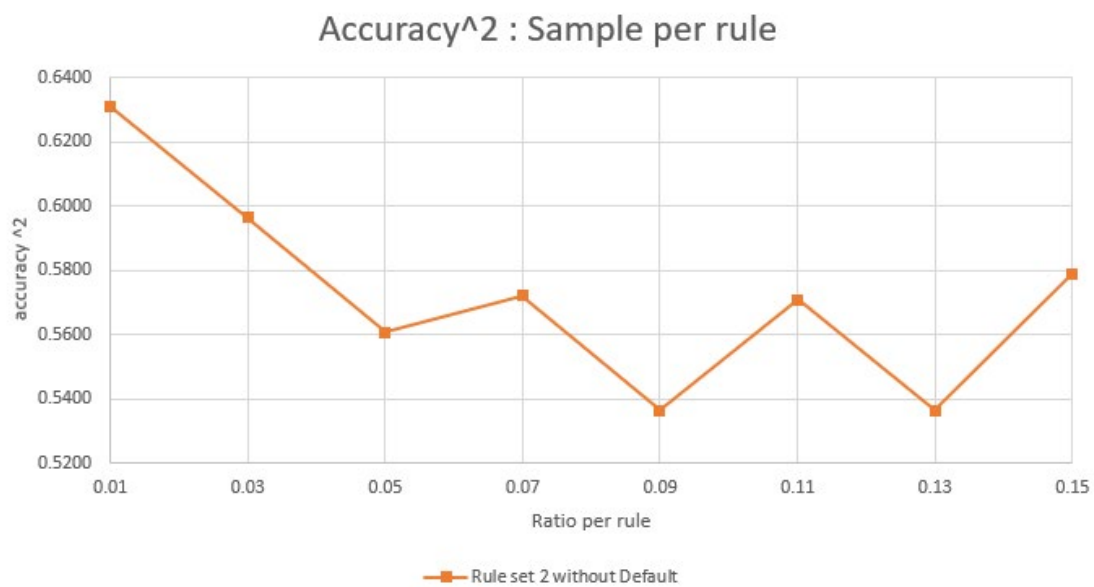
กราฟแสดงผลลัพธ์ที่ได้จากการคำนวณใหม่แบบอัตราส่วน Ratio



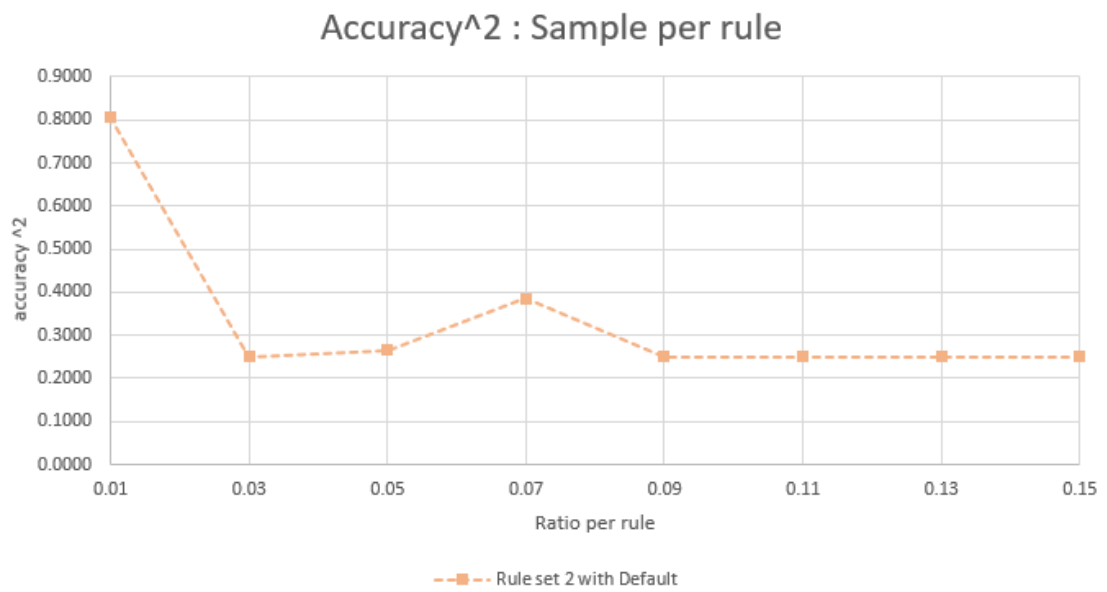
รูปที่ 5.14 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R1, Without Default)



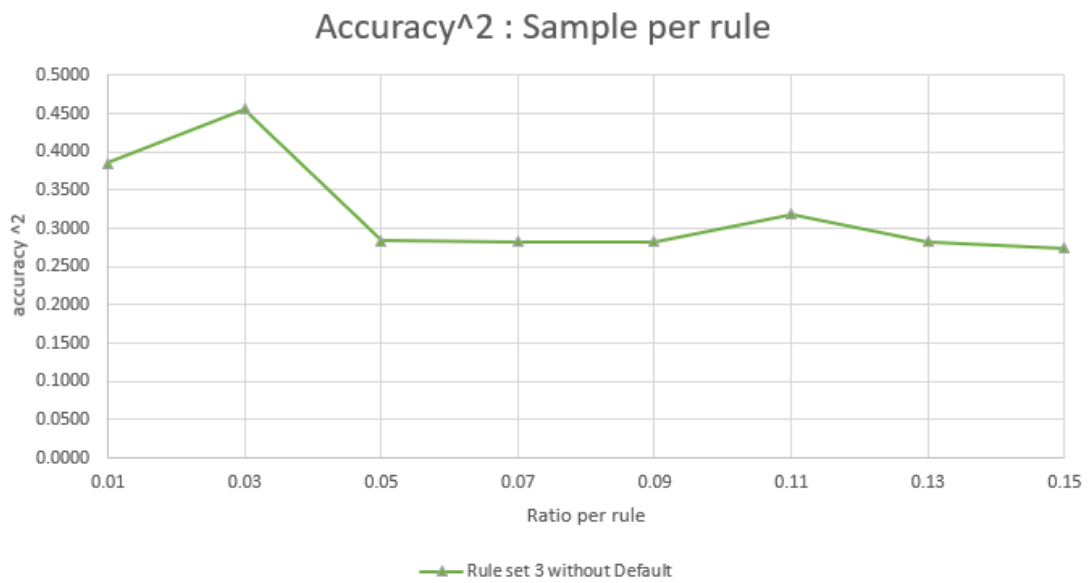
รูปที่ 5.15 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R1, With Default)



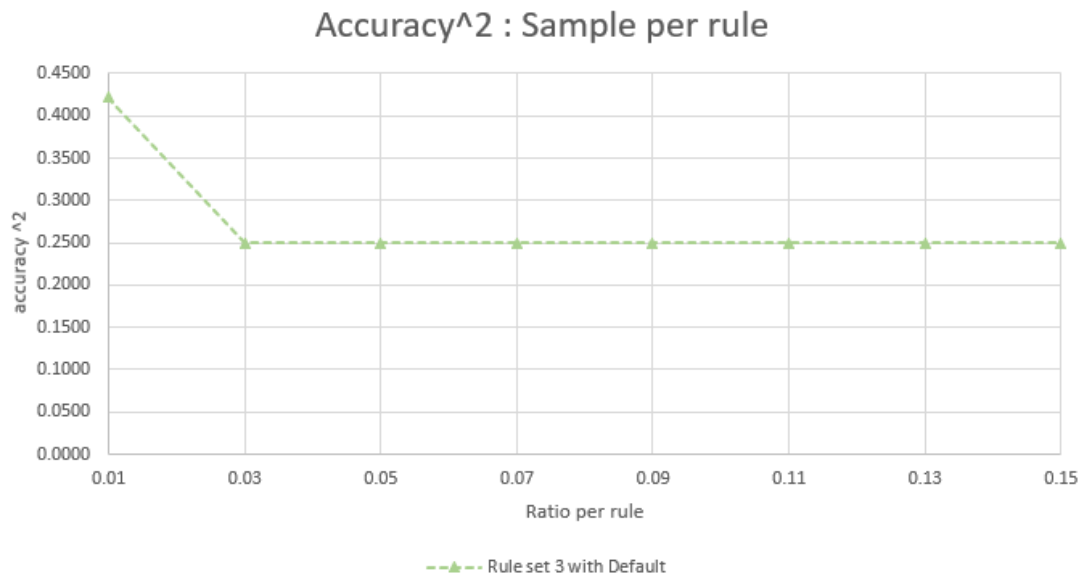
รูปที่ 5.16 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R2, Without Default)



