การปรับปรุงอัลกอริทึมนาอีฟเบย์โดยการลดความสำคัญของคำที่มีความถี่ต่ำพิจารณาจากค่าเอนโทรปีของคำเพื่อการจำแนกประเภทอีเมลสแปม

IMPROVING NAIVE BAYES BY REDUCING THE IMPORTANCE OF LOW-FREQUENCY WORDS BASED ON ENTROPY OF WORDS FOR SPAM EMAIL CLASSIFICATION

ไพบูลย์ ตรีกาญจนานันท์

PHAIBOON TRIKANJANANUN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2566

KMITL-2023-EN-M-060-014

IMPROVING NAIVE BAYES BY REDUCING THE IMPORTANCE OF LOW-FREQUENCY WORDS BASED ON ENTROPY OF WORDS FOR SPAM EMAIL CLASSIFICATION

PHAIBOON TRIKANJANANUN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN INSTRUMENTATION ENGINEERING

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2023

KMITL-2023-EN-M-060-014

หัวข้อวิทยานิพนธ์	การปรับปรุงอัลกอริทึมนาอีฟเบย์โดยการลดความสำคัญของคำที่มีความถี่ต่ำพิจารณาจากค่าเอนโทรปีของคำสำหรับการจำแนกประเภทอีเมลสแปม
นักศึกษา	นายไพบูลย์ ตรีกาญจนานันท์
รหัสประจำตัว	62601249
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมการวัดคุม
พ.ศ.	2566
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ. ดร. อาจินต์ น่วมสำราญ

บทคัดย่อ

อัลกอริทึมนาอีฟเบย์หรือ NB คือหนึ่งในอัลกอริทึมที่นิยมใช้ในการจำแนกอีเมลสแปม เนื่องจากใช้เวลาในการฝึกฝนอย่างรวดเร็วโดยการใช้เทคนิคอย่างง่ายและยังให้ความแม่นยำสูง หนึ่งในงานวิจัยที่ปรับปรุงอัลกอริทึมนาอีฟเบย์คืออัลกอริทึม AWF-NB ในวิทยานิพนธ์นี้เราจะเรียกงานวิจัยดังกล่าวว่าอัลกอริทึม AWF-NB เพื่อความสะดวกในการกล่าวถึง อัลกอริทึม AWF-NB มุ่งเน้นแก้ปัญหาความสำคัญของคำที่เท่าเทียมกันในแต่ละคลาสเพราะในความเป็นจริงไม่ได้เป็นแบบนั้นเสมอไป นี้คืออีกหนึ่งปัญหาของอัลกอริทึม NB แบบดั้งเดิมเพื่อที่จะแก้ปัญหานี้อัลกอริทึม AWF-NB จะทำการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าให้ต่ำลงอย่างมาก เพื่อให้คำในคลาสที่มีความสำคัญน้อยกว่ามีบทบาทน้อยลงอย่างมากในการจำแนก อย่างไรก็ตามการดำเนินการนี้จะทำให้ความแม่นยำลดลงอย่างมากในกรณีที่ความสำคัญของคำในแต่ละคลาสแตกต่างกันเพียงเล็กน้อยเท่านั้น ดังนั้นวัตถุประสงค์ของวิทยานิพนธ์นี้คือมุ่งเน้นเพื่อปรับปรุงอัลกอริทึม AWF-NB โดยการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าพิจารณาจากค่าเอนโทรปีของคำในแต่ละคลาส วิทยานิพนธ์นี้จะคำนวณค่าเอนโทรปีของคำเพื่อตัดสินใจว่าควรจะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าลงหรือไม่ ผลการทดลองใน 15 ชุดข้อมูลอีเมลสแปมจากเว็บไซต์ Kaggle แสดงให้เห็นว่าอัลกอริทึมที่เสนอหรืออัลกอริทึม RIWE-NB สามารถเพิ่มความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดั้งเดิมและ AWF-NB ในชุดข้อมูลส่วนมากในขณะที่เวลาในการดำเนินการยังคงถูกรักษาไว้ในระดับที่ใกล้เคียงกัน

Thesis	Improving Naive Bayes by Reducing the Importance of Low-Frequency Words Based on Entropy of Words for Spam Email Classification
Student	Mr.Phaiboon Trikanjananun
Student ID.	62601249
Degree	Master of Engineering
Program	Instrumentation Engineering
Year	2023
Thesis Advisor	Assoc.Prof.Dr.Arjin Numsomran

ABSTRACT

The Naive Bayes algorithm (NB algorithm) is a popular one for spam email classification due to fast training, using simple techniques and high accuracy. One of many research improving NB algorithms are the AWF-NB algorithm. In this paper, we call the research an AWF-algorithm for convenient mention. The AWF-NB algorithm focuses on solving the equally important word in each class because it is not always the case. Another problem of the NB algorithm to solve this problem, the AWF-NB extremely reduces the importance of words in the class that has lower importance. However, this action will lead to reducing the accuracy in cases that slightly differ among the importance of words in each class. Therefore, the goal of the research is to improve the AWF-NB algorithm by reducing the importance of words based on entropy of words. We compute the entropy of a word to decide if it should be reduced in importance. The experimental results on ten spam email datasets from Kaggle website indicated that the RIWE-NB algorithm can remarkably increase the classification accuracy of the NB algorithm and the AWF-NB algorithm in majority datasets while the execution time is still conserved

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สำเร็จได้ด้วยความกรุณาจากอาจารย์ที่ปรึกษา รศ.ดร.อาจินต์ น่วมสำราญ และ รศ.ดร.วิทยา ทิพย์สุวรรณพร ที่ให้ความช่วยเหลือ ให้คำชี้แนะช่วยแก้ปัญหา ตลอดจนให้ความรู้และประสบการณ์ที่ดีแก่ข้าพเจ้า

ขอขอบพระคุณกรรมการสอบหัวข้อและโครงร่างวิทยานิพนธ์ทุกท่านที่ได้กรุณาให้คำแนะนำตลอดจนข้อชี้แนะ จนในที่สุดทำให้วิทยานิพนธ์นี้สำเร็จลงได้

สุดท้ายต้องขอขอบคุณภรรยาของข้าพเจ้า ที่คอยสนับสนุนและเป็นกำลังใจให้กับข้าพเจ้าในการทำเล่มวิทยานิพนธ์ จนกระทั่งทำให้วิทยานิพนธ์นี้เสร็จสมบูรณ์ไปด้วยดี

สำหรับคุณงามความดีอันใดที่เกิดจากวิทยานิพนธ์นี้ ข้าพเจ้าขอมอบให้กับบิดามารดา ซึ่งเป็นที่รักและเคารพยิ่ง ตลอดจนครูอาจารย์ที่เคารพทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้และถ่ายทอดประสบการณ์ที่ดีให้แก่ข้าพเจ้า

ไพบูลย์ ตรีกาญจนานันท์

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

อัลกอริทึมนาอีฟเบย์หรืออัลกอริทึม NB (Naive bayes algorithm) ถูกใช้อย่างแพร่หลายในการจำแนกประเภท เนื่องจากมีการฝึกฝนอย่างรวดเร็วโดยใช้เทคนิคอย่างง่ายและยังให้ความแม่นยำสูง ทำให้เป็นที่นิยมในการจำแนกประเภทอีเมลสแปม (Spam email) และการจำแนกประเภทข้อความ นอกจากนี้อัลกอริทึม NB ยังใช้ชุดข้อมูลสำหรับฝึกฝนขนาดเล็กและสามารถจัดการกับปัญหาการจำแนกประเภทแบบหลายคลาสได้ [1], [2] อย่างไรก็ตามในอัลกอริทึม NB มีปัญหาที่ร้ายแรงอยู่เนื่องจากอัลกอริทึม NB ใช้ทฤษฎีเบย์ (Bayes theorem) [3] ซึ่งเป็นความน่าจะเป็น ดังนั้นถ้าต้องจำแนกอีเมลที่มีคำที่ไม่ปรากฏในชุดข้อมูลสำหรับฝึกฝน ความน่าจะเป็นจะมีค่าเป็นศูนย์ ส่งผลให้ความแม่นยำในการจำแนกลดลง ซึ่งปัญหานี้โดยปกติจะใช้วิธีการทำลาปลาซสมูททิง (Laplace smoothing) [4] เพื่อแก้ไขปัญหานี้ในอัลกอริทึม NB และด้วยข้อดีของอัลกอริทึม NB ดังที่กล่าวไว้ วิทยานิพนธ์นี้จึงมุ่งเน้นไปที่การปรับปรุงอัลกอริทึม NB เพื่อนำมาประยุกต์ใช้กับการจำแนกประเภทอีเมลสแปม มีงานวิจัยหลายงานที่ประยุกต์ใช้อัลกอริทึมที่หลากหลายเพื่อการจำแนกประเภทอีเมลสแปม

Tuteja และ Bogiri [5] ใช้อัลกอริทึม BPNN (Back Propagation Neural Network) เพื่อจำแนกประเภทอีเมลสแปม แม้ว่าความแม่นยำจะสูงแต่อัลกอริทึม BPNN ใช้เวลาในการฝึกฝนค่อนข้างมากและอัลกอริทึมมีความซับซ้อนในการพัฒนา

Vishagini และ Rajan [6] ได้ทำการพัฒนาและปรับปรุง SVM แบบถ่วงน้ำหนัก (Support Vector Machine) โดยใช้ค่าน้ำหนักจากอัลกอริทึม KFCM วิธีการนี้สามารถลดอัตราการจำแนกประเภทที่ผิดพลาดได้มากกว่า SVM แบบธรรมดาและ SVM แบบถ่วงน้ำหนัก แต่อย่างไรก็ตามอัลกอริทึมนี้มีความซับซ้อนและใช้เวลามากกว่าอัลกอริทึม NB

Agarwal และ Kumar [7] ได้รวมอัลกอริทึม PSO (Particle Swarm Optimization) เข้ากับอัลกอริทึม NB เพื่อจำแนกอีเมลสแปม ผลการทดลองแสดงให้เห็นว่าอัลกอริทึมของพวกเขามีความแม่นยำมากกว่าอัลกอริทึม NB ทั่วไป เช่นเดียวกับอัลกอริทึมที่กล่าวถึงข้างต้น การรวม PSO กับ NB ใช้เวลาในการฝึกฝนและทดสอบอย่างมาก มันเป็นแบบจำลองที่ซับซ้อนและยากที่จะปรับปรุง

งานวิจัยต่อไปเป็นเรื่องเกี่ยวกับการปรับปรุงอัลกอริทึม NB สำหรับการจำแนกประเภทข้อความ Guo [8] ประยุกต์ใช้การปรับน้ำหนักผ่านความถี่ในอัตราส่วนของคำกับอัลกอริทึม NB เพื่อแก้ปัญหาความสำคัญของคำที่เท่าเทียมกันในแต่ละคลาสเพราะในความเป็นจริงไม่ได้เป็นแบบนั้นเสมอไป ในวิทยานิพนธ์นี้จะเรียกวิธีการนี้ว่าอัลกอริทึม AWF-NB เพื่อความสะดวกในการกล่าวถึง

จากเอกสารที่ได้รับการตีพิมพ์จากการประชุมทางวิชาการของอัลกอริทึม AWF-NB ผลการทดลองแสดงให้เห็นว่าอัลกอริทึม AWF-NB มีความแม่นยำมากกว่าอัลกอริทึม NB แบบดั้งเดิมในชุดข้อมูลส่วนมาก จะเห็นได้ว่าอัลกอริทึม AWF-NB เป็นอัลกอริทึมที่ไม่ซับซ้อนเพราะแค่ประยุกต์ใช้การปรับน้ำหนักกับอัลกอริทึม NB แบบดั้งเดิม อีกทั้งยังมีการฝึกฝนและการทดสอบที่รวดเร็วเหมือนอัลกอริทึม NB อย่างไรก็ตามการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าให้ต่ำลงอย่างมากนั้น จะส่งผลให้ความแม่นยำลดลงในกรณีที่ความสำคัญของคำในแค่ละคลาสแตกต่างกันเพียงเล็กน้อยเท่านั้น ดังนั้นวิทยานิพนธ์นี้จึงมีแนวคิดที่จะปรับปรุงอัลกอริทึม AWF-NB เพื่อจัดการข้อเสียดังกล่าวและจะเป็นการเพิ่มความแม่นยำของอัลกอริทึม AWF-NB ด้วยวิธีการคำนวณค่าเอนโทรปีของคำเพื่อเป็นเครื่องตัดสินใจว่าควรจะลดความสำคัญในคลาสที่มีความสำคัญต่ำกว่าหรือไม่

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

เสนอวิธีการปรับปรุงอัลกอริทึม AWF-NB เพื่อจัดการกับปัญหาความแม่นยำที่ลดลง เนื่องจากการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าลงอย่างมากในกรณีที่ความสำคัญของคำในแต่ละคลาสแตกต่างกันเพียงเล็กน้อย เพื่อเพิ่มความแม่นยำในการจำแนก และเวลาในการดำเนินการถูกรักษาไว้ในระดับที่ใกล้เคียงกัน

1.3 สมมติฐานของการศึกษา

วิธีการที่เสนอสามารถเพิ่มความแม่นยำในการจำแนกให้มากขึ้นกว่าเดิมได้ เมื่อเทียบกับอัลกอริทึม NB แบบดั้งเดิมและอัลกอริทึม AWF-NB ในขณะที่เวลาดำเนินการยังคงถูกรักษาไว้ในระดับที่ใกล้เคียงกัน

1.4 ขอบเขตของการวิจัย

1) ศึกษาขั้นตอนของอัลกอริทึม NB แบบดั้งเดิม
2) ศึกษาขั้นตอนและปัญหาของอัลกอริทึม AWF-NB หรืออัลกอริทึมที่ต้องการปรับปรุง
3) อัลกอริทึมที่นำเสนอมุ่งเน้นที่จะแก้ปัญหาความแม่นยำที่ลดลงของอัลกอริทึม AWF-NB ในกรณีที่ลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าเพียงเล็กน้อย
4) อัลกอริทึมที่นำเสนอจะถูกทดสอบประสิทธิภาพโดยใช้ชุดข้อมูล 15 ชุดจากเว็บไซต์ Kaggle

1.5 ขั้นตอนของการศึกษา

1) ศึกษาการทำงานของอัลกอริทึม NB แบบดั้งเดิม และทฤษฎีพื้นฐานที่เกี่ยวข้อง
2) ศึกษางานวิจัยที่ต้องการปรับปรุงและปัญหาที่มุ่งเน้น

3) ทำการคิดค้นและพัฒนาอัลกอริทึมเพื่อแก้ปัญหาที่มุ่งเน้น
4) ทดสอบประสิทธิภาพของอัลกอริทึมที่เสนอและปรับปรุงเพื่อผลการทดลองที่ดีขึ้น
5) วิเคราะห์และทำความเข้าใจผลการทดลอง
6) เรียบเรียงและสรุปสิ่งที่ได้ศึกษาวิจัย แล้วจัดทำเล่มวิทยานิพนธ์

1.6 เครื่องมือและอุปกรณ์ที่ใช้ในงานวิจัย

1) แมคบุ๊คส่วนบุคคล ชิป Apple M1 หน่วยความจำหลักขนาด 8 GB
2) ระบบปฏิบัติการ Mac OS Ventura 13
3) โปรแกรม Visual Studio Code

1.7 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์นี้ประกอบไปด้วย 7 บท ซึ่งแต่ละบทมีหัวข้อดังนี้