รูปที่ 5.17 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R2, With Default)



รูปที่ 5.18 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R3, Without Default)



รูปที่ 5.19 กราฟแม่นยำยกกำลังสอง: จำนวนข้อมูลฝึกสอนที่ใช้ (Ratio, R3, With Default)

จากกราฟที่ได้ทำการปรับค่าตัวแปรใหม่ จะเห็นได้ว่าทั้งกราฟการแบ่งจำนวนชุดฝึกสอนแบบ N Sample และแบบอัตราส่วน Ratio เมื่อมีกฎเดียวจะสามารถหาค่าจุดสูงสุดที่คุ้มค่าทั้งความแม่นยำและเวลาที่ใช้ได้ในกราฟที่มีเงื่อนไขกฎไฟร์วอลล์ 2 กฎ แต่ถ้าหากคิดในเรื่องจำนวนชุดข้อมูลฝึกสอนที่ใช้แล้ว วิธีแบบ N Sample ที่จำนวน 600 ได้ค่ายกกำลังที่ 0.7 ซึ่งเท่ากับการแบ่งแบบอัตราส่วนที่ 0.11 แต่มีการใช้ Packet จำนวนมากถึง 9,010 ชุด นั้นหมายความว่าวิธีแบบ N Sample ใช้เวลาได้ดีกว่าแบบอัตราส่วน Ratio ดีกว่าถึง 15 เท่า ในขณะที่มีการใช้กฎไฟร์วอลล์เหมือนกัน

เมื่อมาถึงเงื่อนไขกฎไฟร์วอลล์ที่มี 4 กฎพบว่าแบบ N Sample ยังสามารถหาจุดเหมาะสมได้อยู่แต่ความแม่นยำของ Ratio ที่ Without Default ให้ผลได้ดีกว่าในช่วงอัตราส่วน 0.01 ซึ่งมีจำนวนชุดข้อมูลทั้งหมดเพียง 800 และให้ผลลัพธ์ที่ดีกว่า แต่เมื่อนำ Default Rule มาใช้คิดคำนวณด้วยปรากฏว่าแบบ N Sample ให้ค่าความแม่นยำได้ถึง 0.8 ในขณะที่การแบ่งอัตราส่วน Ratio มีอัตราความแม่นยำที่ตกลงเรื่อยๆและมีความแม่นยำน้อยกว่าแบบ N Sample มาก

ต่อมาก็คือเงื่อนไขที่มี 6 กฎไฟร์วอลล์ จากการทดลองพบว่าแบบอัตราส่วน Ratio ให้ผลความแม่นยำที่ต่ำกว่าแบบ N Sample ในทุกจุด ซึ่งหากดูจากการทดลองที่ผ่านมาแล้ว เมื่อมีเงื่อนไขมากขึ้น แบบอัตราส่วน Ratio แทบจะด้อยกว่าแบบ N Sample ในทุกด้าน ไม่ว่าจะเป็นทั้งเรื่องความแม่นยำและเวลาที่ใช้ จึงสรุปได้ว่าเราไม่ควรนำวิธีการแบ่งจำนวนชุดฝึกสอนแบบอัตราส่วน Ratio ที่เท่ากันมาใช้ในการหาจำนวนชุดข้อมูลฝึกสอนที่จะเอามาป้อนเข้าโมเดล



## บทที่ 6

# สรุปผลและข้อเสนอแนะ

### 6.1. สรุปผลการดำเนินงานวิจัย

จากการทดลองและวิเคราะห์ผลพบว่า ความแม่นยำในการทำนายผลลัพธ์ทั้งการแบ่งชุดข้อมูลฝึกสอนทั้ง 2 แบบ คือการแบ่งแบบ N Sample และแบบอัตราส่วน Ratio ผลการทดลองจะสังเกตได้ชัดเจนว่า แบบ N Sample ให้ผลดีกว่าในเกือบทุกๆด้าน ไม่ว่าจะเป็นเวลาที่ใช้และความแม่นยำ และเมื่อกลุ่มมีความซับซ้อนหรือจำนวนมากขึ้นเรื่อยๆ ก็จะยิ่งผลความแตกต่างได้ชัดมากขึ้น คาดว่าการแบ่งที่อัตราส่วน Ratio ให้ความคุ้มค่าได้น้อยกว่าเพราะความไม่เท่าเทียมกันของจำนวนชุดฝึกสอนในแต่ละกลุ่มไฟร่วอลล์ จำนวนที่ต่างกันมากเกินไปทำให้ความสามารถในการเรียนรู้ลดลง จนโมเดลไม่สามารถทำนายผลลัพธ์ได้ดีอีกต่อไป

ในงานวิจัยถัดไปจะเป็นการลงรายละเอียดเกี่ยวกับการพัฒนาแบ่งชุดข้อมูลฝึกสอนด้วยอัลกอริทึมแบบใหม่ ซึ่งเราได้คาดเดาว่าวิธีนี้จะเป็นการแก้ไขปัญหาวิธีการแบ่งชุดข้อมูลที่เป็นแบบอัตราส่วน โดยประเด็นปัญหาที่สามารถเห็นได้ชัดคือ การแบ่งข้อมูลฝึกสอนที่มีความแตกต่างกันทางด้านกลุ่มของไฟร่วอลล์มากเกินไปจนทำให้ไม่สามารถทำนายชุดข้อมูลที่มีหลายเงื่อนไขได้ หรืออาจเพิ่มประเด็นวิจัยเพื่อเพิ่มความแม่นยำในการทำนายผล เช่น การปรับโมเดลหรือเปลี่ยนแปลงโครงสร้างของชุดข้อมูลฝึกสอน เป็นต้น

### 6.2. ปัญหาและอุปสรรคที่พบในงานวิจัย

- การพัฒนางานวิจัยใช้เวลานานมากกว่าที่คาดเอาไว้ เนื่องจากต้องพัฒนาโปรแกรมทั้งระบบควบคู่กับการทำทดลองไปด้วย ซึ่งการทดลองปัญหาประดิษฐ์ในเชิงเปรียบเทียบจำเป็นต้องทดลองซ้ำหลายรอบเพื่อให้ได้ผลลัพธ์ที่แม่นยำและวิเคราะห์ได้ ถ้าหากมีเวลาสำหรับการทดลองมากขึ้น อาจทำให้ได้ผลลัพธ์ที่แม่นยำและมีรายละเอียดที่น่าพึงพอใจมากขึ้น
- โปรแกรมในการสร้างชุดข้อมูลฝึกสอนมีข้อจำกัดหลายอย่าง เพราะเป็นเพียงการจำลองข้อมูลจาก Packet Header เพียงอย่างเดียว ยังไม่ได้ลงรายละเอียดในส่วนของ Data Field และยังจำเป็นต้องลดความเป็นไปได้ของ Possible Packet เนื่องจากมีปัญหาที่เครื่องคอมพิวเตอร์ที่ใช้ประมวลผลไม่สามารถรับภาระแบนด์วิดท์ที่มากเกินไปได้

### 6.3. ข้อเสนอแนะและแนวทางการพัฒนางานวิจัยในอนาคต

- อาจมีวิธีการแก้ไขปัญหการแบ่งอัตราส่วนชุดข้อมูลที่มีจำนวนต่างกันมากเกินไป อาจมีการใช้สูตรทางคณิตศาสตร์หรือมีอัลกอริทึมอื่นในการแบ่งจำนวนมาช่วยในการคำนวณหาจำนวนชุดข้อมูลที่เหมาะสมกับโมเดลได้
- พัฒนาโปรแกรมสร้างชุดข้อมูลฝึกสอนให้มีประสิทธิภาพมากขึ้น ให้สามารถออกแบบได้ใกล้เคียงกับข้อมูล Packet ในเครือข่ายจริง และประมวลผลสร้างชุดข้อมูลได้รวดเร็วขึ้น
- พัฒนาเครื่องมือโมเดลโครงข่ายประสาทเทียมให้มีประสิทธิภาพมากขึ้น อาจลองศึกษาความสัมพันธ์ของตัวแปรที่ส่งผลต่อการเรียนรู้ของโมเดล ซึ่งประกอบไปด้วย จำนวนรอบที่เรียนรู้ จำนวนโหนดและวิธีการประมวลผลในรูปแบบต่างๆ และสังเกตว่าค่าเหล่านี้มีผลกับความแม่นยำและเวลาที่ใช้ในการฝึกสอนของชุดข้อมูลฝึกสอนที่สร้างไว้หรือไม่
- มีการเพิ่มสมมติฐานขึ้นใหม่ให้ใกล้เคียงกับเครือข่ายจริงมากขึ้น เช่น การเพิ่มกฎไฟร์วอลล์ที่มีความกระชับ หรือกำหนดให้มีข้อมูลที่จะพิจารณามากขึ้น เพิ่มจำนวน Rule set หรืออาจลองนำข้อมูลฝึกสอนจาก Application Layer มาใช้ควบคู่ด้วย

## บรรณานุกรม

- [1] TensorFlow Teams. “**Essential Documentation**” [Online]. Available :  
<https://www.tensorflow.org/guide>. 2020
- [2] nessesence. “ปัญญาประดิษฐ์ (AI: Artificial Intelligence) คืออะไร” [Online]. Available :  
<https://www.thaiprogrammer.org/2018/12/whatisai/>. 2018
- [3] Rene Molenaar. “**IPv4 Packet Header**” [Online]. Available :  
<https://networklessons.com/cisco/ccna-routing-switching-icnd1-100-105/ipv4-packet-header>. 2020
- [4] Sci-kit learn developers. “**scikit classification model**” [Online]. Available : <https://scikit-learn.org/stable/search.html?q=classification>. 2020
- [5] TensorFlow Teams. “**พื้นฐาน Deep Learning**” [Online]. Available :  
<https://www.tensorflow.org/guide>. 2020
- [6] sinlapachai lorpaiboon. “**มาเรียนรู้คำสั่งของ Pandas ใน Python ที่เอาไว้ใช้สำหรับการจัดการข้อมูลกัน**” [Online]. Available : <https://medium.com/@sinlapachai.hon/มาเรียนรู้การใช้-การทำความสะอาดข้อมูลด้วย-python-โดยการใช-pandas-กัน-2f5049640e70>. 2020
- [7] T. Hastie, R. Tibshirani, J. Friedman. **The Elements of Statistical Learning (Second Edition)**. : Springer-Verlag. 2009
- [8] Saishruthi Swaminathan. “**Logistic Regression — Detailed Overview**” [Online]. Available :  
<https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc>. 2018
- [9] Anas Al-Masri. “**What Are Overfitting and Underfitting in Machine Learning?**” [Online]. Available : <https://towardsdatascience.com/what-are-overfitting-and-underfitting-in-machine-learning-a96b30864690>. 2019
- [10] Will Koehrsen. “**Overfitting vs. Underfitting: A Complete Example**” [Online]. Available :  
<https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765>. 2018

- [11] Ahmed Gad. “**Beginners Ask ‘How Many Hidden Layers/Neurons to Use in Artificial Neural Networks?’**” [Online]. Available :  
<https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks-51466afa0d3e>. 2018
- [12] Aurélien Géron. **Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow**. Sixth Release. United States of America : O’Reilly Media, Inc. 2019
- [13] D. STATHAKIS. “How many hidden layers and nodes?” **International Journal of Remote Sensing**, Vol. 30, No. 8, 20 April 2009. pp2133–2147
- [14] Jeff Heaton. “**Heaton Research The Number of Hidden Layers**” [online]. Available : The Number of Hidden Layers | Heaton Research. 2017

ภาคผนวก

# ขั้นตอนการติดตั้งไลบรารีและเครื่องมือสำหรับการใช้งานโครงข่ายประสาทเชิงลึกด้วยคอมพิวเตอร์ส่วนบุคคล

## ส่วนที่ 1 ส่วนประกอบที่จำเป็นในการติดตั้งโปรแกรม

### 1.1. ส่วนประกอบที่จำเป็นในการติดตั้งโปรแกรม

1.1.1. Windows 10 x64 bits

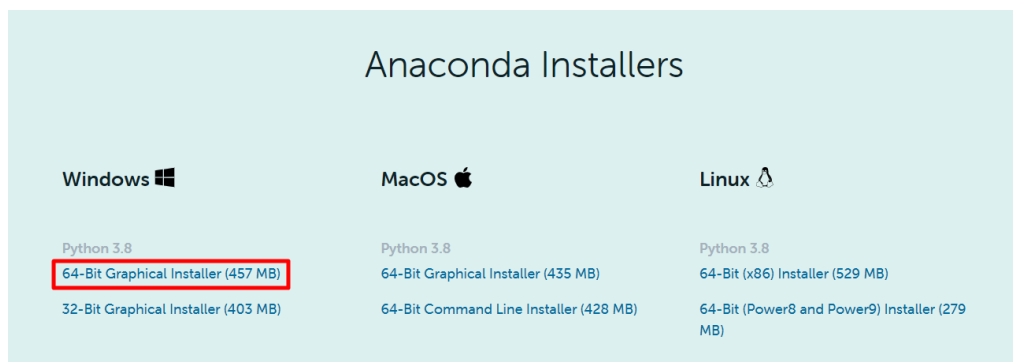
1.1.2. Python 3.7

1.1.3. Anaconda Navigator

## ส่วนที่ 2 ขั้นตอนการใช้งานและการทำงานของโปรแกรมที่เกี่ยวข้อง

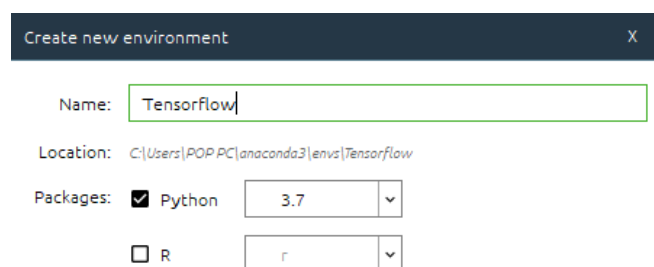
### 2.1. การติดตั้งสภาพแวดล้อมที่จำเป็นโดยใช้ Anaconda Navigator