บทที่ 1 อธิบายถึงความเป็นมาและความสำคัญของปัญหาและวัตถุประสงค์
บทที่ 2 อธิบายทฤษฎีพื้นฐานที่เกี่ยวข้อง
บทที่ 3 นำเสนอเกี่ยวกับงานวิจัยที่เกี่ยวข้อง
บทที่ 4 อธิบายเกี่ยวกับอัลกอริทึมที่ต้องการปรับปรุง
บทที่ 5 อธิบายเกี่ยวกับอัลกอริทึมที่เสนอ
บทที่ 6 อธิบายการวิเคราะห์ผลการทดลอง
บทที่ 7 สรุปและข้อเสนอแนะ

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 เอนโทรปี (Entropy)

ในวิทยานิพนธ์นี้จะใช้การคำนวณค่าเอนโทรปี [9] ในทางทฤษฎีสารสนเทศ (Information theory) ดังนั้นในที่นี้จะอธิบายเอนโทรปีในขอบเขตของทฤษฎีสารสนเทศ เอนโทรปีในทางทฤษฎีสารสนเทศหมายถึงค่าความไม่เป็นระเบียบของข้อมูลมีหน่วยเป็นบิต (Bit) ยิ่งข้อมูลมีการกระจายมากหรือไม่เป็นระเบียบมาก ค่าเอนโทรปีจะยิ่งมาก เอนโทรปีมีสมการดังนี้

$$En(S) = -\sum_{n \in N} p(x_n) \log_2 p(x_n) \tag{2.1}$$

En(S) คือค่าเอนโทรปี

N คือเซตของผลลัพธ์ที่เป็นไปได้ทั้งหมด

$p(x_n)$ คือความน่าจะเป็นที่จะเกิดผลลัพธ์ n หารด้วยจำนวนผลลัพธ์ทั้งหมด

จากสมการค่าเอนโทรปีจะมีค่าสูงสุดที่ $\log_2(N)$ เพื่อให้เข้าใจเรื่องเอนโทรปีมากขึ้นจะอธิบายโดยยกตัวอย่างเหตุการณ์สมมุติในการซื้อกล่องสุ่มสินค้าจากร้านค้า 3 ร้าน

1) ร้านค้า A มีโอกาส 90% ที่จะได้สินค้าที่ต้องการ
2) ร้านค้า B มีโอกาส 50% ที่จะได้สินค้าที่ต้องการ
3) ร้านค้า C มีโอกาส 5% ที่จะได้สินค้าที่ต้องการ

คำนวณค่าเอนโทรปีของร้านค้าทั้ง 3 ร้านเพื่อที่จะทราบว่าร้านค้าไหนที่มีความไม่แน่นอนที่จะได้สินค้าที่ต้องการมากที่สุด ร้านค้าทั้งหมดจะมีเซตของผลลัพธ์ที่เป็นไปได้ทั้งหมด 2 คือ {สุ่มได้สินค้าที่ต้องการ, สุ่มได้สินค้าอื่น} ร้านค้า A มีความน่าจะเป็นที่จะได้สินค้าที่ต้องการ 0.9 และมีความน่าจะเป็นที่จะได้สินค้าอื่น 0.1 ต่อมาร้านค้า B มีความน่าจะเป็นที่จะได้สินค้าที่ต้องการและสินค้าอื่นเท่ากันคือ 0.5 สุดท้ายร้านค้า C มีความน่าจะเป็นที่จะได้สินค้าที่ต้องการ 0.05 และมีความน่าจะเป็นที่จะได้สินค้าอื่น 0.95

คำนวณค่าเอนโทรปีของร้านค้า A

$$En(A) = (0.9 \log_2 0.9) + (0.1 \log_2 0.1) = 0.47 \tag{2.2}$$

คำนวณค่าเอนโทรปีของร้านค้า B

$$En(A) = (0.5 \log_2 0.5) + (0.5 \log_2 0.5) = 1 \tag{2.3}$$

คำนวณค่าเอนโทรปีของร้านค้า C

$$En(A) = (0.05 \log_2 0.05) + (0.95 \log_2 0.95) = 0.29 \quad (2.4)$$

จากการคำนวณพบว่าหากเราซื้อกล่องสุ่มร้าน B จะมีความไม่แน่นอนมากที่สุดเพราะโอกาสที่จะได้สินค้าที่ต้องการคือ 50%:50% ในขณะที่ร้านค้า A มีความไม่แน่นอนรองลงมา เนื่องจากเรามีโอกาสที่จะได้สินค้าที่ต้องการถึง 90% อีก 10% ที่จะได้สินค้าอื่น สุดท้ายร้านค้า C มีความไม่แน่นอนน้อยที่สุด เนื่องจากเห็นได้ชัดว่าเรามีโอกาสที่จะได้สินค้าอื่นถึง 95% และมีโอกาสได้สินค้าที่ต้องการ 5% เท่านั้น ซึ่งตรงกับที่ได้กล่าวไว้ว่ายิ่งข้อมูลหรือเซตของผลลัพธ์กระจายออกจากกันมาก ส่งผลให้ค่าเอนโทรปีจะยิ่งมาก เช่นร้านค้า B ผลลัพธ์ที่เป็นไปได้คือการสุ่มได้สินค้าที่ต้องการและการสุ่มได้สินค้าอื่น ซึ่งกระจายตัวออกจากกันมากที่สุดเป็น 50%:50% ดังนั้นจึงมีค่าเอนโทรปีเท่ากับ 1 ซึ่งเป็นค่าเอนโทรปีสูงสุด $\log_2 2$ เท่ากับ 1 ส่วนร้านค้า A มีการกระจายตัวของผลลัพธ์น้อยกว่าร้านค้า B คือ 90%:10% ดังนั้นค่าเอนโทรปีที่ได้จึงน้อยกว่ามาก ส่วนร้านค้า C มีการกระจายตัวของผลลัพธ์น้อยที่สุดคือ 5%:95% จึงมีค่าเอนโทรปีน้อยที่สุด

2.2 ทฤษฎีเบย์ (Bayes' Theorem)

เป็นทฤษฎีที่อธิบายถึงความสัมพันธ์ระหว่าง P(A|B) และ P(B|A) โดยที่ P(A|B) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ A เมื่อเหตุการณ์ B เกิดขึ้นแล้ว ส่วน P(B|A) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ B เมื่อเหตุการณ์ A เกิดขึ้นแล้ว ทั้งความน่าจะเป็น P(A|B) และ P(B|A) ถูกเรียกว่าความน่าจะเป็นแบบมีเงื่อนไข (Conditional probability) สมการของทฤษฎีเบย์แสดงดังนี้

$$P(B|A) = \frac{P(A|B) \times P(B)}{P(A)} \quad (2.5)$$

โดยที่ P(B|A) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ B เมื่อเหตุการณ์ A เกิดขึ้นแล้ว

P(A|B) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ A เมื่อเหตุการณ์ B เกิดขึ้นแล้ว

P(B) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ B

P(A) คือความน่าจะเป็นที่จะเกิดเหตุการณ์ A

ที่มาของสมการเบย์มาจากการหาความน่าจะเป็นแบบมีเงื่อนไขทั้งสอง ซึ่งก็คือ P(A|B) และ P(B|A) โดย P(A|B) และ P(B|A) มีสมการดังนี้

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad (2.6)$$

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \quad (2.7)$$

ทั้งสมการ 2.6 และ 2.7 มีจุดที่เหมือนกันคือ P(A∩B) ดังนั้นเมื่อย้ายข้างจะได้ P(A∩B) ของทั้งสองสมการดังนี้

$$P(A \cap B) = P(A|B) \times P(B) \tag{2.8}$$

$$P(A \cap B) = P(B|A) \times P(A) \tag{2.9}$$

เมื่อเอาสมการที่ 2.8 และสมการที่ 2.9 มาเท่ากัน และทำการย้ายข้างของสมการ จะได้เป็นสมการของทฤษฎีเบย์หรือสมการที่ 2.5

2.3 อัลกอริทึมนาอีฟเบย์ (Naive Bayes)

อัลกอริทึมนาอีฟเบย์คือหนึ่งในอัลกอริทึมยอดนิยมสำหรับการจำแนกประเภทอีเมลสแปม มีพื้นฐานมาจากทฤษฎีเบย์ด้วยสมมติฐานความเป็นอิสระระหว่างตัวทำนาย [10] จุดแข็งของอัลกอริทึม NB แบบดั้งเดิมคือใช้เวลาในการฝึกฝนอย่างรวดเร็ว ใช้เทคนิคอย่างง่ายไม่ซับซ้อน อีกทั้งยังให้ความแม่นยำในการจำแนกที่สูง นอกจากนี้อัลกอริทึม NB ยังใช้ชุดข้อมูลสำหรับฝึกฝนขนาดเล็กและสามารถจัดการกับปัญหาการจำแนกประเภทแบบหลายคลาสได้

2.3.1 ขั้นตอนของอัลกอริทึม NB ในการจำแนกประเภทอีเมลสแปม

1) กำหนดให้ X คือเซตของคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท และกำหนดให้ x เป็นสมาชิกหรือคำแต่ละคำในเซต X เขียนเป็นสัญลักษ์ดังนี้ X = { x | x คือคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท }
2) คำนวณความน่าจะเป็นของอีเมลสแปมและอีเมลแฮม (อีเมลที่ไม่ใช่สแปม) ด้วยสมการที่ 2.10

$$P(C|X) = P(C) \prod_{x \in X} P(x|C) \tag{2.10}$$

โดยที่ P(C|X) คือความน่าจะเป็นที่จะเป็นคลาส C (อีเมลสแปม หรืออีเมลแฮม) เมื่อมีคำทั้งหมด (X) ในอีเมล,

X คือเซตของคำทั้งหมด,

P(C) คือความน่าจะเป็นของคลาส C,

P(x|C) คือความน่าจะเป็นของคำ x ที่เป็นคลาส C

$$P(C) = \frac{w_C}{W} \tag{2.11}$$

โดยที่ w_C คือจำนวนคำทั้งหมดในคลาส C,
W คือจำนวนคำทั้งหมด

$$P(x|C) = \frac{w_{x,C}}{w_C} \quad (2.12)$$

โดยที่ $w_{x,C}$ คือจำนวนคำ x ทั้งหมดในคลาส C

3) เปรียบเทียบความน่าจะเป็นที่จะเป็นอีเมลสแปม P(S|X) กับความน่าจะเป็นที่จะเป็นอีเมลแฮม P(H|X) ถ้า P(S|X) มากกว่า P(H|X) จำแนกอีเมลว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

```
1  function NB (email) returns classification
       words ← List of word in email
       pa_spam ← 1
       pa_ham ← 1
       for each value w of words do
           pw_spam ← Compute the probability of word w in spam email
7          pw_ham ← Compute the probability of word w in ham email
8          pa_spam ← pa_spam * pw_spam
           pa_ham ← pa_ham * pw_ham
       PS ← Compute the probability of spam using pa_spam
       PH ← Compute the probability of ham using pa_ham
       if PS is greater than PH,
           then return spam class
       else
           return ham class
```

รูปที่ 2.1 แสดงรหัสเทียมของอัลกอริทึม NB

จากรหัสเทียมของอัลกอริทึม NB ในรูปที่ 2.1 ฟังก์ชัน NB จะคืนค่าผลการจำแนกประเภทอีเมล โดยจะรับพารามิเตอร์เป็นอีเมลที่ต้องการจำแนกประเภท ตัวแปร words จะเก็บรายการของคำทั้งหมดในอีเมล pa_spam คือผลคูณความน่าจะเป็นของคำทั้งหมดในอีเมลสแปม ส่วน pa_ham คือผลคูณความน่าจะเป็นของคำทั้งหมดในอีเมลแฮม w คือคำแต่ละคำที่อยู่ในรายการ words ทำการวนลูปทีละคำเพื่อคำนวณหา pa_spam และ pa_ham ด้วยการหาค่า pw_spam ความน่าจะเป็นที่จะมีคำ w ในอีเมลสแปม และ pw_ham ความน่าจะเป็นที่จะมีคำ w ในอีเมลแฮม จากนั้นนำ pw_spam และ pw_ham ไปคูณสะสมในตัวแปร pa_spam และ pa_ham ตามลำดับจนครบทุกคำในอีเมล นำตัวแปร pa_spam ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลสแปม PS

และนำตัวแปร pa_ham ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลแฮม PH ถ้า PS มากกว่า PH จะจำแนกว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

2.3.2 การทำลาปลาซสมูททิง (Laplace smoothing) เพื่อแก้ปัญหาความแม่นยำที่ลดลงของอัลกอริทึม NB เนื่องจากมีค่าความน่าจะเป็นของคำบางคำเป็นศูนย์

ดังที่กล่าวไว้ อัลกอริทึม NB ใช้ทฤษฎีเบย์ (Bayes theorem) ซึ่งเป็นความน่าจะเป็น ดังนั้นถ้าต้องจำแนกอีเมลที่มีคำที่ไม่ปรากฏในชุดข้อมูลสำหรับฝึกฝน ความน่าจะเป็นจะมีค่าเป็นศูนย์ส่งผลให้ความแม่นยำในการจำแนกลดลง การทำลาปลาซสมูททิงจะช่วยป้องกันปัญหานี้ ดังนั้นการหาค่า P(x|C) จากสมการ 2.12 เมื่อเพิ่มการทำลาปลาซสมูททิงจะได้สมการที่ 2.13

$$P(x|C) = \frac{w_{x,C}+\alpha}{w_C+\alpha*A} \tag{2.13}$$

โดยที่ P(x|C) คือความน่าจะเป็นของคำ x ที่เป็นคลาส C,

$w_{x,C}$ คือจำนวนคำ x ทั้งหมดในคลาส C,

w_C คือจำนวนคำทั้งหมดในคลาส C,

A คือจำนวนของคำที่ไม่ซ้ำกัน,

α คือพารามิเตอร์ของการทำให้เรียบ มีค่าเท่ากับ 1

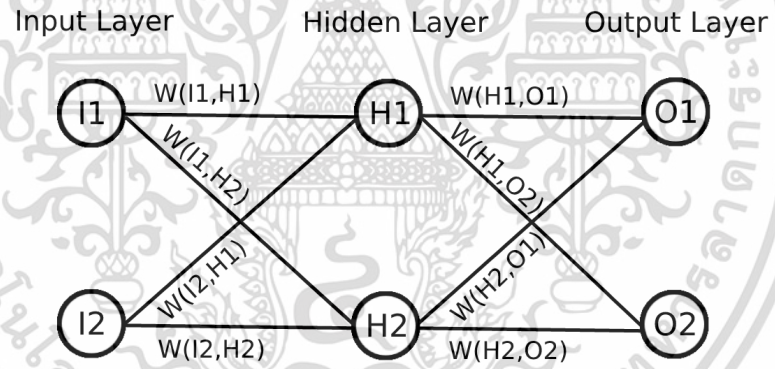
บทที่ 3
งานวิจัยที่เกี่ยวข้อง

3.1 การประยุกต์ใช้อัลกอริทึม BPNN เพื่อการจำแนกอีเมลสแปม

งานวิจัยนี้ [5] ประยุกต์ใช้อัลกอริทึม BPNN (Back Propagation Neural Network) เพื่อจำแนกประเภทอีเมลสแปม โดยก่อนที่จะนำชุดข้อมูลสำหรับฝึกฝนไปฝึกฝนด้วยอัลกอริทึม BPNN จะนำชุดข้อมูลสำหรับฝึกฝนไปจัดการผ่านกระบวนการก่อนการประมวลผล (Pre-procession) ด้วยการสกัดคุณลักษณะแบบไบนารี (Binary feature extraction) และนำคุณลักษณะที่สกัดได้ไปจัดกลุ่มด้วยอัลกอริทึมการแบ่งกลุ่มข้อมูลแบบเคมีน (K-mean clustering algorithm)