#### 2.1.1. เข้าเว็บไซต์ และเลือกดาวน์โหลดแอปพลิเคชันสำหรับ Windows 64 bit



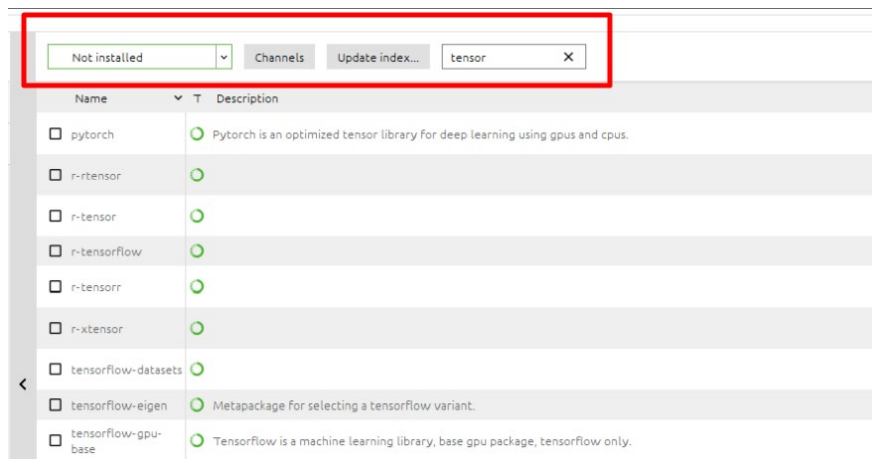
รูปที่ ผ.1 การโหลดแอปพลิเคชัน Anaconda Navigator ผ่านเว็บไซต์

#### 2.1.2. สร้างสภาพแวดล้อมใหม่เลือกเป็น Python เวอร์ชัน 3.7

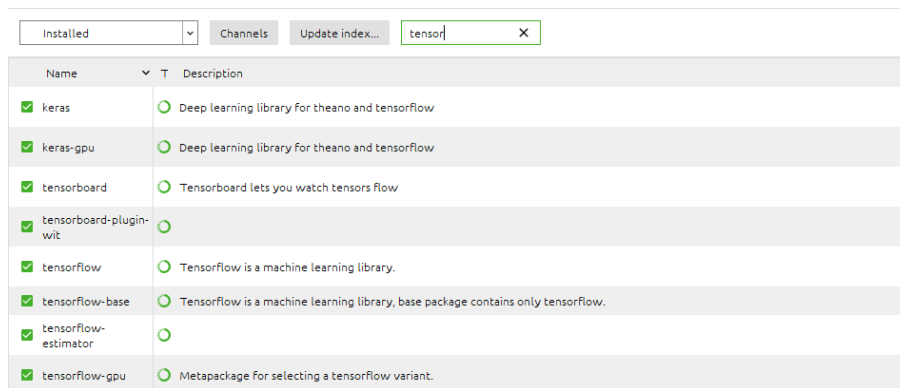


รูปที่ ผ.2 การสร้าง Environment เพื่อใช้งานโปรแกรมทั้งหมดในการทำวิจัย

### 2.1.3. ติดตั้งไลบรารีที่จำเป็น อย่างน้อยจะต้องมี Tensorflow และ Keras จึงจะสามารถทำงานได้

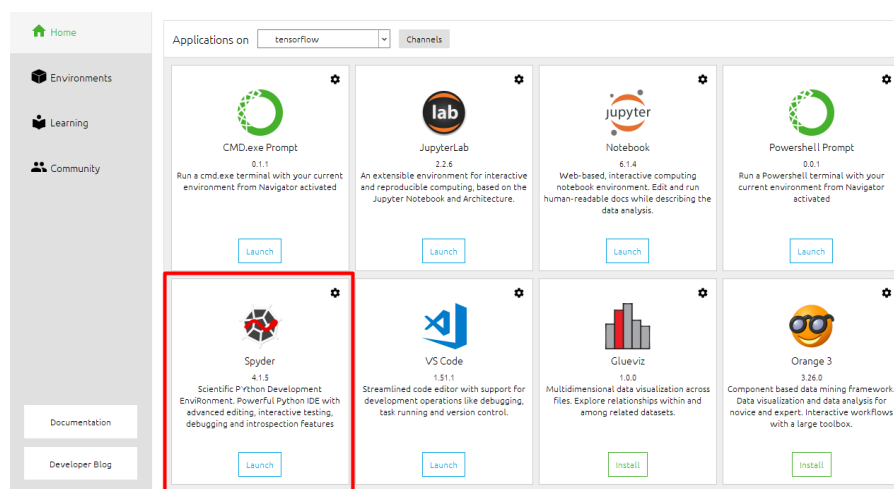


รูปที่ ผ.3 การค้นหาเครื่องมือ Tensorflow และ Keras



รูปที่ ผ.4 รูปไลบรารีที่จำเป็นหลังติดตั้งเสร็จสิ้นแล้ว

### 2.1.3. เมื่อติดตั้งเสร็จ ให้เปิดด้วยโปรแกรม Spyder ผ่านสภาพแวดล้อมที่ Anaconda สร้างเอาไว้



รูปที่ ผ.5 การเปิดแอปพลิเคชัน Spyder ผ่าน Anaconda Navigator

## 2.2. โปรแกรม Packet Generator

### 2.2.1. ทำการแตกไฟล์ Packet Generator.rar

### 2.2.2. กำหนดค่า Parameter ต่างๆที่ใช้ในการสร้างชุดข้อมูล

```
import csv

csv_file_text = "%s.csv" % "train_text"
csv_file_bin = "%s.csv" % "train_binary"
```

รูปที่ ๒.๖ การกำหนดชื่อไฟล์ที่ต้องการ

```
ip_src_all = []
net4 = ipaddress.ip_network('192.168.0.0/16')

for x in net4.hosts():
    ip_src_all.append(str(x))

"""Assign IP Destination Address here"""
ip_dst_all = ['161.246.34.11/24']

"""Assign Port here"""
port_all = ['22', '80']

"""Assign Protocol here"""
protocol_all = ['6', '17']
```

รูปที่ ๒.๗ การกำหนดขอบเขตของ Data Field ที่จะศึกษา

```
# ----- RULES -----
"""Assign Firewall Rule here"""
rule_1 = ['allow', '192.168.0.0/16', '161.246.34.11/24', '80', '6']
rule_2 = ['deny', '192.168.0.0/16', '161.246.34.11/24', '80', '17']
rule_3 = ['deny', '192.168.0.0/16', '161.246.34.11/24', '22', '6']
rule_4 = ['deny', '192.168.0.0/16', '161.246.34.11/24', '22', '17']

# ----- RATIO -----
"""Assign Packet Number Wanted"""
ruleN_1 = 100
ruleN_2 = 100
ruleN_3 = 100
ruleN_4 = 100
default = 0
```

รูปที่ ๒.๘ การกำหนดเงื่อนไขของชุดกฎไฟร์วอลล์และจำนวนข้อมูลในแต่ละกฎ



### 2.2.3. กดคำสั่งเริ่มเพื่อให้โปรแกรมทำงาน

```
def random_packet(): # will fix it later

    src_address = random.choice(ip_src_all)
    src_mask = "255.255.0.0"
    dst_address = str(ipaddress.IPv4Interface(ip_dst_all[0]).ip)
    dst_mask = str(ipaddress.IPv4Interface(ip_dst_all[0]).netmask)
    port = random.choice(port_all)
    protocol = random.choice(protocol_all)

    raw_data_packet = [src_address, src_mask, dst_address, dst_mask, port, protocol]

    return raw_data_packet

def rule_packet_possible(firewall_rule):

    raw_data_packet = []

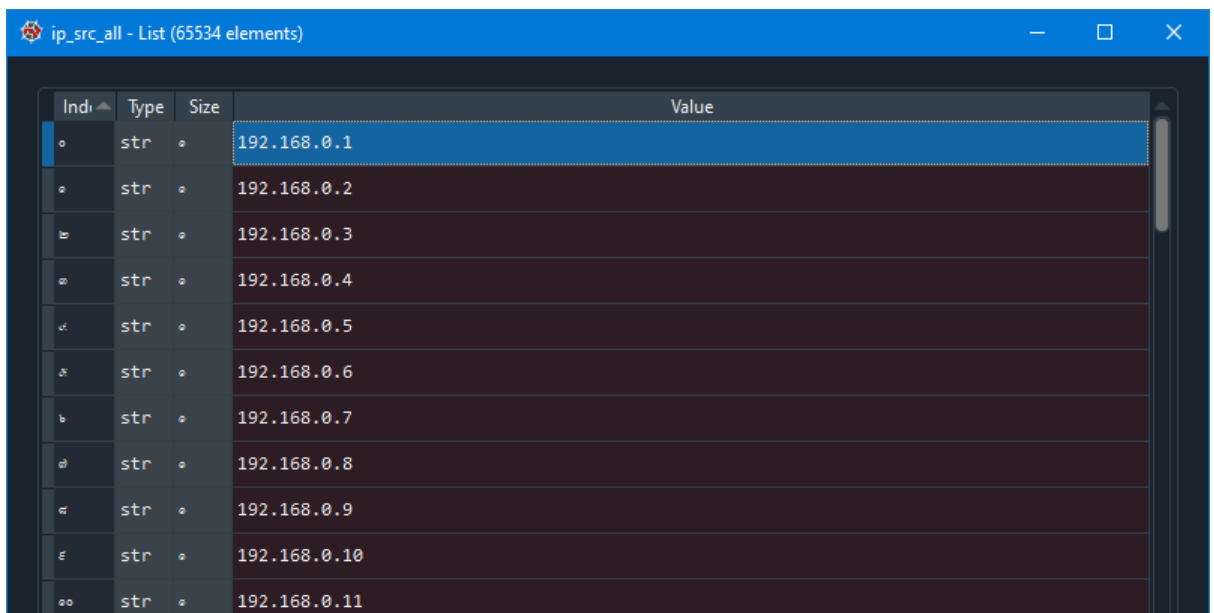
    net4 = ipaddress.ip_network(firewall_rule[1])

    for x in net4.hosts():
        raw_data_packet.append([str(x), '255.255.0.0', str(ipaddress.IPv4Interface(ip_dst_all[0]).ip),
                                str(ipaddress.IPv4Interface(ip_dst_all[0]).netmask), firewall_rule[3], firewall_rule[4]])

    return raw_data_packet
```

รูปที่ ผ.9 โค้ดการทำงานสำหรับการสุ่มชุดข้อมูล

รูปที่ ผ.9 เป็นฟังก์ชันการทำงานโดยการป้อนกฎไฟร์วอลล์เข้าไป แยกส่วนของชุดกฎไฟร์วอลล์มาตีความและสร้างออกมาเป็น List ที่ประกอบไปด้วยชุดข้อมูลที่เป็นไปได้ทั้งหมดของกฎไฟร์วอลล์นั้น โดยจะเก็บเป็นตัวแปรเอาไว้เพื่อใช้หาชุดข้อมูลที่เป็น Default Rule



Indi	Type	Size	Value
0	str	0	192.168.0.1
1	str	0	192.168.0.2
2	str	0	192.168.0.3
3	str	0	192.168.0.4
4	str	0	192.168.0.5
5	str	0	192.168.0.6
6	str	0	192.168.0.7
7	str	0	192.168.0.8
8	str	0	192.168.0.9
9	str	0	192.168.0.10
10	str	0	192.168.0.11

รูปที่ ผ.10 สร้าง List ที่ประกอบไปด้วยจำนวนข้อมูลที่เป็นไปได้ทั้งหมดในกฎไฟร์วอลล์นั้น

rule\_2\_possible - List (65534 elements)

Indi	Type	Size	Value
๐	list	๖	['192.168.0.1', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๑	list	๖	['192.168.0.2', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๒	list	๖	['192.168.0.3', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๓	list	๖	['192.168.0.4', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๔	list	๖	['192.168.0.5', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๕	list	๖	['192.168.0.6', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๖	list	๖	['192.168.0.7', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๗	list	๖	['192.168.0.8', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๘	list	๖	['192.168.0.9', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๙	list	๖	['192.168.0.10', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๑๐	list	๖	['192.168.0.11', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]
๑๑	list	๖	['192.168.0.12', '255.255.0.0', '161.246.34.11', '255.255.255.0', '80', ...]

รูปที่ ๗.11 ตัวอย่างของชุดข้อมูลที่ได้มาจากการสุ่ม

```
#----- raw train data set from rule -----

rule_1_possible = rule_packet_possible(rule_1)
rule_1_quota = [] # use this as output
for i in range(ruleN_1):
    temp = [rule_1[0]] + random.choice(rule_1_possible)
    rule_1_quota.append(temp)

rule_2_possible = rule_packet_possible(rule_2)
rule_2_quota = [] # use this as output
for i in range(ruleN_2):
    temp = [rule_2[0]] + random.choice(rule_2_possible)
    rule_2_quota.append(temp)

rule_3_possible = rule_packet_possible(rule_3)
rule_3_quota = [] # use this as output
for i in range(ruleN_3):
    temp = [rule_3[0]] + random.choice(rule_3_possible)
    rule_3_quota.append(temp)

rule_4_possible = rule_packet_possible(rule_4)
rule_4_quota = [] # use this as output
for i in range(ruleN_4):
    temp = [rule_4[0]] + random.choice(rule_4_possible)
    rule_4_quota.append(temp)
```

รูปที่ ๗.12 กำหนด List ทั้งหมดที่ประกอบไปด้วยชุดข้อมูลไฟร์วอลล์ที่เป็นไปได้

รูปที่ ผ.12 เป็นการเรียกใช้ฟังก์ชันจาก รูป ผ.9 ซ้ำๆกัน แต่มาจากแต่ละกฎไฟร์วอลล์ ซึ่งในแต่ละกฎจะได้ตัวแปรอีกตัวหนึ่งซึ่งเป็น List ที่ใช้เก็บจำนวนโควต้าของชุดข้อมูลที่จะสร้างขึ้น โดยเราได้กำหนดไว้ให้แต่ละกฎในรูป ผ.8

Indi	Type	Size	Value
0	list	4	['deny', '192.168.245.249', '255.255.0.0', '161.246.34.11', '255.255.2 ...
1	list	4	['deny', '192.168.233.86', '255.255.0.0', '161.246.34.11', '255.255.25 ...
2	list	4	['deny', '192.168.209.187', '255.255.0.0', '161.246.34.11', '255.255.2 ...
3	list	4	['deny', '192.168.2.21', '255.255.0.0', '161.246.34.11', '255.255.255. ...
4	list	4	['deny', '192.168.84.226', '255.255.0.0', '161.246.34.11', '255.255.25 ...
5	list	4	['deny', '192.168.51.207', '255.255.0.0', '161.246.34.11', '255.255.25 ...
6	list	4	['deny', '192.168.51.161', '255.255.0.0', '161.246.34.11', '255.255.25 ...
7	list	4	['deny', '192.168.49.235', '255.255.0.0', '161.246.34.11', '255.255.25 ...
8	list	4	['deny', '192.168.199.157', '255.255.0.0', '161.246.34.11', '255.255.2 ...
9	list	4	['deny', '192.168.42.178', '255.255.0.0', '161.246.34.11', '255.255.25 ...
10	list	4	['deny', '192.168.195.132', '255.255.0.0', '161.246.34.11', '255.255.2 ...
11	list	4	['deny', '192.168.249.102', '255.255.0.0', '161.246.34.11', '255.255.2 ...