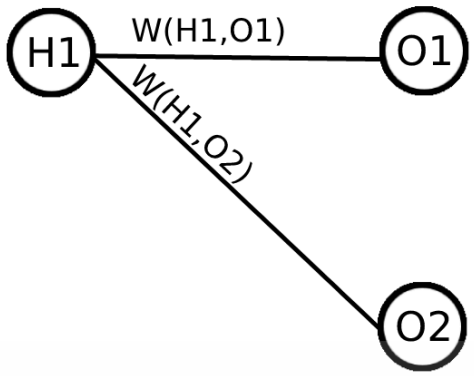
3.1.1 ขั้นตอนของอัลกอริทึม BPNN

โครงสร้างพื้นฐานของนิวรัลเน็ตเวิร์ก (Neural Network) ประกอบไปด้วยชั้นอินพุต (Input layer), ชั้นฮิดเดน (Hidden layer) และชั้นเอาต์พุต (Output layer) ดังรูป



รูปที่ 3.1 แสดงโครงสร้างพื้นฐานของนิวรัลเน็ตเวิร์ก

เพื่อหาค่าน้ำหนักที่เหมาะสม ตอนแรกจึงต้องสุ่มค่าเริ่มต้นของค่าน้ำหนักหรือ W ทุกตัวขึ้นมาก่อน และเมื่อทำงานไปข้างหน้าจนจบ 1 รอบ จะเปรียบเทียบผลการทำนายที่ได้กับผลการทำนายที่รู้อยู่แล้ว จากนั้นจะทำการแพร่แบบย้อนกลับ (Back Propagation) เพื่อปรับค่า W ทุกค่าให้เข้าใกล้ผลลัพธ์ที่ถูกต้องมากขึ้น จะทำซ้ำอย่างนี้หลายรอบจนกระทั่งได้ความแม่นยำในการจำแนกที่ต้องการ เพื่อให้เข้าใจมากขึ้น วิทยานิพนธ์นี้จะอธิบายตัวอย่างการทำการแพร่แบบย้อนกลับ (Back Propagation) ของโหนด H1



รูปที่ 3.2 แสดงบางส่วนของโครงสร้างพื้นฐานของนิวรัลเน็ตเวิร์ก

1) ระบุค่าเริ่มต้นของค่าน้ำหนักทุกค่าโดยการสุ่มตัวเลขที่มีค่าน้อยๆ เช่นระหว่าง -1 ถึง 1 ดังนั้น W(H1,O1) และ W(H1,O2) จะมีค่าเริ่มต้นที่กำหนดให้
2) ใช้ค่าอินพุตและค่าน้ำหนักเพื่อคำนวณผลลัพธ์ที่ได้
3) คำนวณค่าผิดพลาด (Error) ของเอาต์พุต O1 โดยใช้ค่าผลลัพธ์ที่ได้กับค่าผลลัพธ์ที่รู้อยู่แล้ว ด้วยสมการต่อไปนี้

$$Error_{O1} = Output_{O1} \times (1 - Output_{O1}) \times (Target_{O1} - Output_{O1}) \quad (3.1)$$

โดยที่ $Error_{O1}$ คือค่าผิดพลาดของเอาต์พุต O1,
$Output_{O1}$ คือค่าผลลัพธ์ O1 ที่ได้,
$Target_{O1}$ คือค่าผลลัพธ์ O1 ที่รู้อยู่แล้ว

4) ปรับค่าน้ำหนัก W(H1,O1) ด้วยสมการต่อไปนี้

$$W(H1,O1)' = W(H1,O1) + (Error_{O1} \times Output_{H1}) \quad (3.2)$$

โดยที่ W(H1,O1)' คือค่าน้ำหนักใหม่ที่ปรับได้ของ W(H1,O1),
W(H1,O1) คือค่าน้ำหนักเดิมของ W(H1,O1),
$Error_{O1}$ คือค่าผิดพลาดของเอาต์พุต O1,
$Output_{H1}$ คือค่าผลลัพธ์ H1 ที่ได้

5) ย้อนกลับไปขั้นตอนที่ 3 – 4 เพื่อคำนวณค่าผิดพลาดของเอาต์พุต O2 หรือ $Error_{O2}$ และปรับค่าน้ำหนัก W(H1,O2) ใหม่เช่นเดียวกับเอาต์พุต O1
6) ต่อมาย้อนกลับขึ้นมาคำนวณค่าผิดพลาดของชั้นฮิดเดน (H1) การคำนวณค่าผิดพลาดในชั้นฮิดเดนไม่สามารถคำนวณจากผลลัพธ์ที่รู้อยู่แล้วได้โดยตรง

เหมือนชั้นเอาต์พุต จึงต้องใช้ค่าผิดพลาดจากชั้นเอาต์พุต O1 และ O2 มาคำนวณย้อนกลับผ่านค่าน้ำหนัก

$$Error_{H1} = Output_{H1}(1 - Output_{H1})(Error_{O1}W(H1, O1) - Error_{O2}W(H1, O2)) \quad (3.3)$$

โดยที่ $Error_{H1}$ คือค่าผิดพลาดของ H1,
$Output_{H1}$ คือค่าผลลัพธ์ H1 ที่ได้,
$Error_{O1}$ คือค่าผิดพลาดของเอาต์พุต O1,
W(H1,O1) คือค่าน้ำหนักเดิมของ W(H1,O1),
$Error_{O2}$ คือค่าผิดพลาดของเอาต์พุต O2,
W(H1,O2) คือค่าน้ำหนักเดิมของ W(H1,O2)

7) กลับไปที่ขั้นตอนที่ 4 และทำการปรับค่าน้ำหนักย้อนกลับขึ้นไปทุกค่า โหนดอื่นๆก็สามารถปรับด้วยสูตรและวิธีการเดียวกัน ปรับจนกว่าจะได้ค่าผลลัพธ์ที่ได้ใกล้เคียงกับค่าผลลัพธ์ที่รู้อยู่แล้วในระดับที่พอใจ ซึ่งค่าน้ำหนักสุดท้ายที่ได้จะถูกนำไปใช้ในการจำแนกประเภทต่อไป

3.1.2 ประสิทธิภาพและข้อจำกัด

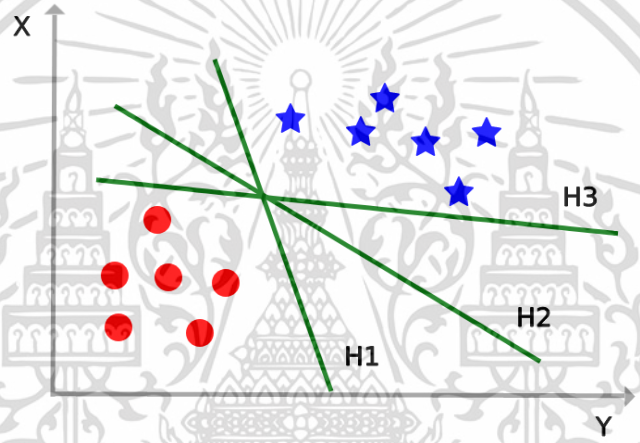
การประยุกต์ใช้อัลกอริทึม BPNN (Back Propagation Neural Network) ของงานวิจัยนี้ [5] เพื่อจำแนกประเภทอีเมลสแปม ให้ผลของค่าแม่นยำในการจำแนกที่สูง อย่างไรก็ตามก่อนที่จะทำการฝึกฝนต้องนำชุดข้อมูลสำหรับฝึกฝนเข้าสู่กระบวนการก่อนการประมวลผล นั้นคือการการสกัดคุณลักษณะแบบไบนารี และนำคุณลักษณะที่สกัดได้ไปจัดกลุ่มด้วยอัลกอริทึมการแบ่งกลุ่มข้อมูลแบบเคมีน และมีขั้นตอนการฝึกฝืนที่ซับซ้อนกว่าอัลกอริทึม NB แบบดั้งเดิม ซึ่งยากต่อการพัฒนาภายในตัวอัลกอริทึม BPNN อีกทั้งยังใช้เวลาในการฝึกฝนมากกว่าอัลกอริทึม NB แบบดั้งเดิม

3.2 การปรับปรุงอัลกอริทึม SVM แบบถ่วงน้ำหนักโดยใช้น้ำหนักจากอัลกอริทึม KFCM เพื่อการจำแนกอีเมลสแปม

งานวิจัยนี้ [6] ทำการปรับปรุงอัลกอริทึม SVM แบบถ่วงน้ำหนัก (Support Vector Machine) โดยจะใช้น้ำหนักจากอัลกอริทึม KFCM เพื่อการจำแนกอีเมลสแปม โดยมีแนวคิดที่ว่า SVM แบบถ่วงน้ำหนักทั่วไป จะพอใจกับอัตราการจำแนกประเภทที่ไม่ถูกต้องพร้อมกับการเปลี่ยนแปลงของความแม่นยำในการจำแนก งานวิจัยนี้จะกำหนดค่าเริ่มต้นของน้ำหนักเป็นหลัก โดยหาค่าน้ำหนักจากอัลกอริทึม KFCM ซึ่งให้ผลลัพธ์ที่ดีกว่า KPCM

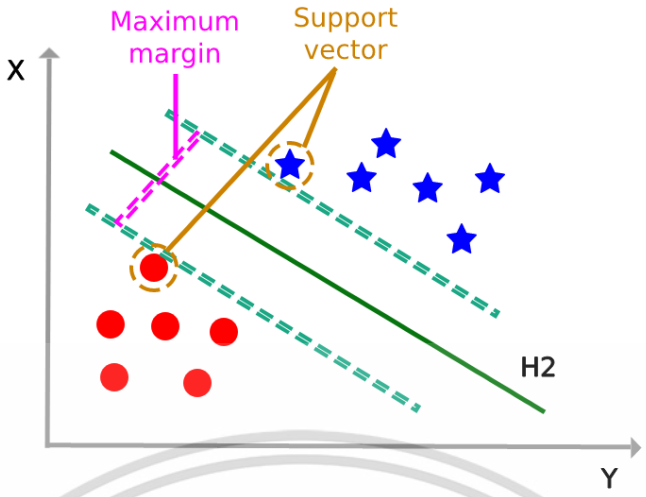
3.2.1 ขั้นตอนของอัลกอริทึมที่ปรับปรุงของงานวิจัยนี้

โดยทั่วไปอัลกอริทึม SVM จะใช้สำหรับการจำแนกประเภทแบบไบนารี (Binary classification) หรือปัญหาการจำแนกประเภทที่มี 2 คลาส อย่างไรก็ตามอัลกอริทึม SVM ยังสามารถดัดแปลงให้ใช้ได้กับการจำแนกประเภทข้อมูลที่มีมากกว่า 2 คลาส อัลกอริทึม SVM เหมาะสมสำหรับปัญหาการจำแนกที่มีข้อมูลไม่ใหญ่มาก แต่มีจำนวนคุณลักษณะ (Feature) มาก หลักการสำคัญของอัลกอริทึม SVM คือจะใช้การสร้างเส้นแบ่ง หรือไฮเปอร์เพลน (Hyperplane) เพื่อจำแนกข้อมูลออกจากกัน โดยไฮเปอร์เพลนมีเป็นจำนวนมาก อัลกอริทึม SVM จะทำการเลือกไฮเปอร์เพลนที่ดีที่สุด (Optimal Hyperplane) โดยมีหลักการเลือกดังนี้



รูปที่ 3.3 แสดงข้อมูลตัวอย่างบนกราฟ 2 มิติที่สามารถถูกแบ่งด้วยไฮเปอร์เพลนที่หลากหลาย

จากรูปข้างบนแสดงข้อมูลตัวอย่างบนกราฟที่สามารถถูกแบ่งด้วยไฮเปอร์เพลนจำนวนมาก สมมติว่าข้อมูลตัวอย่างมี 2 คลาสคือวงกลมและดาว มีแค่ 2 คุณลักษณะ (Feature) คือ X และ Y โดย H1, H2 และ H3 คือไฮเปอร์เพลนที่เป็นไปได้ทั้งหมด ทั้งไฮเปอร์เพลน H1, H2 และ H3 คือไฮเปอร์เพลนที่สามารถแบ่งข้อมูลออกจากกันได้ทั้งหมด ซึ่งอัลกอริทึม SVM จะทำการเลือกไฮเปอร์เพลนที่ดีที่สุด (Optimal Hyperplane) โดยพิจารณาว่าไฮเปอร์เพลนเส้นใดที่มีผลรวมของระยะห่างระหว่างเส้นไฮเปอร์เพลนกับเส้นตรงที่ลากผ่านข้อมูลของแต่ละคลาสที่ใกล้ที่สุดและเส้นตรงนั้นขนานกับเส้นไฮเปอร์เพลนมากที่สุด ซึ่งระยะห่างที่มากที่สุดของแต่ละไฮเปอร์เพลนนี้จะถูกเรียกว่าแม็กซิมัมมาร์จิน (Maximum Margin)



รูปที่ 3.4 แสดงไฮเปอร์เพลนที่ดีที่สุดของข้อมูลตัวอย่างบนกราฟ 2 มิติ

จากรูปข้างบนอัลกอริทึม SVM จะเลือกไฮเปอร์เพลน H2 เป็นไฮเปอร์เพลนที่ดีที่สุดของข้อมูลตัวอย่างนี้ เนื่องจากไฮเปอร์เพลน H2 มีแม็กซิมัมมาร์จิน (Maximum Margin) มากที่สุด ก็คือไฮเปอร์เพลน H2 มีระยะห่างของเส้นไฮเปอร์เพลนไปจนถึงเส้นตรงที่ลากผ่านข้อมูลที่ใกล้ที่สุดมากกว่าไฮเปอร์เพลน H1 และ H3 จึงถูกเลือกเป็นไฮเปอร์เพลนที่ดีที่สุด และจะเรียกข้อมูลที่ใกล้ที่สุดที่อยู่บนเส้นลากผ่านที่ขนานกับเส้นไฮเปอร์เพลนมากที่สุดว่าซัพพอร์ตเวกเตอร์ (Support Vector) จากตัวอย่างที่อธิบายข้างต้นจะเป็นการยกตัวอย่างในกรณีข้อมูลเป็นข้อมูลเชิงเส้น อย่างไรก็ตาม อัลกอริทึม SVM สามารถใช้กับข้อมูลที่ไม่เป็นข้อมูลเชิงเส้นได้โดยใช้การปรับค่าสมการ ซึ่งเรียกว่าวิธีการเคอร์เนล (Kernel) การจำแนกประเภทข้อมูลด้วยอัลกอริทึม SVM ให้ได้ประสิทธิภาพสูงนั้นจะต้องเลือกใช้เคอร์เนลฟังก์ชัน (Kernel Function) ให้เหมาะสมกับปัญหาหรือลักษณะของข้อมูล รวมถึงการปรับพารามิเตอร์ของอัลกอริทึม SVM โดยทั่วไปเคอร์เนลฟังก์ชันที่รู้จักกันดีและถูกใช้อย่างแพร่หลายเช่น Polynomial, Sigmoid และ Radial เป็นต้น ซึ่งงานวิจัยนี้จะใช้เคอร์เนลฟังก์ชันเป็นเรเดียน (Radial) ซึ่งเหมาะสมกับปัณหาและลักษณะข้อมูลในงานวิจัยมากที่สุด

อัลกอริทึม WSVM (Weighted Support Vector Machine) หรืออัลกอริทึม SVM แบบถ่วงน้ำหนัก เป็นการปรับปรุงอัลกอริทึม SVM ให้ได้ผลลัพธ์ความแม่นยำที่มากขึ้นโดยเพิ่มตัวแปรที่มีไว้ถ่วงน้ำหนัก โดยถ้ามีอัตราการจำแนกประเภทผิดสูงค่าของตัวแปรถ่วงน้ำหนักจะลดลง ซึ่งอัลกอริทึม WSVM จะใช้ค่าน้ำหนักจากอัลกอริทึม KPCM งานวิจัยนี้จะปรับปรุงอัลกอริทึม SVM แบบถ่วงน้ำหนัก (WSVM) โดยจะใช้ค่าน้ำหนักจากอัลกอริทึม KFCM แทนค่าน้ำหนักจากอัลกอริทึม KPCM ซึ่งค่าน้ำหนักจากอัลกอริทึม KFCM ให้ผลลัพธ์ที่ดีกว่าค่าน้ำหนักจากอัลกอริทึม KPCM ขั้นตอนของอัลกอริทึมที่ปรับปรุงของงานวิจัยนี้แสดงดังต่อไปนี้

1) รับค่าเอกสารในเทอมของเมทริกซ์
2) ตั้งค่าเคอร์เนลฟังก์ชัน (Kernel Function) เป็นเรเดียน (Radial) และฝึกฝน DTM สำหรับการดำเนินการ SVM
3) คำนวณอัตราการทำนายของอัลกอริทึม SVM
4) ตั้งค่าเคอร์เนลฟังก์ชัน (Kernel Function) เป็นเรเดียน (Radial) และฝึกฝน DTM สำหรับการดำเนินการ WSVM
5) คำนวณอัตราการทำนายของอัลกอริทึม WSVM
6) ใช้ฟังก์ชันของอัลกอริทึม KFCM เพื่อรับน้ำหนักและนำน้ำหนักที่ได้เก็บค่าไว้ในตัวแปร S
7) ใช้ตัวแปรน้ำหนัก S ในอัลกอริทึม WSVM
8) สรุปผลที่ได้ และคำนวณอัตราการทำนาย

3.2.2 ประสิทธิภาพและข้อจำกัด

การปรับปรุงอัลกอริทึม SVM แบบถ่วงน้ำหนัก (Support Vector Machine) โดยจะใช้น้ำหนักจากอัลกอริทึม KFCM เพื่อการจำแนกอีเมลสแปม [6] สามารถลดอัตราการจำแนกประเภทที่ผิดพลาดได้มากกว่า SVM แบบธรรมดา และ SVM แบบถ่วงน้ำหนัก อย่างไรก็ตามอัลกอริทึมนี้มีความซับซ้อนและใช้เวลามากกว่าอัลกอริทึม NB แบบดั้งเดิม และมีความยากในการปรับปรุงแก้ไข

3.3 การปรับปรุงอัลกอริทึม NB โดยการใช้อัลกอริทึม PSO เพื่อการจำแนกอีเมลสแปม

งานวิจัยนี้ [7] ทำการปรับปรุงอัลกอริทึม NB โดยการใช้อัลกอริทึม PSO (Particle Swarm Optimization) เพื่อการจำแนกประเภทอีเมลสแปม อัลกอริทึม NB คืออัลกอริทึมที่ใช้ในการจำแนกประเภทอีเมลโดยมีพื้นฐานจากความน่าจะเป็นในทฤษฎีเบย์ ส่วนอัลกอริทึม PSO เป็นอัลกอริทึมที่ทำงานในลักษณะวนซ้ำ เพื่อหาคำตอบที่ดีที่สุด

3.3.1 ขั้นตอนของอัลกอริทึมที่ปรับปรุงของงานวิจัยนี้

ในส่วนนี้จะขออธิบายแค่อัลกอริทึม PSO เนื่องจากรายละเอียดและขั้นตอนการทำงานของอัลกอริทึม NB ได้ถูกอธิบายไว้ในบทที่ 2 ของวิทยานิพนธ์นี้แล้ว อัลกอริทึม PSO หรือ Particle Swarm Optimization เป็นอัลกอริทึมที่ใช้หาคำตอบที่ดีที่สุดของปัญหา นิยมใช้ในการหาจุดต่ำสุด (Minimum Point) ในกราฟต่างๆ ที่จะส่งผลให้สมการกราฟมีค่าต่ำที่สุด โดยมีแรงบันดาลใจมาจากการหาอาหารของฝูงนก โดยนกแต่ละตัวในฝูงจะถูกแทนด้วยอนุภาค (Particle) และอนุภาคแต่ละตัวจะมีค่าตำแหน่ง (Position) และความเร็ว (Velocity) ซึ่งค่าตำแหน่งของอนุภาคจะถูกนำมาคำนวณค่าความเหมาะสม (Fitness Value) ที่บอกถึงระยะห่างของอนุภาคกับแหล่งอาหารหรือ

ระยะห่างของอนุภาคกับคำตอบที่ดีที่สุด อนุภาคแต่ละตัวจะมีค่า Pbest คือค่าตำแหน่งที่ดีที่สุดของอนุภาคตัวนั้นที่อยู่ใกล้คำตอบมากที่สุด ส่วนค่า Gbest คือค่าตำแหน่งที่ดีที่สุดของอนุภาคทั้งหมด ในแต่ละรอบการทำงานของอัลกอริทึม PSO อนุภาคแต่ละตัวจะปรับเปลี่ยนค่าตำแหน่งโดยใช้ค่า Pbest และ Gbest เป็นปัจจัยในการเปลี่ยนตำแหน่งเพื่อให้เข้าใกล้คำตอบที่ดีที่สุด ขั้นตอนของอัลกอริทึม PSO แสดงดังนี้

1) กำหนดจำนวนอนุภาคทั้งหมดเท่ากับ I สุ่มตำแหน่ง (Position) และความเร็ว (Velocity) เริ่มต้นในแต่ละอนุภาค

$$X_i = (x_{i1}, x_{i2}, \dots, x_{id}) \tag{3.4}$$

โดยที่ X_i คือค่าตำแหน่งของอนุภาคตัวที่ i,
d คือจำนวนมิติของปัญหา,
x_{id} คือค่าตำแหน่งของอนุภาคตัวที่ i ในมิติที่ d

$$V_i = (v_{i1}, v_{i2}, \dots, v_{id}) \tag{3.5}$$

โดยที่ V_i คือความเร็วของอนุภาคตัวที่ i,
d คือจำนวนมิติของปัญหา,
v_{id} คือความเร็วของอนุภาคตัวที่ i ในมิติที่ d

2) คำนวณค่าความเหมาะสม (Fitness Value) ของอนุภาคทุกตัว $f(X_i)$ โดยใช้สมการของปัญหาที่ต้องการหาจุดที่ต่ำที่สุด เลือกอนุภาคที่มีค่าความเหมาะสมต่ำที่สุดมาเก็บไว้ในตัวแปร min(f)
3) ถ้าค่าความเหมาะสมของ Gbest หรือ f(Gbest) มากกว่าค่า min(f) จะแทนที่ค่า f(Gbest) ด้วย min(f)
4) อัปเดตค่า Pbest ของอนุภาคแต่ละตัว ถ้าค่าความเหมาะสมของ Pbest หรือค่า $f(X_{ibest})$ มากกว่าค่าความเหมาะสมของตำแหน่งปัจจุบันของอนุภาคตัวนั้น หรือ $f(X_i)$ ให้แทนที่ค่า Pbest ของอนุภาคตัวนั้นด้วยค่า X_i
5) อัปเดตความเร็วของแต่ละอนุภาค โดยคำนวณความเร็วใหม่ด้วยสมการที่ 3.6

$$V_i' = wV_i + c_1 r_1 (Pbest_i - X_i) + c_2 r_2 (Gbest - X_i) \tag{3.6}$$

โดยที่ V'_i คือความเร็วใหม่ของอนุภาคตัวที่ i,
w คือค่าน้ำหนักแรงเฉื่อย (Inertia Weight) มีค่าเริ่มต้นเป็น 1,

c_1 และ c_2 คือค่าคงที่โดยทั่วไปมีค่าเป็น 2,

r_1 และ r_2 คือตัวเลขที่ถูกสุ่มระหว่าง 0 - 1,

$Pbest_i$ คือค่าตำแหน่งที่ดีที่สุดของอนุภาค i,

X_i คือค่าตำแหน่งของอนุภาคตัวที่ i,

Gbest คือค่าตำแหน่งที่ดีที่สุดของอนุภาคทั้งหมด

6) ลดค่า w ด้วยสมการที่ 3.7

$$w = w \times r \tag{3.7}$$

โดยที่ r คือตัวคูณที่ทำให้ค่า w ลดลง ให้กำหนดค่าที่น้อยกว่า 0 เช่น 0.99

7) อัปเดตค่าตำแหน่งของแต่ละอนุภาคด้วยสมการที่ 3.8

$$X_i' = X_i + V_i' \tag{3.8}$$

โดยที่ X'_i คือค่าตำแหน่งใหม่ของอนุภาคตัวที่ i,

X_i คือค่าตำแหน่งของอนุภาคตัวที่ i,

V'_i คือความเร็วใหม่ของอนุภาคตัวที่ i

8) หากเป็นไปตามเงื่อนไขต่อไปนี้ เงื่อนไขใดเงื่อนไขหนึ่งจะจบการทำงานของอัลกอริทึม PSO มิฉะนั้นวนกลับไปทำขั้นตอนที่ 2

เงื่อนไขที่ 1: ค่า f(Gbest) น้อยกว่าหรือเท่ากับค่า tolerance ที่กำหนดไว้เช่น กำหนดค่า tolerance เท่ากับ 10^{-5} ถ้าค่า f(Gbest) ในรอบนั้นน้อยกว่าหรือเท่ากับ 10^{-5} จะจบการทำงานของอัลกอริทึม SVM

เงื่อนไขที่ 2: จำนวนรอบในการทำซ้ำเท่ากับจำนวนรอบสูงสุดที่กำหนดไว้เช่น กำหนดจำนวนรอบสูงสุดไว้ที่ 1000 รอบ เมื่อถึงรอบที่ 1000 จะจบการทำงานของอัลกอริทึม SVM

เงื่อนไขที่ 3: ถ้าค่า f(Gbest) ไม่ลดลงหลังจากรอบที่กำหนดไว้เช่น กำหนดรอบในการหยุดก่อนกำหนดเป็น 500 หลังจากรอบที่ 500 ถ้าค่า f(Gbest) ไม่ลดลงจะจบการทำงานของอัลกอริทึม SVM

งานวิจัยนี้จะใช้อัลกอริทึม PSO เพื่อเพิ่มประสิทธิภาพพารามิเตอร์ในอัลกอริทึม NB เพื่อปรับปรุงความแม่นยำของอัลกอริทึม NB ให้มากขึ้น โดยจะมองโทเค็นทั้งหมดเป็นอนุภาค ขั้นตอนของอัลกอริทึมที่ปรับปรุงของงานวิจัยนี้แสดงดังต่อไปนี้

1) พิจารณาอีเมลในรูปแบบข้อความ

2) นำอีเมลไปเข้าสู่กระบวนการก่อนการประมวลผล (Pre-procession)

3) ประยุกต์ใช้ขั้นตอนก่อนการประมวลผลคือการทำให้อยู่ในรูปโทเค็น (Tokenization) และตัดคำ
4) ใช้ CFS เพื่อทำการเลือกคุณลักษณะ (Feature selection)
5) ใช้อัลกอริทึม NB เพื่อแจกแจงความน่าจะเป็น
6) ใช้อัลกอริทึม PSO เพื่อเพิ่มประสิทธิภาพ
7) หากค่าความน่าจะเป็นของโทเค็นสแปมมีค่ามากกว่า อีเมลนั้นจะถูกพิจารณาว่าเป็นอีเมลสแปม มิฉะนั้นจะพิจารณาว่าเป็นอีเมลปกติหรืออีเมลแฮม

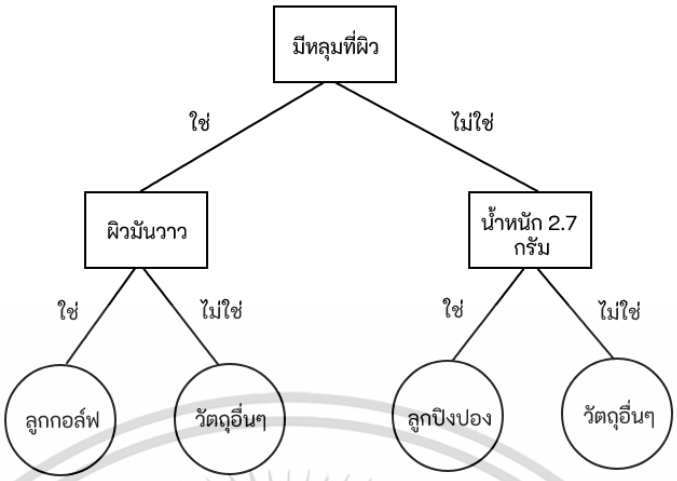
3.3.2 ประสิทธิภาพและข้อจำกัด

การปรับปรุงอัลกอริทึม NB แบบดั้งเดิมโดยการใช้อัลกอริทึม PSO หรือ Particle Swarm Optimization เพื่อการจำแนกประเภทอีเมลสแปม [7] ผลการทดลองแสดงให้เห็นว่าอัลกอริทึมนี้มีความแม่นยำมากกว่าอัลกอริทึม NB ทั่วไป อย่างไรก็ตามการรวม PSO กับ NB ใช้เวลาในการฝึกฝนและทดสอบมาก ซึ่งเป็นแบบจำลองที่มีความซับซ้อนสูงและยากในการแก้ไขและปรับปรุง

3.4 การจำแนกกลุ่มคำถามอัตโนมัติบนกระดานสนทนา โดยใช้เทคนิคเหมืองข้อความ

งานวิจัยนี้ [11] ทำการทดลองเปรียบเทียบประสิทธิภาพของอัลกอริทึม NB, การเรียนรู้ต้นไม้ตัดสินใจ (Decision Tree Learning) และอัลกอริทึม KNN (K-Nearest Neighbors) โดยการนำอัลกอริทึมทั้งสามมาทำการทดลองจำแนกกลุ่มคำถามอัตโนมัติบนกระดานสนทนา ในส่วนนี้จะอธิบายแค่การเรียนรู้ต้นไม้ตัดสินใจและอัลกอริทึม KNN เนื่องจากรายละเอียดและขั้นตอนการทำงานของอัลกอริทึม NB ได้ถูกอธิบายไว้ในบทที่ 2 ของวิทยานิพนธ์นี้แล้ว

การเรียนรู้ต้นไม้ตัดสินใจคือวิธีการเรียนรู้แบบมีผู้สอน (Supervised Learning) ที่ใช้ทั่วไปในการทำเหมืองข้อมูล (Data Mining), สถิติ (Statistics) และแมชชีนเลิร์นนิง (Machine Learning) การเรียนรู้ต้นไม้ตัดสินใจมีเป้าหมายเพื่อสร้างต้นไม้ตัดสินใจ (Decision Tree) เพื่อใช้ในการจำแนกประเภทข้อมูล ซึ่งอัลกอริทึมที่ใช้ในการสร้างต้นไม้ตัดสินใจยอดนิยมคืออัลกอริทึม ID3 (Iterative Dichotomiser 3) และอัลกอริทึม C4.5 งานวิจัยนี้ไม่ได้ระบุว่าใช้อัลกอริทึมใดในการสร้างต้นไม้ตัดสินใจ ดังนั้นวิทยานิพนธ์นี้จะอธิบายอัลกอริทึมที่นิยมใช้ในการสร้างต้นไม้ตัดสินใจ 2 อัลกอริทึมหลักได้แก่ อัลกอริทึม ID3 และอัลกอริทึม C4.5 เพื่อให้เห็นภาพชัดเจนรูปต่อไปนี้แสดงตัวอย่างต้นไม้ตัดสินใจที่ใช้ในการจำแนกประเภทลูกกอล์ฟ, ลูกปิงปอง และวัตถุอื่นๆ