รูปที่ ผ.13 ตัวอย่าง List ที่มีจำนวนชุดข้อมูลฝึกสอนตามโควต้าที่กำหนดไว้ในแต่ละกฎไฟร์วอลล์

```
#----- merge all packet in rule to check default-----
all_rule_possible = rule_1_possible + rule_2_possible + rule_3_possible + rule_4_possible

#----- raw train data set from universe -----
default_quota = []
while True:
    rand = random_packet()
    if len(default_quota) == default:
        break
    elif rand not in all_rule_possible:
        temp = ["deny"] + rand
        default_quota.append(temp)

#----- merge list of all train set -----

data_set_text = rule_1_quota + rule_2_quota + rule_3_quota + rule_4_quota + default_quota
train_set_text = rule_1_quota + rule_2_quota + rule_3_quota + rule_4_quota + default_quota
random.shuffle(train_set_text)
```

รูปที่ ผ.14 คัดกรอง Default โดยข้อมูลต้องอยู่นอกขอบเขตของกฎไฟร์วอลล์ที่กำหนดจากรูป ผ.12

รูปที่ ผ.14 เป็นการรวม List ที่ประกอบไปด้วยชุดข้อมูลที่เข้าเงื่อนไขกฎไฟร์วอลล์ที่กำหนด และเริ่มสุ่มชุดข้อมูลทีมาจาก Default Rule ในส่วนนี้ต้องมีการทำงานเป็นลูป เนื่องจากเราไม่ทราบ ว่าข้อมูลฝึกสอนที่ทำการสุ่มได้ออกมาอยู่ในเงื่อนไขกฎไฟร์วอลล์หรือไม่ ถ้าหากอยู่ในเงื่อนไขก็ทำการสุ่มใหม่ โดยจะทำซ้ำไปเรื่อยๆจนได้ชุดข้อมูลที่อยู่นอกเงื่อนไขตามจำนวนที่กำหนด และรวมเข้ากับโควต้าของชุดข้อมูลฝึกสอน

```
#----- binary convert -----
train_set_binary = []

for train_packet in train_set_text:
    binary_a_packet = []
    if train_packet[0] == 'allow':
        binary_a_packet.append('1')
    else:
        binary_a_packet.append('0')
    for j in range(1, 5):
        ip = train_packet[j]
        list_octet = [bin(int(x)+256)[3:] for x in ip.split('.')]
        binary_a_packet.append(list_octet[0])
        binary_a_packet.append(list_octet[1])
        binary_a_packet.append(list_octet[2])
        binary_a_packet.append(list_octet[3])
    binary_a_packet.append(bin(int(train_packet[5])+65536)[3:])
    binary_a_packet.append(bin(int(train_packet[6])+256)[3:])
    # train_packet[6]
    train_set_binary.append(binary_a_packet)
```

รูปที่ ผ.14 รวมชุดฝึกสอนที่อยู่ในจำนวนโควต้าที่กำหนด ทำเป็นเลขฐานสอง

train_set_text - List (4000 elements)				
Indi	Type	Size	Value	
๐	list	๘	['deny', '192.168.33.154', '255.255.0.0', '161.246.34.11', '255.255.25 ...	
๑	list	๘	['deny', '192.168.0.153', '255.255.0.0', '161.246.34.11', '255.255.255 ...	
๒	list	๘	['allow', '192.168.249.39', '255.255.0.0', '161.246.34.11', '255.255.2 ...	
๓	list	๘	['allow', '192.168.218.215', '255.255.0.0', '161.246.34.11', '255.255. ...	
๔	list	๘	['allow', '192.168.252.10', '255.255.0.0', '161.246.34.11', '255.255.2 ...	
๕	list	๘	['allow', '192.168.167.45', '255.255.0.0', '161.246.34.11', '255.255.2 ...	
๖	list	๘	['allow', '192.168.197.171', '255.255.0.0', '161.246.34.11', '255.255. ...	
๗	list	๘	['deny', '192.168.226.36', '255.255.0.0', '161.246.34.11', '255.255.25 ...	
๘	list	๘	['deny', '192.168.72.27', '255.255.0.0', '161.246.34.11', '255.255.255 ...	

รูปที่ ผ.15 รวมชุดข้อมูลฝึกสอนที่เลือกมาแล้ว ประกอบไปด้วยทุกกฎไฟร์วอลล์ที่กำหนด

Ind	Type	Size	Value
0	list	๑๕	['0', '11000000', '10101000', '00100001', '10011010', '11111111', '111 ...
๑	list	๑๕	['0', '11000000', '10101000', '00000000', '10011001', '11111111', '111 ...
๒	list	๑๕	['1', '11000000', '10101000', '11111001', '00100111', '11111111', '111 ...
๓	list	๑๕	['1', '11000000', '10101000', '11011010', '11010111', '11111111', '111 ...
๔	list	๑๕	['1', '11000000', '10101000', '11111100', '00001010', '11111111', '111 ...
๕	list	๑๕	['1', '11000000', '10101000', '10100111', '00101101', '11111111', '111 ...
๖	list	๑๕	['1', '11000000', '10101000', '11000101', '10101011', '11111111', '111 ...
๗	list	๑๕	['0', '11000000', '10101000', '11100010', '00100100', '11111111', '111 ...
๘	list	๑๕	['0', '11000000', '10101000', '01001000', '00011011', '11111111', '111 ...

รูปที่ ๑.16 แปลงชุดข้อมูลฝึกสอนเป็นเลขฐานสอง

```
#----- csv write -----
with open(csv_file_text, 'w', newline='') as myfile:
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    wr.writerow(["Action", "Source address", "Source Mask", "Destination address", "Destination Mask", "Port", "Protocol"])

    for i in train_set_text:
        wr.writerow(i)

with open(csv_file_bin, 'w', newline='') as myfile:
    column = ["Act", "src_a1", "src_a2", "src_a3", "src_a4", "src_m1", "src_m2", "src_m3", "src_m4", "dst_a1", "dst_a2", "dst_a3", "dst_a4"]
    wr = csv.writer(myfile, quoting=csv.QUOTE_ALL)
    wr.writerow(column)

    for i in train_set_binary:
        wr.writerow(i)
```

รูปที่ ๑.17 นำชุดข้อมูลฝึกสอนทั้งหมด บันทึกลงในไฟล์ CSV

2.2.4. เมื่อโปรแกรมทำงานเสร็จสิ้น จะได้ไฟล์ชุดข้อมูลนามสกุล .CSV พร้อมรายงานสรุปออกมา

```
In [13]: runfile('C:/Users/POP PC/Documents/GitHub/AI-Firewall-
Training-set-Researching/beta 0.6/1_Packet Gen 4 rule.py', wdir='C:/
Users/POP PC/Documents/GitHub/AI-Firewall-Training-set-Researching/
beta 0.6')
SUMMARY:
Packet Created: 400 packets
Time used: 15.022166013717651 seconds
```

รูปที่ ๑.18 โปรแกรมสร้างชุดข้อมูลรายงานผลสรุปและเวลาที่ใช้

## 2.3. โปรแกรมฝึกโมเดลหรือเครื่องมือโครงข่ายประสาทเทียมเชิงลึก

### 2.3.1. กำหนดตัวแปรต่างๆที่จำเป็นต้องใช้ในการเรียนรู้ของโมเดล

```
# 1 insert local variable here

# File Configuration
csv_file_input = "train_binary" # place the name of data here
csv_file_use = "%s.csv" % csv_file_input

# Model Configuration
node_layer_1 = 150
node_layer_2 = 150
node_layer_3 = 150
epoch = 50

name_model = "model_test" # place the name of model here
name_model_use = "%s.h5" % name_model
```