รูปที่ 3.5 แสดงตัวอย่างต้นไม้ตัดสินใจที่ใช้จำแนกลูกกอล์ฟ, ลูกปิงปอง และวัตถุอื่นๆ

จากรูปด้านบนโหนดสีเหลี่ยมคือโหนดคุณลักษณะของข้อมูล (Attribute Node) ได้แก่ คุณลักษณะมีหลุมที่ผิว, คุณลักษณะผิวมันวาว, คุณลักษณะน้ำหนัก 2.7 กรัม โดยคุณลักษณะทั้งหมดมีค่าเหมือนกันคือใช่และไม่ใช่ โหนดวงกลมคือโหนดคำตอบ (Leaf Node) ที่เป็นตัวระบุผลการจำแนกประเภทข้อมูลได้แก่ ลูกกอล์ฟ, ลูกปิงปอง และวัตถุอื่นๆ

อัลกอริทึม ID3 และอัลกอริทึม C4.5 คืออัลกอริทึมที่ใช้ในการสร้างต้นไม้ต้นใจเพื่อใช้ในการจำแนกประเภทข้อมูล โดยอัลกอริทึม C4.5 จะพัฒนามาจากอัลกอริทึม ID3 อัลกอริทึมทั้งสองมีความแตกต่างกันดังนี้ อัลกอริทึม ID3 จะใช้การคำนวณค่าเกนความรู้ (Information Gain) ในการเลือกคุณลักษณะที่สำคัญที่สุดที่สามารถจำแนกได้มากที่สุดก่อน ในขณะที่อัลกอริทึม C4.5 จะใช้อัตราส่วนเกน (Gain Radio) เป็นเกณฑ์ในการเลือก อัตราส่วนเกนจะช่วยแก้ปัญหาความลำเอียงในการเลือกคุณลักษณะของค่าเกนความรู้ได้ โดยหลังจากการสร้างต้นไม้ตัดสินใจจนเสร็จสิ้น อัลกอริทึม C4.5 จะทำการ Pruning ต้นไม้ตัดสินใจที่ได้ด้วยวิธีการ Pessimistic Pruning ซึ่งจะทำให้ต้นไม้ตัดสินใจที่ได้มีขนาดเล็กลง เนื่องจากกฎการตัดสินใจ (Decision Rule) ที่ไม่สำคัญจะถูกตัดออก ทำให้มีความแม่นยำเพิ่มขึ้นจากอัลกอริทึม ID3 ขั้นตอนของอัลกอริทึม ID3 และอัลกอริทึม C4.5 แสดงดังนี้

1) อัลกอริทึม ID3 จะคำนวณค่าเกนความรู้ของคุณลักษณะทั้งหมดด้วยสมการที่ 3.9 ในขณะที่อัลกอริทึม C4.5 จะคำนวณอัตราส่วนเกนด้วยสมการที่ 3.10

$$IG(I,A) = Entropy(I) - Remainder(I,A) \tag{3.9}$$

โดยที่ IG(I,A) คือค่าเกนความรู้ของคุณลักษณะ A,

Entropy(I) คือค่าเอนโทรปีของกรณีตัวอย่างก่อนถูกแบ่งด้วยคุณลักษณะ A,
Remainder(I,A) คือผลรวมค่าเอนโทรปีหลังกรณีตัวอย่างถูกแบ่งด้วยคุณลักษณะ A

$$GR(I,A) = \frac{IG(I,A)}{SplitEntropy(I,A)} \quad (3.10)$$

โดยที่ GR(I,A) คืออัตราส่วนเกนของคุณลักษณะ A,
IG(I,A) คือค่าเกนความรู้ของคุณลักษณะ A,
SplitEntropy(I,A) คือค่าเอนโทรปีของค่าที่เป็นไปได้ทั้งหมดของคุณลักษณะ A

2) เลือกคุณลักษณะที่สำคัญที่สุด โดยอัลกอริทึม ID3 จะเลือกจากคุณลักษณะที่มีค่าเกนความรู้มากที่สุด ในขณะที่อัลกอริทึม C4.5 จะเลือกคุณลักษณะที่มีอัตราส่วนเกนมากที่สุดเพื่อนำไปสร้างโหนด
3) แบ่งกรณีตัวอย่างไปตามค่าที่เป็นไปได้ทั้งหมดของคุณลักษณะที่ถูกเลือก
4) ทำซ้ำขั้นตอนที่ 1 - 3 จนไม่เหลือกรณีตัวอย่างที่จำแนกไม่ได้อีก
5) เฉพาะอัลกอริทึม C4.5 จะทำขั้นตอนที่ 5 ต่อนั้นคือทำการ Pessimistic Pruning ต้นไม้ตัดสินใจที่ได้

อัลกกอริทึม KNN (K-Nearest Neighbors) คืออัลกอริทึมที่ใช้ในการจำแนกประเภท โดยใช้ฟังก์ชันระยะทาง (Distance Function) เพื่อเป็นตัวระบุความคล้ายกันของข้อมูลเพื่อจำแนกประเภท โดยฟังก์ชันระยะทางที่นิยมใช้ตัวหนึ่งคือฟังก์ชันระยะทางแบบยุคลิด (Euclidean Distance) ขั้นตอนของอัลกอริทึม KNN แสดงดังนี้

1) กำหนดค่า K คือจำนวนเพื่อนบ้านที่ใกล้เคียงที่ต้องการเลือก (Nearest Neighbors) เช่น กำหนด K เท่ากับ 5
2) คำนวณระยะทางแบบยุคลิดของกรณีตัวอย่างทั้งหมด
3) เลือกกรณีตัวอย่างที่มีค่าระยะทางยุคลิดต่ำที่สุดเป็นจำนวนเท่ากับค่า K ที่กำหนดไว้
4) จำแนกประเภทตามเสียงส่วนมากของกรณีตัวอย่างที่ถูกเลือกจากขั้นตอนที่ 3

การทดสอบในงานวิจัยนี้แต่ละอัลกอริทึมจะถูกทดสอบด้วยการทำ K-Fold Cross Validation ก็คือการแบ่งข้อมูลเป็นส่วนๆตามที่ต้องการ เช่นถ้าจะทำ 5-Fold Cross Validation ข้อมูลจะถูกแบ่งเป็น 5 ส่วน ข้อมูล 1 ส่วนจะใช้เป็นชุดข้อมูลสำหรับทดสอบ และส่วนที่เหลือจะเป็นชุดข้อมูลสำหรับฝึกฝนในแต่ละรอบ เพื่อให้เข้าใจมากขึ้นจะอธิบายการทำ 5-Fold Cross Validation สมมุติว่าข้อมูลทั้งหมดที่ถูกแบ่งออกเป็น 5 ส่วนคือ {1,2,3,4,5} จะทำการทดสอบการจำแนกประเภทข้อมูลของอัลกอริทึมใดๆเป็นจำนวน 5 รอบตามจำนวน Fold

รอบที่ 1: ชุดข้อมูลสำหรับฝึกฝนคือ {1,2,3,4} และชุดข้อมูลสำหรับทดสอบคือ {5}
รอบที่ 2: ชุดข้อมูลสำหรับฝึกฝนคือ {1,2,3,5} และชุดข้อมูลสำหรับทดสอบคือ {4}

รอบที่ 3: ชุดข้อมูลสำหรับฝึกฝนคือ {1,2,4,5} และชุดข้อมูลสำหรับทดสอบคือ {3}

รอบที่ 4: ชุดข้อมูลสำหรับฝึกฝนคือ {1,3,4,5} และชุดข้อมูลสำหรับทดสอบคือ {2}

รอบที่ 5: ชุดข้อมูลสำหรับฝึกฝนคือ {2,3,4,5} และชุดข้อมูลสำหรับทดสอบคือ {1}

เมื่อทดสอบครบทั้ง 5 รอบ จะนำความแม่นยำที่ได้ในแต่ละรอบมาหาค่าเฉลี่ย ได้เป็นค่าความแม่นยำเฉลี่ยของอัลกอริทึมนั้นๆที่ได้จากการทำ K-Fold Cross Validation และนำความแม่นยำเฉลี่ยที่ได้นี้มาเปรียบเทียบเพื่อวัดประสิทธิภาพในการจำแนกประเภทของแต่ละอัลกอริทึม

3.4.1 ขั้นตอนการทดสอบประสิทธิภาพอัลกอริทึม NB, การเรียนรู้ต้นไม้ตัดสินใจ และอัลกอริทึม KNN ของงานวิจัยนี้

1) นำข้อความจากกลุ่มคำถามไปเข้าสู่กระบวนการก่อนการประมวลผล (Pre-procession)
   - 1.1) การตัดคำ (Word Segmentation) ซึ่งงานวิจัยนี้ใช้โปรแกรมเล็กซ์โต (Thai Lexeme Tokenizer : LexTo) ในการตัดคำภาษาไทย โดยโปรแกรมนี้จะใช้หลักการเปรียบเทียบคำที่ยาวที่สุดที่มีในพจนุกรมภาษาไทยในโปรแกรมเพื่อตัดคำออกมา
   - 1.2) การกำจัดคำหยุด (Stop-Word Removal) คือการกำจัดคำที่ไม่มีความสำคัญออก คำหยุดในภาษาไทยได้แก่ คำบุพบท, คำสันธาน, คำสรรพนาม, คำวิเศษณ์ และคำอุทาน
   - 1.3) การหารากศัพท์ (Stem) ในภาษาไทยใช้การหารากศัพท์โดยวิธีรวมคำศัพท์ที่มีความหมายคล้ายกันให้เป็นรากศัพท์ และจัดเก็บลงคลังเพื่อนำไปเปรียบเทียบรากศัพท์
   - 1.4) การสร้างดัชนีคำสำคัญ (Indexing) งานวิจัยนี้ใช้การสร้างดัชนีคำสำคัญโดยการหาค่าน้ำหนักรูปแบบคำเดี่ยว ใช้วิธี TFIDF-Weighting (Term Frequency Inverse Document Frequency)
   - 1.5) การเลือกคุณสมบัติ (Feature selection) สร้างคุณสมบัติใหม่จากคุณสมบัติเดิม เลือกเฉพาะคุณสมบัติที่สำคัญๆ
2) สร้างแบบจำลองเพื่อใช้ทดสอบการจำแนกกลุ่มข้อความ งานวิจัยนี้จะทดสอบการจำแนกกลุ่มข้อความด้วยวิธี K-Fold Cross Validation ซึ่งกำหนดไว้ 3 K-Fold ดังนั้น 10-Fold, 15-Fold และ 20-Fold ทดลองบนโปรแกรม Weka (Waikato Environment for Knowledge Analysis)
3) เปรียบเทียบประสิทธิภาพการจำแนกประเภทของอัลกอริทึม NB, การเรียนรู้ต้นไม้ตัดสินใจ และอัลกอริทึม KNN

3.4.2 ผลการทดลองเพื่อเปรียบเทียบของอัลกอริทึม NB, การเรียนรู้ต้อนไม้ตัดสินใจ และอัลกอริทึม KNN

ผลการทดลองของงานวิจัยนี้แสดงให้เห็นว่าอัลกอริทึม KNN มีประสิทธิภาพในการจำแนกกลุ่มคำถามอัตโนมัติบนกระดารสนทนามากที่สุด โดยค่า k ที่ดีที่สุดของอัลกอริทึม KNN คือ 3 และการทำ K-Fold Cross Validation ที่ให้ผลลัพธ์ที่ดีที่สุดคือ 15-Fold Cross Validation

3.5 การเปรียบเทียบประสิทธิภาพของการจำแนกหมวดหมู่ของข้อคิดเห็นในแบบสอบถามปลายเปิด โดยอัลกอริทึม NB และอัลกอริทึม SVM

งานวิจัยนี้ [12] ทำการทดลองเปรียบเทียบประสิทธิภาพของอัลกอริทึม NB และอัลกอริทึม SVM (Support Vector Machine) โดยการทดลองจำแนกหมวดหมู่ของข้อคิดเห็นในแบบสอบถามปลายเปิด เนื่องจากอัลกอริทึม NB ได้ถูกอธิบายอย่างละเอียดในบทที่ 2 ของวิทยานิพนธ์นี้แล้ว และอัลกอริทึม SVM ได้ถูกอธิบายไว้แล้วเช่นกันในงานวิจัยเรื่องการปรับปรุงอัลกอริทึม SVM แบบถ่วงน้ำหนักโดยใช้น้ำหนักจากอัลกอริทึม KFCM เพื่อการจำแนกอีเมลสแปม ดังนั้นในส่วนนี้จะไม่ได้เขียนอธิบายเกี่ยวกับอัลกอริทึม NB และอัลกอริทึม SVM งานวิจัยนี้ใช้ชุดข้อมูลที่ได้จากแบบสอบถามเรื่องความคิดเห็นที่มีต่อสถาบันการศึกษาไทยระหว่างภาครัฐและเอกชนของประชาชนในเขตกรุงเทพฯ ในการทดลองชุดข้อมูลจะถูกแบ่งเป็น 2 ส่วนคือชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบด้วยสัดส่วน 80% : 20% คือชุดข้อมูลสำหรับฝึกฝน 80 ส่วนต่อชุดข้อมูลสำหรับทดสอบ 20 ส่วน และหมวดหมู่ของข้อคิดเห็นจะจำแนกเป็นเชิงบวก, กลาง และลบ

ชุดข้อมูลที่ใช้ทั้งหมด งานวิจัยนี้ได้เก็บรวบรวมข้อมูลเองจากการทำแบบสอบถาม 400 ฉบับ โดยสุ่มกลุ่มตัวอย่าง 400 คนจากประชาชนกรคนกรุงเทพฯทั้งหมด โดยแบบสอบถามทั้งหมดจะแบ่งได้เป็นชุดข้อมูลทั้งหมด 8 ชุดได้แก่ ชุดข้อมูลด้านราคาของสถาบันภาครัฐ, ชุดข้อมูลด้านราคาของสถาบันภาคเอกชน, ชุดข้อมูลด้านสถานที่ของสถาบันภาครัฐ, ชุดข้อมูลด้านสถานที่ของสถาบันภาคเอกชน, ชุดข้อมูลด้านความรู้และทักษะที่ได้รับของสถาบันภาครัฐ, ชุดข้อมูลด้านความรู้และทักษะที่ได้รับของสถาบันภาคเอกชน, ชุดข้อมูลด้านความยอมรับในสังคมของสถาบันภาครัฐ และชุดข้อมูลด้านความยอมรับในสังคมของสถาบันภาคเอกชน

การวัดประสิทธิภาพระหว่างอัลกอริทึม NB และอัลกอริทึม SVM จะใช้การวัดแบบ F-measure ซึ่งเป็นการวัดที่นิยมสำหรับอัลกอริทึมการจำแนกประเภท ค่า F-measure มีค่าอยู่ระหว่าง 0 - 1 เป็นค่าที่บอกได้ว่าอัลกอริทึมนั้นๆมีประสิทธิภาพการจำแนกประเภทได้ดีแค่ไหน การจะหาค่า F-measure จะต้องคำนวณหาค่า Precision และค่า Recall ก่อน สมการต่อไปนี้แสดงการหาค่า Precision, Recall และค่า F-measure

$$P = \frac{TP}{TP+FP} \tag{3.11}$$

โดยที่ P คือค่า Precision,
TP คือจำนวนกรณีตัวอย่างที่ทำนายถูกของคลาสที่สนใจ,
FP คือจำนวนกรณีตัวอย่างที่ทำนายผิดของคลาสที่สนใจ

$$R = \frac{TP}{TP+FN} \tag{3.12}$$

โดยที่ R คือค่า Recall,
TP คือจำนวนกรณีตัวอย่างที่ทำนายถูกของคลาสที่สนใจ,
FN คือจำนวนกรณีตัวอย่างที่ทำนายผิดของคลาสอื่น

$$F = \frac{2PR}{P+R} \tag{3.13}$$

โดยที่ F คือค่า F-measure มีค่าอยู่ระหว่าง 0 - 1,
P คือค่า Precision,
R คือค่า Recall

3.5.1 ขั้นตอนการทดสอบประสิทธิภาพระหว่างอัลกอริทึม NB และอัลกอริทึม SVM ของงานวิจัยนี้

1) การเลือกคุณสมบัติ (Feature selection) นำข้อความจากชุดข้อมูลสำหรับฝึกฝนมาทำการเลือกคำที่มีผลต่อจากจำแนกประเภท และตัดคำที่ไม่มีผลต่อการจำแนกประเภทข้อมูลออก โดยทำผ่านขั้นตอนดังนี้ การตัดคำ (Word Segmentation), การกำจัดคำหยุด (Stop-Word Removal), การหารากศัพท์ (Stem) และการสร้างดัชนีคำสำคัญ (Indexing)
2) เปรียบเทียบประสิทธิภาพการจำแนกประเภทของอัลกอริทึม NB และอัลกอริทึม SVM โดยพิจารณาจากค่า F-measure ของทั้งสองอัลกอริทึม

3.5.2 ผลการทดลองเพื่อเปรียบเทียบระหว่างอัลกอริทึม NB และอัลกอริทึม SVM

ผลการทดลองของงานวิจัยนี้แสดงให้เห็นว่าอัลกอริทึม SVM มีค่า F-measure อยู่ระหว่าง 0.903 - 0.963 ในชุดข้อมูลแบบสอบถามความคิดเห็นที่มีต่อสถาบันการศึกษาไทยทั้งภาครัฐและเอกชนทั้ง 4 ด้าน (ด้านราคา, ด้านสถานที่, ด้านความรู้และทักษะที่ได้รับ และด้านความยอมรับในสังคม) ในขณะที่อัลกอริทึม NB มีค่า F-measure อยู่ระหว่าง 0.719 - 0.835 เห็นได้ชัดว่าอัลกอริทึม SVM มีค่า F-measure ที่มากกว่าอัลกอริทึม NB ดังนั้นจึงสามารถสรุปได้ว่าอัลกอริทึม SVM มีประสิทธิภาพในการจำแนกหมวดหมู่ของข้อคิดเห็นที่มีต่อสถาบันการศึกษาไทยในเชิงบวก, กลางและลบมากกว่าอัลกอริทึม NB ในชุดข้อมูลความคิดเห็นทั้งหมด

บทที่ 4
งานวิจัยที่ต้องการปรับปรุง

4.1 อัลกอริทึม AWF-NB

อัลกอริทึม AWF-NB [8] คืองานวิจัยที่ทำการปรับปรุงอัลกอริทึม NB แบบดั้งเดิมด้วยการลดความสำคัญของคำในคลาสที่มีความถี่หรือความสำคัญต่ำกว่า โดยมุ่งเน้นที่จะแก้ปัญหาที่อัลกอริทึม NB จะมองคำทุกคำมีความสำคัญเท่ากันในแค่ละคลาส ซึ่งในความเป็นจริงไม่ได้เป็นเช่นนั้นทุกกรณี ซึ่งเป็นแนวคิดและปัญหาที่น่าสนใจสำหรับอัลกอริทึม NB แบบดั้งเดิม

4.1.1 ขั้นตอนของอัลกอริทึม AWF-NB ในการจำแนกประเภทอีเมลสแปม

1) กำหนดให้ X คือเซตของคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท และกำหนดให้ x เป็นสมาชิกหรือคำแต่ละคำในเซต X เขียนเป็นสัญลักษ์ดังนี้ X = { x | x คือคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท }
2) คำนวณความน่าจะเป็นของคำ x ในอีเมลสแปม P(x|S) และคำนวณความน่าจะเป็นของคำ x ในอีเมลแฮม P(x|H) โดยใช้สมการที่ 2.12
3) เปรียบเทียบระหว่าง P(x|S) และ P(x|H) ถ้าใครมีค่าน้อยกว่าจะถูกแทนค่าด้วยสมการที่ 4.1 ยกตัวอย่างเช่นถ้า P(x|S) มากกว่า P(x|H) จะแทนค่า P(x|H) ด้วยสมการที่ 4.1 และ P(x|S) ยังคงค่าเดิมไม่เปลี่ยนแปลง แสดงให้เห็นว่าคำ x ในคลาส S (อีเมลสแปม) มีความสำคัญมากกว่าคำ x ในคลาส H (อีเมลแฮม) ดังนั้น P(x|H) จึงควรถูกลดความสำคัญลง และนี้คือแนวคิดหลักของอัลกอริทึม AWF-NB แต่ถ้า P(x|H) มากกว่า P(x|S) จะแทนค่า P(x|S) ด้วยสมการที่ 4.1 และ P(x|H) ยังคงค่าเดิม

$$P(x|C) = \frac{1}{w_C} \tag{4.1}$$

โดยที่ P(x|C) คือความน่าจะเป็นของคำ x ที่เป็นคลาส C,
C คือคลาสที่มีความสำคัญน้อยกว่า,
w_C คือจำนวนคำทั้งหมดในคลาส C

4) ทำซ้ำในขั้นตอนที่ 2 - 3 เพื่อหาค่า P(x|S) และ P(x|H) ใหม่ในคำทั้งหมด ก็คือ วนตรวจสอบคำทั้งหมดเพื่อลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า

5) ใช้ P(x|S) และ P(x|H) ที่ได้ในแต่ละคำเพื่อคำนวณความน่าจะเป็นที่จะเป็นอีเมลสแปม P(S|X) กับความน่าจะเป็นที่จะเป็นอีเมลแฮม P(H|X) โดยใช้สมการที่ 2.10

6) ถ้า P(S|X) มากกว่า P(H|X) จำแนกอีเมลว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

เพื่อให้เข้าใจความแตกต่างระหว่างอัลกอริทึม AWF-NB และอัลกอริทึม NB แบบดั้งเดิม วิทยานิพนธ์นี้จะแสดงให้เห็นถึงขั้นตอนของอัลกอริทึม AWF-NB ที่ถูกเพิ่มจากอัลกอริทึม NB ดังนั้นจะทำการปรับเปลี่ยนจากรหัสเทียมของอัลกอริทึม NB ในรูปที่ 2.1 ให้เป็นรหัสเทียมของอัลกอริทึม AWF-NB เริ่มจากการเปลี่ยนบรรทัดที่ 1 ของรูปที่ 2.1 เป็น “function AWF-NB (email) returns classification” ต่อมาแทรกคำสั่งทั้งหมดในรูปที่ 4.1 ระหว่างบรรทัดที่ 7 และบรรทัดที่ 8 ของรูปที่ 2.1 จะได้เป็นรหัสเทียมแบบสมบูรณ์ของอัลกอริทึม AWF-NB แสดงในรูปที่ 4.2

```
if pw_ham is less than pw_spam and pw_ham isn't equal to 0
    then reducing the importance of pw_ham
else if pw_spam is less than pw_ham and pw_spam isn't equal to 0
    then reducing the importance of pw_spam
```

รูปที่ 4.1 แสดงบางส่วนของรหัสเทียมของอัลกอริทึม AWF-NB

```
1  function AWF-NB (email) returns classification
       words ← List of word in email
3      pa_spam ← 1
4      pa_ham ← 1
       for each value w of words do
           pw_spam ← Compute the probability of word w in spam email
7          pw_ham ← Compute the probability of word w in ham email
8          if pw_ham is less than pw_spam and pw_ham isn't equal to 0
               then reducing the importance of pw_ham
           else if pw_spam is less than pw_ham and pw_spam isn't equal to 0
11             then reducing the importance of pw_spam
12         pa_spam ← pa_spam * pw_spam
13         pa_ham ← pa_ham * pw_ham
       PS ← Compute the probability of spam using pa_spam
       PH ← Compute the probability of ham using pa_ham
       if PS is greater than PH,
           then return spam class
       else
           return ham class
```

รูปที่ 4.2 แสดงรหัสเทียมของอัลกอริทึม AWF-NB

จากรหัสเทียมของอัลกอริทึม AWF-NB ในรูปที่ 4.1 ฟังก์ชัน AWF-NB จะคืนค่าผลการจำแนกประเภทอีเมล โดยจะรับพารามิเตอร์เป็นอีเมลที่ต้องการจำแนกประเภท ตัวแปร words จะเก็บรายการของคำทั้งหมดในอีเมล ตัวแปร pa_spam คือผลคูณความน่าจะเป็นของคำทั้งหมดในอีเมลสแปม ส่วน pa_ham คือผลคูณความน่าจะเป็นของคำทั้งหมดในอีเมลแฮม w คือคำแต่ละคำที่อยู่ในรายการ words ทำการวนลูปทีละคำเพื่อคำนวณหา pa_spam และ pa_ham ซึ่งตอนแรกจะหาค่า pw_spam ความน่าจะเป็นที่จะมีคำ w ในอีเมลสแปม และ pw_ham ความน่าจะเป็นที่จะมีคำ w ในอีเมลแฮม แล้วตรวจสอบเงื่อนไขดังนี้

1) เงื่อนไขที่ 1: ถ้า pw_ham น้อยกว่า pw_spam และ pw_ham ไม่เท่ากับ 0 ให้ลดความสำคัญของ pw_ham
2) เงื่อนไขที่ 2: ถ้า pw_spam น้อยกว่า pw_ham และ pw_spam ไม่เท่ากับ 0 ให้ลดความสำคัญของ pw_spam

จากนั้นนำ pw_spam และ pw_ham ไปคูณสะสมในตัวแปร pa_spam และ pa_ham ตามลำดับจนครบทุกคำในอีเมล นำตัวแปร pa_spam ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลสแปม PS และนำตัวแปร pa_ham ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลแฮม PH ถ้า PS มากกว่า PH จะจำแนกว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

4.1.2 ประยุกต์ใช้การทำลาปลาซสมูททิงกับอัลกอริทึม AWF-NB

ตามที่ได้อธิบายไว้ว่าอัลกอริทึม NB แบบดั้งเดิมมีปัญหาความน่าจะเป็นมีค่าเป็นศูนย์ ในกรณีที่มีคำบางคำในอีเมลทดสอบหรืออีเมลที่ต้องการจำแนกประเภท ไม่มีอยู่ในชุดข้อมูลสำหรับฝึกฝน ดังนั้นค่าความจะเป็นจึงมีค่าเป็นศูนย์ ส่งผลให้ความแม่นยำลดลงเนื่องจากไม่สามารถจำแนกประเภทอีเมลได้ เมื่ออัลกอริทึม AWF-NB ปรับปรุงมาจากอัลกอริทึม NB ดังนั้นจึงควรใช้การทำลาปลาซสมูททิงเช่นเดียวกัน ในการทดลองจริงขั้นตอนของอัลกอริทึม AWF-NB ที่ต้องปรับเปลี่ยนเพื่อรวมการทำลาปลาซสมูททิงคือขั้นตอนที่ 2 และ 3

ขั้นตอนที่ 2 จากการหาค่า P(x|S) และ P(x|H) ให้เปลี่ยนไปใช้สมการที่ 2.13 ซึ่งเป็นสมการที่รวมการทำลาปลาซสมูททิงแล้ว

ขั้นตอนที่ 3 เมื่อต้องการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า ให้เปลี่ยนจากสมการที่ 4.1 มาใช้สมการที่ 4.2 ซึ่งเป็นสมการที่รวมการทำลาปลาซสมูททิง

$$P(x|C) = \frac{1+\alpha}{w_C+\alpha*A} \tag{4.2}$$

โดยที่ P(x|C) คือความน่าจะเป็นของคำ x ที่เป็นคลาส C,

w_C คือจำนวนคำทั้งหมดในคลาส C,

A คือจำนวนของคำที่ไม่ซ้ำกัน,

α คือพารามิเตอร์ของการทำให้เรียบ มีค่าเท่ากับ 1

4.1.3 ประสิทธิภาพและข้อจำกัด

จากแนวคิดหลักและขั้นตอนของอัลกอริทึม AWF-NB ใช้วิธีการปรับปรุงอัลกอริทึม NB แบบดั้งเดิมอย่างง่ายและไม่ซับซ้อนด้วยวิธีลดความสำคัญของคำที่อยู่ในคลาสที่สำคัญน้อยกว่า เมื่อพิจารณาเวลาในการดำเนินการ อัลกอริทึม AWF-NB ยังคงใช้เวลาในการดำเนินการอย่างรวดเร็วเหมือนอัลกอริทึม NB แบบดั้งเดิม อย่างไรก็ตามการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าลงอย่างมากนั้น จะส่งผลให้ความแม่นยำตกลงในกรณีที่ความสำคัญของคำในแต่ละคลาสแตกต่างกันเพียงเล็กน้อยเท่านั้น และนี้ก็คือปัญหาที่สำคัญของอัลกอริทึม AWF-NB

บทที่ 5
งานวิจัยที่เสนอ

ตามที่ได้อธิบายปิดท้ายในบทที่แล้ว อัลกอริทึม AWF-NB [8] มีปัญหาที่สำคัญซึ่งจะอธิบายในบทนี้ ตลอดจนวิธีการแก้ปัญหาของงานวิจัยที่เสนอหรืออัลกอริทึม RIWE-NB ปัญหาที่สำคัญของอัลกอริทึม AWF-NB นั้นคือ ปัญหาความแม่นยำที่ลดลงเนื่องจากการลดความสำคัญของคำในคลาสที่มีความสำคัญแตกต่างกันเพียงเล็กน้อย

5.1 ปัญหาความแม่นยำที่ลดลงเนื่องจากการลดความสำคัญของคำในคลาสที่มีความสำคัญแตกต่างกันเพียงเล็กน้อยของอัลกอริทึม AWF-NB

ตามที่ได้อธิบายแนวคิดหลักของอัลกอริทึม AWF-NB ในบทที่แล้ว อัลกอริทึม AWF-NB จะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าลงอย่างมาก ตัวอย่างเช่น สมมติว่าจำนวนคำทั้งหมดในคลาส S เท่ากับ 47 คำและ H เท่ากับ 60 คำ จำนวนคำ x ในอีเมลสแปมเท่ากับ 20 คำ จำนวนคำ x ในอีเมลแฮมเท่ากับ 19 คำ ดังนั้น P(x|S) เท่ากับ 0.43, P(x|H) เท่ากับ 0.32 ในกรณีนี้ อัลกอริทึม AWF-NB จะลด P(x|H) ซึ่งมีค่าน้อยกว่า P(x|S) ให้เหลือ 0.02 โดยแทนค่า P(x|H) ด้วยสมการที่ 4.1 จะเห็นว่าการกระทำนี้ได้ลดความสำคัญของคำ x ในคลาส H หรือ P(x|H) ลงอย่างมาก ในขณะที่ P(x|H) และ P(x|S) มีค่าต่างกันเพียงเล็กน้อยเท่านั้น นี้คือสาเหตุของปัญหาความแม่นยำที่ลดลงในอัลกอริทึม AWF-NB เพื่อให้เห็นภาพมากขึ้น วิทยานิพนธ์นี้จะแสดงขั้นตอนการจำแนกประเภทอีเมลของอัลกอริทึม AWF-NB โดยไม่ใช้การทำลาปลาซสมูททิงในการลดความความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าจากกรณีตัวอย่างดังนี้