รูปที่ ๒.19 การกำหนดตัวแปรต่างๆที่ใช้ในการเรียนรู้ของโมเดล

### 2.3.2. กดคำสั่งเริ่มเพื่อให้โปรแกรมทำงาน

```
import pandas as pd
import numpy as py

data = pd.read_csv(csv_file_use)

train_x = data.iloc[:,1:data.shape[1]].values
train_y = data.iloc[:,0].values

train_x = train_x.astype('float32')

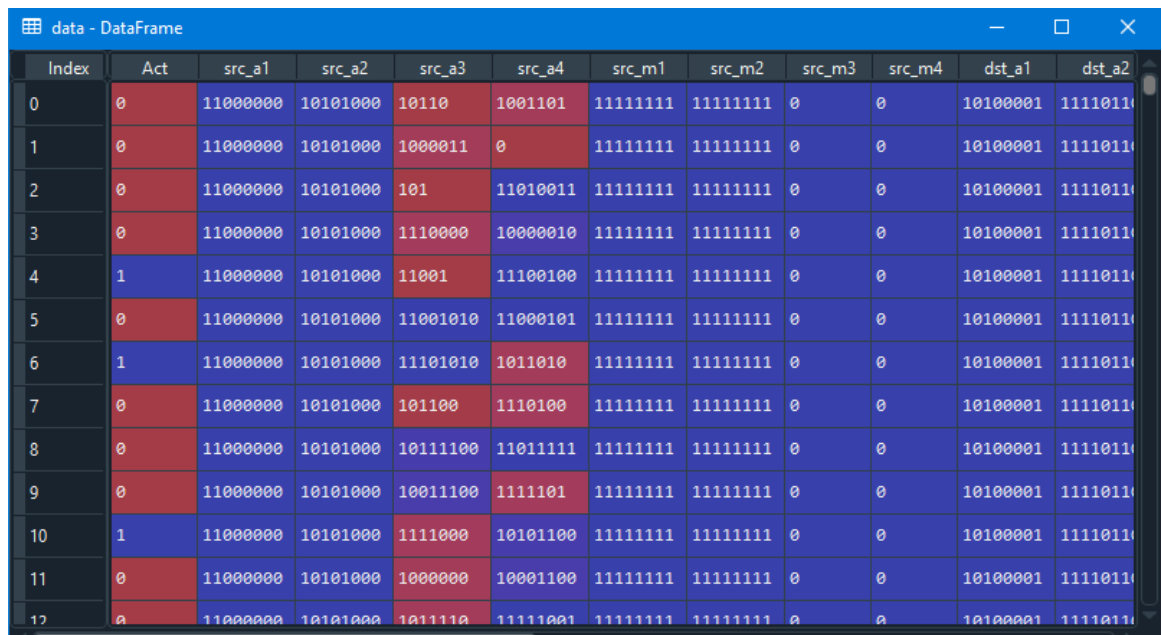
import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from keras.optimizers import adam

model = Sequential()

model.add(Dense(node_layer_1, activation='relu', input_shape = (data.shape[1]-1,)))
model.add(Dense(node_layer_2, activation='relu'))
model.add(Dense(node_layer_3, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer="adam", loss='binary_crossentropy', metrics=['accuracy'])
```

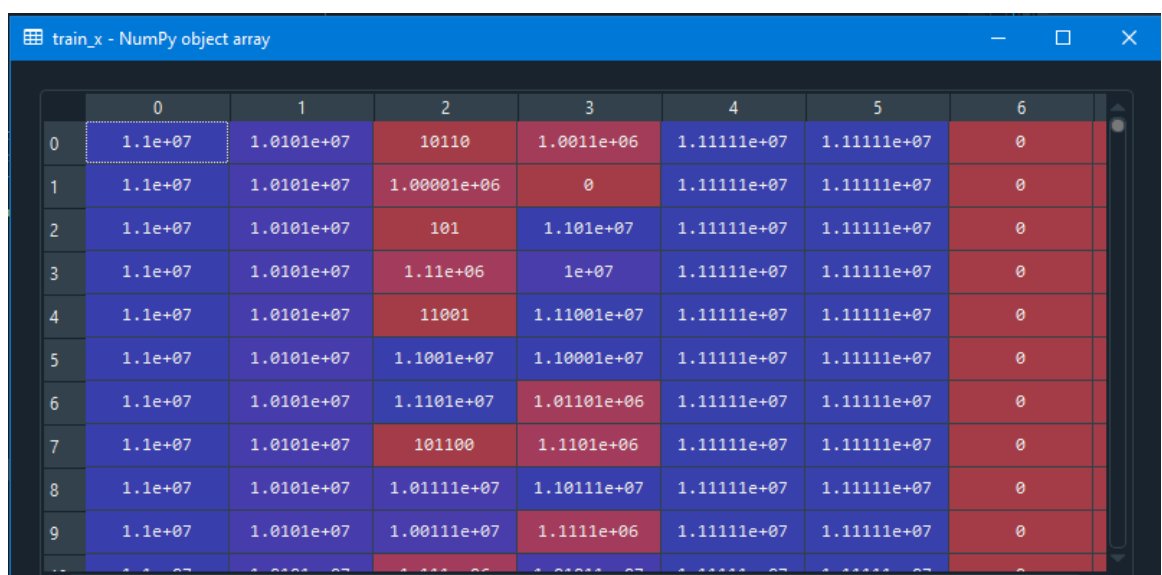
รูปที่ ๒.20 โค้ดกระบวนการออกแบบโครงสร้างภายในโมเดล

ในรูปที่ ผ.20 เป็นการตั้งค่าการทำงานและการเรียนรู้ของโมเดล โดยส่วนใหญ่ได้อิงการตั้งค่าแบบ Default และ Rule of Thumb จากปัญญาประดิษฐ์ที่มีข้อมูลและรูปแบบการทำนายที่เหมือนกัน ส่วนที่เป็นการตั้งค่าจะถูกกำหนดไว้ในรูป ผ.19 โดยในส่วนของโค้ดจะเป็นการเรียกใช้งานโมดูล Keras และออกแบบสร้างโมเดลตามจำนวนโหนดและชั้นที่กำหนด



Index	Act	src_a1	src_a2	src_a3	src_a4	src_m1	src_m2	src_m3	src_m4	dst_a1	dst_a2
0	0	11000000	10101000	10110	1001101	11111111	11111111	0	0	10100001	11110111
1	0	11000000	10101000	1000011	0	11111111	11111111	0	0	10100001	11110111
2	0	11000000	10101000	101	11010011	11111111	11111111	0	0	10100001	11110111
3	0	11000000	10101000	1110000	10000010	11111111	11111111	0	0	10100001	11110111
4	1	11000000	10101000	11001	11100100	11111111	11111111	0	0	10100001	11110111
5	0	11000000	10101000	11001010	11000101	11111111	11111111	0	0	10100001	11110111
6	1	11000000	10101000	11101010	1011010	11111111	11111111	0	0	10100001	11110111
7	0	11000000	10101000	101100	1110100	11111111	11111111	0	0	10100001	11110111
8	0	11000000	10101000	10111100	11011111	11111111	11111111	0	0	10100001	11110111
9	0	11000000	10101000	10011100	1111101	11111111	11111111	0	0	10100001	11110111
10	1	11000000	10101000	1111000	10101100	11111111	11111111	0	0	10100001	11110111
11	0	11000000	10101000	1000000	10001100	11111111	11111111	0	0	10100001	11110111
12	0	11000000	10101000	1011110	11111001	11111111	11111111	0	0	10100001	11110111