ตารางที่ 5.1 ลักษณะชุดข้อมูลสำหรับฝึกฝนของกรณีตัวอย่างที่แสดงสาเหตุของปัญหาความแม่นยำที่ลดลงของอัลกอริทึม AWF-NB

คำ	จำนวนคำในคลาส S	จำนวนคำในคลาส H	รวม
x	20	19	39
y	15	31	46
z	12	10	22
รวม	47	60	107

จากตารางที่ 6.1 สมมุติว่าอีเมลทั้งหมดในชุดข้อมูลสำหรับฝึกฝนมีคำแค่ 3 คำคือคำ x, คำ y และคำ z จำนวนคำทั้งหมดในชุดข้อมูลสำหรับฝึกฝนเท่ากับ 107 แบ่งเป็นจำนวนคำในอีเมลสแปม (คลาส S) 47 คำ และจำนวนคำในอีเมลแฮม (คลาส H) 60 คำ จำนวนคำ x ที่เป็นอีเมลสแปม 20 คำ,

จำนวนคำ x ที่เป็นอีเมลแฮม 19 คำ, จำนวนคำ y ที่เป็นอีเมลสแปม 15 คำ, จำนวนคำ y ที่เป็นอีเมลแฮม 31 คำ, จำนวนคำ z ที่เป็นอีเมลสแปม 12 คำ, จำนวนคำ z ที่เป็นอีเมลแฮม 10 คำ และสมมุติว่าอีเมลที่ใช้ทดสอบเป็นอีเมลแฮม และอีเมลนี้ประกอบไปด้วยคำ x, คำ y และคำ z ขั้นตอนการลดความสำคัญของอัลกอริทึม AWF-NB เพื่อจำแนกประเภทอีเมลแสดงดังนี้

1) คำนวณ P(x|S) (ความน่าจะเป็นของคำ x ที่มีในคลาส S) และ P(x|H) (ความน่าจะเป็นของคำ x ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(x|S) = \frac{20}{47} = 0.43 \tag{5.1}$$

$$P(x|H) = \frac{19}{60} = 0.32 \tag{5.2}$$

2) เปรียบเทียบระหว่าง P(x|S) และ P(x|H) พบว่า P(x|H) 0.32 น้อยกว่า P(x|S) 0.43 ดังนั้น P(x|H) จะถูกแทนค่าด้วยสมการที่ 4.1 ผลลัพธ์ที่ได้แสดงในสมการที่ 5.3 ส่วน P(x|S) ยังจะคงเท่าเดิม

$$P(x|H) = \frac{1}{60} = 0.02 \tag{5.3}$$

3) คำนวณ P(y|S) (ความน่าจะเป็นของคำ y ที่มีในคลาส S) และ P(y|H) (ความน่าจะเป็นของคำ y ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(y|S) = \frac{15}{47} = 0.32 \tag{5.4}$$

$$P(y|H) = \frac{31}{60} = 0.52 \tag{5.5}$$

4) เปรียบเทียบระหว่าง P(y|S) และ P(y|H) พบว่า P(y|S) 0.32 น้อยกว่า P(y|H) 0.52 ดังนั้น P(y|S) จะถูกแทนค่าด้วยสมการที่ 4.1 ผลลัพธ์ที่ได้แสดงในสมการที่ 5.6 ส่วน P(y|H) ยังจะคงเท่าเดิม

$$P(y|S) = \frac{1}{47} = 0.02 \tag{5.6}$$

5) คำนวณ P(z|S) (ความน่าจะเป็นของคำ z ที่มีในคลาส S) และ P(z|H) (ความน่าจะเป็นของคำ z ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(z|S) = \frac{12}{47} = 0.26 \quad (5.7)$$

$$P(z|H) = \frac{10}{60} = 0.17 \quad (5.8)$$

6) เปรียบเทียบระหว่าง P(z|S) และ P(z|H) พบว่า P(z|H) 0.17 น้อยกว่า P(z|S) 0.26 ดังนั้น P(z|H) จะถูกแทนค่าด้วยสมการที่ 4.1 ผลลัพธ์ที่ได้แสดงในสมการที่ 5.9 ส่วน P(z|S) ยังจะคงเท่าเดิม

$$P(z|H) = \frac{1}{60} = 0.02 \quad (5.9)$$

7) จะได้ค่าทั้งหมดดังนี้ P(x|S) = 0.43, P(x|H)* = 0.02, P(y|S)* = 0.02, P(y|H) = 0.52, P(z|S) = 0.26 และ P(z|H)* = 0.02 ค่าที่ทำเครื่องหมาย * คือคลาสที่มีความสำคัญต่ำกว่าและถูกลดความสำคัญลงโดยการแทนค่าจากสมการที่ 4.1

8) นำค่าที่ได้ทั้งหมดไปใช้ในการคำนวณเพื่อหาความน่าจะเป็นที่อีเมลจะเป็นอีเมลสแปม P(S|X) และความน่าจะเป็นที่อีเมลจะเป็นอีเมลแฮม P(H|X) ด้วยสมการที่ 2.10

$$P(S|X) = \frac{47}{107} \times (0.43 \times 0.02 \times 0.26) = 0.0010 \quad (5.10)$$

$$P(H|X) = \frac{60}{107} \times (0.02 \times 0.52 \times 0.02) = 0.0001 \quad (5.11)$$

9) เปรียบเทียบความน่าจะเป็นที่จะเป็นอีเมลสแปม P(S|X) กับความน่าจะเป็นที่จะเป็นอีเมลแฮม P(H|X) พบว่า P(S|X) มากกว่า P(H|X) ดังนั้นจึงจำแนกอีเมลทดสอบเป็นอีเมลสแปม

จากการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าเพียงเล็กน้อยของอัลกอริทึม AWF-NB ในเคสตัวอย่าง จะเห็นว่าการกระทำนี้ทำให้จำแนกประเภทอีเมลทดสอบผิด แต่เดิมอีเมลทดสอบเป็นอีเมลแฮม แต่พอผ่านการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าเพียงเล็กน้อยของอัลกอริทึม AWF-NB ส่งผลให้จำแนกอีเมลทดสอบเป็นอีเมลสแปม และนี้คือเหตุผลที่ความแม่นยำของอัลกอริทึม AWF-NB นั้นลดลง ถ้าหากยิ่งมีอีเมลที่มีความสำคัญของคำในแต่ละคลาสไม่ต่างกันมากหลายๆอีเมล ความแม่นยำของอัลกอริทึม AWF-NB ก็จะยิ่งลดลง เพื่อพิสูจน์ว่าการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าเพียงเล็กน้อยของอัลกอริทึม AWF-NB นั้นทำให้จำแนกประเภทอีเมลทดสอบในกรณีตัวอย่างผิด วิทยานิพนธ์นี้จะแสดงขั้นตอนจากการจำแนกประเภทอีเมลทดสอบในกรณีตัวอย่างด้วยอัลกอริทึม NB แบบดั้งเดิมโดยที่ไม่มีการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าดังนี้

1) คำนวณ P(x|S) (ความน่าจะเป็นของคำ x ที่มีในคลาส S) และ P(x|H) (ความน่าจะเป็นของคำ x ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(x|S) = \frac{20}{47} = 0.43 \quad (5.12)$$

$$P(x|H) = \frac{19}{60} = 0.32 \quad (5.13)$$

2) คำนวณ P(y|S) (ความน่าจะเป็นของคำ y ที่มีในคลาส S) และ P(y|H) (ความน่าจะเป็นของคำ y ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(y|S) = \frac{15}{47} = 0.32 \quad (5.14)$$

$$P(y|H) = \frac{31}{60} = 0.52 \quad (5.15)$$

3) คำนวณ P(z|S) (ความน่าจะเป็นของคำ z ที่มีในคลาส S) และ P(z|H) (ความน่าจะเป็นของคำ z ที่มีในคลาส H) โดยใช้สมการที่ 2.12

$$P(z|S) = \frac{12}{47} = 0.26 \quad (5.16)$$

$$P(z|H) = \frac{10}{60} = 0.17 \quad (5.17)$$

4) จะได้ค่าทั้งหมดดังนี้ P(x|S) = 0.43, P(x|H) = 0.32, P(y|S) = 0.32, P(y|H) = 0.52, P(z|S) = 0.26 และ P(z|H) = 0.17

5) นำค่าที่ได้ทั้งหมดไปใช้ในการคำนวณเพื่อหาความน่าจะเป็นที่อีเมลจะเป็นอีเมลสแปม P(S|X) และความน่าจะเป็นที่อีเมลจะเป็นอีเมลแฮม P(H|X) ด้วยสมการที่ 2.10

$$P(S|X) = \frac{47}{107} \times (0.43 \times 0.32 \times 0.26) = 0.0157 \quad (5.18)$$

$$P(H|X) = \frac{60}{107} \times (0.32 \times 0.52 \times 0.17) = 0.0158 \quad (5.19)$$

6) เปรียบเทียบความน่าจะเป็นที่จะเป็นอีเมลสแปม P(S|X) กับความน่าจะเป็นที่จะเป็นอีเมลแฮม P(H|X) พบว่า P(H|X) มากกว่า P(S|X) ดังนั้นจึงจำแนกอีเมลทดสอบเป็นอีเมลแฮม

เมื่อจำแนกอีเมลทดสอบด้วยอัลกอริทึม NB แบบดั้งเดิมที่ไม่มีการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าเพียงเล็กน้อย ผลที่ได้คือจำแนกประเภทอีเมลเป็นอีเมลแฮมได้ถูกต้อง และนี้คือปัญหาสำคัญของอัลกอริทึม AWF-NB

5.2 แนวคิดการปรับปรุงข้อเสียในอัลกอริทึม AWF-NB ของงานวิจัยที่เสนอ (อัลกอริทึม RIWE-NB)

เนื่องจากปัญหาของอัลกอริทึม AWF-NB คือการลดความสำคัญอย่างมากของคำในคลาสที่มีความสำคัญต่ำกว่า โดยไม่ได้พิจารณาว่าความสำคัญของคำในแต่ละคลาสอาจจะต่างกันเพียงเล็กน้อยเท่านั้น จึงนำไปสู่การลดลงของความแม่นยำในการจำแนก วิธีการแก้ปัญหาที่นำเสนอคือจะใช้การคำนวณค่าเอนโทรปีของคำแต่ละคำ เพื่อเป็นเครื่องช่วยตัดสินใจว่าควรจะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าหรือไม่ ค่าเอนโทรปีคือค่าความไม่เป็นระเบียบของข้อมูลดังที่ได้อธิบายไว้ในบทที่ 2 ยิ่งคำมีการกระจายออกเป็นคลาสอีเมลสแปมและอีเมลแฮมมากเท่าไหร่ เอนโทรปีจะยิ่งมีค่ามาก ถ้าเอนโทรปีมีค่ามากแสดงว่าความสำคัญของคำในแต่ละคลาสมีค่าใกล้เคียงกันหรือแตกต่างกันเพียงเล็กน้อย ดังนั้นจึงไม่ควรที่จะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า ถ้าค่าเอนโทรปีของคำมีค่ามาก ยกตัวอย่างการคำนวณค่าเอนโทรปีของคำ x จากกรณีตัวอย่างในตารางที่ 5.1 โดยใช้สมการที่ 2.1 แทนค่าได้เป็นสมการที่ 5.20 และ 5.21

$$En(x) = -[(P(x|S)\log_2 p(x|S)) + (P(x|H)\log_2 P(x|H))] \tag{5.20}$$

โดยที่ En(x) คือค่าเอนโทรปีของคำ x,
P(x|S) คือความน่าจะเป็นของคำ x ที่เป็นคลาส S (อีเมลสแปม),
P(x|H) คือความน่าจะเป็นของคำ x ที่เป็นคลาส H (อีเมลแฮม)

$$En(x) = -[\left(\frac{20}{39}\log_2\frac{20}{39}\right) + \left(\frac{19}{39}\log_2\frac{19}{39}\right)] = 0.9997 \tag{5.21}$$

จากการคำนวณค่าเอนโทรปีของคำ x พบว่ามีค่าเท่ากับ 0.9997 ซึ่งมีค่ามากเกือบจะเท่ากับค่าเอนโทรปีสูงสุดคือ 1 ในกรณีของการจำแนกอีเมลสแปม ซึ่งมีแค่ 2 คลาสค่าเอนโทรปีจะมีค่าระหว่าง 0 - 1 และเนื่องจากคำ x มีการกระจายตัวออกเป็นคลาส S และคลาส H เกือบจะเท่ากันต่างกันนิดเดียวเท่านั้น ดังนั้นจึงไม่ควรลดความสำคัญของคำ x ในคลาสที่มีความสำคัญน้อยกว่าลง เพราะจะส่งผลให้ความแม่นยำลดลง นี้ก็คือแนวคิดหลักของอัลกอริทึม RIWE-NB ในการปรับปรุงข้อเสียของอัลกอริทึม AWF-NB

5.3 ขั้นตอนของอัลกอริทึม RIWE-NB ในการจำแนกประเภทอีเมลสแปม

1) กำหนดให้ X คือเซตของคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท และกำหนดให้ x เป็นสมาชิกหรือคำแต่ละคำในเซต X เขียนเป็นสัญลักษ์ดังนี้ X = { x | x คือคำทั้งหมดในอีเมลที่ต้องการจำแนกประเภท }
2) คำนวณความน่าจะเป็นของคำ x ในอีเมลสแปม P(x|S) และคำนวณความน่าจะเป็นของคำ x ในอีเมลแฮม P(x|H) โดยใช้สมการที่ 2.13 ซึ่งสมการนี้เป็นสมการที่ได้รวมเข้ากับการทำลาปลาซสมูททิงแล้ว เพื่อป้องกันปัญหาความน่าจะเป็นมีค่าเป็นศูนย์
3) ถ้า P(x|S) ไม่เท่ากับ P(x|H) ให้ทำต่อในขั้นตอนที่ 4 มิฉะนั้นให้ข้ามไปขั้นตอนที่ 8
4) คำนวณค่าเอนโทรปีของคำ x โดยใช้สมการที่ 5.20
5) สุ่มตัวเลข r ระหว่าง 0 - 1 (ค่าเอนโทรปีของการจำแนกอีเมลสแปมมีค่าอยู่ระหว่าง 0 - 1)
6) ถ้า r มากกว่าค่าเอนโทรปีของคำ x ในขั้นตอนที่ 4 ให้ไปขั้นตอนที่ 7 เพื่อลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า มิฉะนั้นข้ามไปขั้นตอนที่ 8
7) เปรียบเทียบระหว่าง P(x|S) และ P(x|H) เพื่อหาว่าใครน้อยกว่า สมมุติ P(x|S) น้อยกว่า P(x|H) และ P(x|H) ไม่เท่ากับ 0 ให้แทนค่า P(x|S) หรือคลาสที่น้อยกว่าด้วยสมการที่ 4.1 ในขณะที่ P(x|H) หรือคลาสที่มากกว่ายังคงค่าเดิม
8) ทำซ้ำในขั้นตอนที่ 2 - 7 เพื่อหาค่า P(x|S) และ P(x|H) ใหม่ในคำทั้งหมด ก็คือวนตรวจสอบคำทั้งหมดเพื่อดูว่าควรจะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าคำไหนบ้าง
9) นำค่า P(x|S) และ P(x|H) ของคำทั้งหมดเพื่อใช้ในการคำนวณหาความน่าจะเป็นที่จะเป็นอีเมลสแปม P(S|X) และความน่าจะเป็นที่จะเป็นอีเมลแฮม P(H|X) โดยใช้สมการที่ 5.22

$$P(C|X) = \log_{10} P(C) + \sum_{x \in X} \log_{10} P(x|C) \qquad (5.22)$$

โดยที่ P(C|X) คือความน่าจะเป็นที่จะเป็นคลาส C (อีเมลสแปม S หรืออีเมลแฮม H) เมื่อมีคำทั้งหมด (X) ในอีเมล,