รูปที่ ผ.21 List ตัวแปรที่ดึงมาจากไฟล์ CSV ที่ประกอบด้วยชุดข้อมูลฝึกสอน



	0	1	2	3	4	5	6
0	1.1e+07	1.0101e+07	10110	1.0011e+06	1.11111e+07	1.11111e+07	0
1	1.1e+07	1.0101e+07	1.00001e+06	0	1.11111e+07	1.11111e+07	0
2	1.1e+07	1.0101e+07	101	1.101e+07	1.11111e+07	1.11111e+07	0
3	1.1e+07	1.0101e+07	1.11e+06	1e+07	1.11111e+07	1.11111e+07	0
4	1.1e+07	1.0101e+07	11001	1.11001e+07	1.11111e+07	1.11111e+07	0
5	1.1e+07	1.0101e+07	1.1001e+07	1.10001e+07	1.11111e+07	1.11111e+07	0
6	1.1e+07	1.0101e+07	1.1101e+07	1.01101e+06	1.11111e+07	1.11111e+07	0
7	1.1e+07	1.0101e+07	101100	1.1101e+06	1.11111e+07	1.11111e+07	0
8	1.1e+07	1.0101e+07	1.01111e+07	1.10111e+07	1.11111e+07	1.11111e+07	0
9	1.1e+07	1.0101e+07	1.00111e+07	1.1111e+06	1.11111e+07	1.11111e+07	0

รูปที่ ผ.22 ผลลัพธ์การหาค่าน้ำหนักจากการแปลงข้อมูล Data Field

train_y - NumPy object array	
	0
0	0
1	0
2	0
3	0
4	1
5	0

รูปที่ ผ.23 ส่วนของ Field Decision ที่แบ่งออกมาใช้ในการอ้างอิงผลลัพธ์และฝึกสอน

```
import time
print("Training . . . . .")
time_start = time.time()
# Training phase
model.fit(train_x, train_y, epochs = epoch)
# End count training time
time_finish = time.time()
time_duration = time_finish - time_start
```

รูปที่ ผ.24 โค้ดการจับเวลา และการเริ่มโมเดลให้ทำเรียนรู้จากชุดข้อมูล

เนื่องจากเวลาที่ใช้ในการฝึกสอน เป็นผลลัพธ์ที่สำคัญในเชิงเปรียบเทียบประสิทธิภาพ จึงจำเป็นต้องมีการจับเวลาตั้งแต่เริ่มฝึกโมเดล และหยุดจับเวลาเมื่อโมเดลมีการรายงานผลลัพธ์การฝึกสอนโมเดล

```
# Do summary of training
model.summary()
score, acc = model.evaluate(train_x, train_y)
print("Training time:", str(time_duration) + " Seconds")
print('Train score:', score)
print('Train accuracy:', acc)
model.save(name_model_use)
```

รูปที่ ผ.25 โค้ดการรายงานและสรุปผลการเรียนรู้ของโมเดล



2.3.3. เมื่อโปรแกรมทำงานเสร็จสิ้น จะได้โมเดลที่มีไฟล์นามสกุล .h5 พร้อมรายงานสรุป

```
400/400 [=====] - 0s 317us/sample - loss:
1532.9993 - accuracy: 0.7825
Training time: 7.046859502792358 Seconds
Train score: 1532.9993408203125
Train accuracy: 0.7825
```

รูปที่ ผ.26 โปรแกรมรายงานผลการฝึกสอนโมเดลหลังบันทึกโมเดล

2.4. ขั้นตอนการใช้งานโปรแกรมตรวจสอบความแม่นยำโมเดล

2.4.1. กำหนดตัวแปร ที่ประกอบไปด้วยชื่อไฟล์และชุดข้อมูลทดสอบที่สร้างขึ้น

```
# File Configuration

csv_file_input = "test_4rule_bin" # place the name of data here
csv_file_use = "%s.csv" % csv_file_input

name_model = "model_test" # place the name of model here
name_model_use = "%s.h5" % name_model
```

รูปที่ ผ.27 การกำหนดตัวแปรต่างๆที่ใช้ในกระบวนการตรวจสอบโมเดล

2.4.2. กดคำสั่งเริ่มเพื่อให้โปรแกรมทำงาน

```
true_positive = 0
true_negative = 0
false_positive = 0
false_negative = 0

import pandas as pd
import numpy as np

data = pd.read_csv(csv_file_use)

test_x = data.iloc[:,1:data.shape[1]].values
test_y = data.iloc[:,0].values

import keras
from tensorflow.keras.models import load_model

model = load_model(name_model_use)
```

รูปที่ ผ.28 การตั้งตัวแปรและโหลดโมเดลที่จะนำมาทดสอบ

```

import time
time_start = time.time()

# prediction = model.evaluate(test_x)
prediction = model.predict(test_x)

# Compare Reference
for i in range(len(prediction)):
    if round(prediction[i][0]) == int(test_y[i]):
        if round(prediction[i][0]) == 1:
            true_positive += 1
        elif round(prediction[i][0]) == 0:
            true_negative += 1

    elif round(prediction[i][0]) != int(test_y[i]):
        if round(prediction[i][0]) == 1:
            false_positive += 1
        elif round(prediction[i][0]) == 0:
            false_negative += 1

time_finish = time.time()
time_duration = time_finish - time_start

```

รูปที่ ผ.29 การจับเวลา การทำนายผลที่อิงตาม Reference Variant Set

```

# Accuracy
test_accuracy = float(true_positive + true_negative) / len(prediction)
loss_rate = float(false_positive + false_negative) / len(prediction)

print("Number of Packet: ",len(prediction))
print("Compare Time: %.6f seconds" % time_duration)
print("Accuracy of testing: " + str(test_accuracy*100) + " %")
print("Loss rate: " + str(loss_rate*100) + " %")
print("TP:", true_positive, "TN:", true_negative, "FP:", false_positive, "FN:", false_negative)

```

รูปที่ ผ.30 การสรุปผลลัพธ์ความแม่นยำในการทำนายของโมเดล

2.4.3. เมื่อโปรแกรมทำงานเสร็จสิ้น จะได้รายงานสรุปความถูกต้องของโมเดลที่ทำการตรวจสอบ

```

In [12]: runfile('C:/Users/POP PC/Documents/GitHub/AI-Firewall-
Training-set-Researching/beta 0.6/3_Evaluate.py', wdir='C:/Users/POP
PC/Documents/GitHub/AI-Firewall-Training-set-Researching/beta 0.6')
Evaluating . . . . .
Number of Packet: 40000
Compare Time: 1.555670 seconds
Accuracy of testing: 74.9175 %
Loss rate: 25.082500000000003 %
TP: 7450 TN: 22517 FP: 7483 FN: 2550

```

รูปที่ ผ.31 โปรแกรมรายงานผลสรุปความถูกต้องจากการทดสอบโมเดล

## ประวัติผู้เขียน

ชื่อ – นามสกุล นาย จูติโชติ ใจเมือง

รหัสนักศึกษา 60070019

วัน เดือน ปีเกิด 7 พฤศจิกายน 2541

ประวัติการศึกษา

วุฒิม.6 โรงเรียนเตรียมอุดมศึกษาพัฒนาการ

ภูมิลำเนา จังหวัดกรุงเทพมหานคร

เบอร์โทร 08-6778-7397

E-Mail

60070019@it.kmitl.ac.th

สาขาที่จบ วิทยาศาสตร์ - คณิตศาสตร์

รุ่นที่ 34

ปีการศึกษา

2559



ชื่อ – นามสกุล นาย พิพัฒน์บุญ พุทธคุณ

รหัสนักศึกษา 60070065

วัน เดือน ปีเกิด 25 เมษายน 2542

ประวัติการศึกษา

วุฒิม.6 โรงเรียนเซนต์ดอมินิก

ภูมิลำเนา จังหวัดกรุงเทพมหานคร

เบอร์โทร 08-6058-0919

E-Mail

60070065@it.kmitl.ac.th

สาขาที่จบ ศิลป์-คำนวณ

รุ่นที่ 48

ปีการศึกษา

2559