X คือเซตของคำทั้งหมด,

P(C) คือความน่าจะเป็นของคลาส C,

P(x|C) คือความน่าจะเป็นของคำ x ที่เป็นคลาส C

สมการที่ 5.22 มาจากการเปลี่ยนรูปของสมการที่ 2.10 โดยการนำ $\log_{10}$ ไปคูณทั้งสองข้างของสมการที่ 2.10 ตามคุณสมบัติ log ที่แสดงในสมการที่ 5.23 จึงได้เป็นสมการที่ 5.22 ที่นำ $\log_{10}$ ไปคูณทั้งสองข้างของสมการเนื่องจากต้องการที่จะให้ค่าตัวเลขความ

น่าจะเป็นมีค่ามากขึ้น เพื่อให้การเปรียบเทียบสะดวกและง่ายยิ่งขึ้น โดยค่าที่ได้จะเป็นค่า $\log_{10}$ ของความน่าจะเป็น P(C|X) ซึ่งเราใช้เปรียบเทียบเพื่อจำแนกประเภทระหว่างคลาสอีเมลสแปมและคลาสอีเมลแฮม โดยในสมการ 5.22 จะไม่ได้ใส่ $\log_{10}$ ไว้หน้า P(C|X) ขอละไว้ในฐานที่เข้าใจว่ามันจะได้เป็นค่า $\log_{10}$ ของความน่าจะเป็นเพื่อนำมาเปรียบเทียบ

$$\log_a(M \times N) = \log_a M + \log_a N \tag{5.23}$$

สมการที่ 5.23 แสดงคุณสมบัติ log ของการคูณ M x N เมื่อ a คือฐานใดๆของ log

10) ถ้า P(S|X) มากกว่า P(H|X) จำแนกอีเมลว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

ตามที่ได้กล่าวไว้อัลกอริทึม RIWE-NB จะใช้การคำนวณค่าเอนโทรปีของคำ ถ้าค่าเอนโทรปีมีค่ามากเข้าใกล้ 1 แสดงว่าคำมีการกระจายออกเป็นคลาสอีเมลสแปมและอีเมลแฮมอย่างมาก นั้นหมายถึงความสำคัญของคำในแต่ละคลาสมีค่าที่ใกล้เคียงกัน จึงไม่ควรลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า พิจารณาขั้นตอนที่ 5 และขั้นตอนที่ 6 ของอัลกอริทึม RIWE-NB ซึ่งมีการสุ่มตัวเลข r ระหว่าง 0 – 1 และถ้าตัวเลข r มากกว่าค่าเอนโทรปีที่คำนวณได้ ถึงจะทำการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่า ซึ่งสอดคล้องกับแนวคิดหลักที่ต้องการ ยิ่งค่าเอนโทรปีมีค่ามากหรือเข้าใกล้ 1 โอกาสที่จะสุ่มตัวเลข r ให้ได้ค่าที่มากกว่าค่าเอนโทรปีจะยิ่งมีโอกาสน้อย เหตุผลที่อัลกอริทึม RIWE-NB ใช้วิธีการสุ่ม เนื่องจากไม่ต้องการให้อัลกอริทึมมีการกำหนดเทรชโฮลด์ (Threshold) เช่นถ้าห่างกันเกินกว่า x จะไม่ทำการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าเป็นต้น

อัลกอริทึม RIWE-NB เป็นอัลกอริทึมที่พัฒนาและปรับปรุงมาจากอัลกอริทึม AWF-NB เพื่อให้เข้าใจความแตกต่างระหว่างอัลกอริทึม RIWE-NB และอัลกอริทึม AWF-NB วิทยานิพนธ์นี้จะแสดงให้เห็นถึงขั้นตอนของอัลกอริทึม RIWE-NB ที่ถูกเพิ่มหรือแทนที่จากอัลกอริทึม AWF-NB ดังนั้นจะทำการปรับเปลี่ยนจากรหัสเทียมของอัลกอริทึม AWF-NB ในรูปที่ 4.2 ให้เป็นรหัสเทียมของอัลกอริทึม RIWE-NB เริ่มจากการเปลี่ยนบรรทัดที่ 1 ของรูปที่ 4.2 เป็น “function RIWE-NB (email) returns classification” ถัดไปให้เปลี่ยนบรรทัดที่ 3 และ 4 เป็น “pa_spam <— 0” และ “pa_ham <— 0” ตามลำดับ เนื่องจากอัลกอริทึม RIWE-NB มีการเปลี่ยนรูปสมการโดยการคูณ $\log_{10}$ เข้าไปทำให้สมการเปลี่ยนจากการคูณเป็นการบวกตามคุณสมบัติ log ดังนั้นค่าเริ่มต้นของ pa_spam และ pa_ham จึงต้องเท่ากับ 0 ต่อมาแทนที่คำสั่งทั้งหมดในรูปที่ 5.1 ในบรรทัดที่ 8 - 11 ของรูปที่ 4.2 และเปลี่ยนบรรทัดที่ 16 และ 17 เป็น “pa_spam <— pa_spam + pw_spam” และ “pa_ham <— pa_ham + pw_ham” ตามลำดับเพราะอัลกอริทึม RIWE-NB ได้มีการเปลี่ยนรูปสมการ รหัสเทียมแบบสมบูรณ์ของอัลกอริทึม RIWE-NB แสดงในรูปที่ 5.2

```
entropy ← Compute the entropy of word w
r ← Random number between 0 to 1
if pw_ham is less than pw_spam and pw_ham isn't equal to 0
        if r is greater than entropy,
                then reducing the importance of pw_ham
else if pw_spam is less than pw_ham and pw_spam isn't equal to 0
        if r is greater than entropy,
                then reducing the importance of pw_spam
```

รูปที่ 5.1 แสดงบางส่วนของรหัสเทียมของอัลกอริทึม RIWE-NB

```
1   function RIWE-NB (email) returns classification
        words ← List of word in email
3       pa_spam ← 0
4       pa_ham ← 0
        for each value w of words do
                pw_spam ← Compute the probability of word w in spam email
                pw_ham ← Compute the probability of word w in ham email
8               entropy ← Compute the entropy of word w
                r ← Random number between 0 to 1
                if pw_ham is less than pw_spam and pw_ham isn't equal to 0
                        if r is greater than entropy,
                                then reducing the importance of pw_ham
                else if pw_spam is less than pw_ham and pw_spam isn't equal to 0
                        if r is greater than entropy,
15                              then reducing the importance of pw_spam
16              pa_spam ← pa_spam + pw_spam
17              pa_ham ← pa_ham + pw_ham
        PS ← Compute the probability of spam using pa_spam
        PH ← Compute the probability of ham using pa_ham
        if PS is greater than PH,
                then return spam class
        else
                return ham class
```

รูปที่ 5.2 แสดงรหัสเทียมของอัลกอริทึม RIWE-NB

จากรหัสเทียมของอัลกอริทึม RIWE-NB ในรูปที่ 5.2 ฟังก์ชัน RIWE-NB จะคืนค่าผลการจำแนกประเภทอีเมล โดยจะรับพารามิเตอร์เป็นอีเมลที่ต้องการจำแนกประเภท ตัวแปร words จะเก็บรายการของคำทั้งหมดในอีเมล pa_spam คือผลบวก log ของความน่าจะเป็นของคำทั้งหมดในอีเมลสแปม ส่วน pa_ham คือผลบวก log ของความน่าจะเป็นของคำทั้งหมดในอีเมลแฮม w คือคำแต่ละคำที่อยู่ในรายการ words ทำการวนลูปทีละคำเพื่อคำนวณหา pa_spam และ pa_ham ซึ่งตอนแรกจะหาค่า pw_spam ความน่าจะเป็นที่จะมีคำ w ในอีเมลสแปม และ pw_ham ความน่าจะเป็นที่จะมีคำ w ในอีเมลแฮม จากนั้นคำนวณค่าเอนโทรปีของคำ w แล้วทำการสุ่มตัวเลขระหว่าง 0 - 1 และนำค่าที่สุ่มได้มาเก็บไว้ใน r แล้วตรวจสอบเงื่อนไขดังนี้

1) เงื่อนไขที่ 1: ถ้า pw_ham น้อยกว่า pw_spam และ pw_ham ไม่เท่ากับ 0 และ r มากกว่าค่าเอนโทรปีของคำ w แล้วให้ลดความสำคัญของ pw_ham
2) เงื่อนไขที่ 2: ถ้า pw_spam น้อยกว่า pw_ham และ pw_spam ไม่เท่ากับ 0 และ r มากกว่าค่าเอนโทรปีของคำ w แล้วให้ลดความสำคัญของ pw_spam

จากนั้นนำ pw_spam และ pw_ham ไปบวกสะสมในตัวแปร pa_spam และ pa_ham ตามลำดับจนครบทุกคำในอีเมล นำตัวแปร pa_spam ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลสแปม PS และนำตัวแปร pa_ham ไปคำนวณความน่าจะเป็นที่อีเมลจะเป็นอีเมลแฮม PH ถ้า PS มากกว่า PH จะจำแนกว่าเป็นอีเมลสแปม มิฉะนั้นจะจำแนกว่าเป็นอีเมลแฮม

บทที่ 6
ผลการทดลอง

6.1 ชุดข้อมูลที่ใช้ในการทดลอง

งานวิจัยนี้ใช้ชุดข้อมูลอีเมลสแปม 15 ชุดข้อมูลจากเว็บไซต์ Kaggle [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25] เพื่อความสะดวกในการอธิบาย จะเปลี่ยนชื่อชุดข้อมูลเหล่านี้ใหม่ว่าชุดข้อมูลอีเมลสแปม 1 – 15 เนื่องจากชุดข้อมูลจำนวนมากไม่มีชื่อที่เฉพาะ คุณสมบัติของชุดข้อมูลที่ใช้ในการทดลองทั้งหมดแสดงดังตารางถัดไป

ตารางที่ 6.1 คุณสมบัติของชุดข้อมูลที่ใช้ในการทดลอง

ชุดข้อมูล	จำนวนอีเมล	จำนวนอีเมลสแปม	จำนวนอีเมลแฮม
ชุดข้อมูลอีเมลสแปม 1 [13]	5,171	1,499	3,672
ชุดข้อมูลอีเมลสแปม 2 [14]	5,572	747	4,825
ชุดข้อมูลอีเมลสแปม 3 [15]	6,046	1,896	4,150
ชุดข้อมูลอีเมลสแปม 4 [15]	10,000	5,000	5,000
ชุดข้อมูลอีเมลสแปม 5 [15]	2,605	433	2,172
ชุดข้อมูลอีเมลสแปม 6 [16]	5,727	1,368	4,359
ชุดข้อมูลอีเมลสแปม 7 [17]	3,000	500	2,500
ชุดข้อมูลอีเมลสแปม 8 [18]	5,854	1,496	4,358
ชุดข้อมูลอีเมลสแปม 9 [19]	5,796	1,896	3,900
ชุดข้อมูลอีเมลสแปม 10 [20]	5,796	1,896	3,900
ชุดข้อมูลอีเมลสแปม 11 [21]	4,845	2,113	2,732
ชุดข้อมูลอีเมลสแปม 12 [22]	5,172	1,500	3,672
ชุดข้อมูลอีเมลสแปม 13 [23]	3,052	501	2,551
ชุดข้อมูลอีเมลสแปม 14 [24]	2,893	481	2,412
ชุดข้อมูลอีเมลสแปม 15 [25]	73,932	48,714	25,218

จากตารางข้างบนแสดงคุณสมบัติของชุดข้อมูลที่ใช้ในการทดลองทั้งหมด จะเห็นได้ว่ามีทั้งชุดข้อมูลขนาดเล็ก ขนาดกลางและขนาดใหญ่ ในการทดลองของวิทยานิพนธ์นี้พยายามจะใช้ชุดข้อมูลหลายขนาดเพื่อเปรียบเทียบประสิทธิภาพของอัลกอริทึมที่นำเสนอหรือ RIWE-NB กับอัลกอริทึม NB และอัลกอริทึม AWF-NB สำหรับวิทยานิพนธ์นี้ชุดข้อมูลที่มีจำนวนอีเมลไม่เกิน 5,000 อีเมลจะถือว่าเป็นชุดข้อมูลขนาดเล็ก และชุดข้อมูลที่มีจำนวนอีเมลตั้งแต่ 5,000 ถึง 10,000 อีเมลเป็นชุดข้อมูลขนาดกลาง ส่วนชุดข้อมูลที่มีจำนวนอีเมลมากกว่า 10,000 อีเมลขึ้นไปคือชุดข้อมูลขนาดใหญ่ ดังนั้น

ชุดข้อมูลขนาดเล็กมีทั้งหมด 5 ชุดข้อมูลได้แก่ ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 7, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 13 และชุดข้อมูลอีเมลสแปม 14 ต่อมาชุดข้อมูลขนาดกลางมีทั้งหมด 9 ชุดข้อมูลได้แก่ ชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 2, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 6, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 10 และชุดข้อมูลอีเมลสแปม 12 สุดท้ายชุดข้อมูลขนาดใหญ่มี 1 ชุดข้อมูลคือชุดข้อมูลอีเมลสแปม 15 เนื่องจากชุดข้อมูลขนาดใหญ่นั้นหาได้ยาก ในชุดข้อมูลอีเมลสแปมทั่วไปที่สามารถดาวน์โหลดมาใช้งานได้จากเว็บไซต์ Kaggle ส่วนมากจะเป็นชุดข้อมูลขนาดกลาง

6.2 เงื่อนไขในการทดลอง

6.2.1 การแบ่งชุดข้อมูลที่ใช้ในการทดลอง

ชุดข้อมูลที่ใช้ในการทดลองทั้งหมด จะถูกสุ่มแบ่งเป็น 2 ส่วนคือชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบโดยใช้สัดส่วน 50% : 50% ในการแบ่งยังคงรักษาสมดุลระหว่างคลาสอีเมลสแปมและคลาสอีเมลแฮม ตารางต่อไปนี้แสดงชุดข้อมูลตัวอย่างเพื่อสาธิตวิธีการแบ่งชุดข้อมูล

ตารางที่ 6.2 ชุดข้อมูลตัวอย่างเพื่อสาธิตวิธีการแบ่งชุดข้อมูล

อีเมล	คลาส
อีเมลที่ 1	อีเมลสแปม
อีเมลที่ 2	อีเมลแฮม
อีเมลที่ 3	อีเมลสแปม
อีเมลที่ 4	อีเมลแฮม
อีเมลที่ 5	อีเมลสแปม
อีเมลที่ 6	อีเมลแฮม
อีเมลที่ 7	อีเมลสแปม
อีเมลที่ 8	อีเมลสแปม
อีเมลที่ 9	อีเมลสแปม
อีเมลที่ 10	อีเมลสแปม

จากชุดข้อมูลตัวอย่างในตารางที่ 6.2 สมมุติว่าอีเมลทั้งหมดในชุดข้อมูลตัวอย่างมี 10 อีเมล ใน 10 อีเมลประกอบด้วยอีเมลสแปม 7 อีเมล และอีเมลแฮม 3 อีเมล ทำการสุ่มแบ่งเป็น 2 ส่วนคือชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบอย่างละ 50% โดยรักษาสมดุลของคลาสภายในชุดข้อมูล จะได้เป็นชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบที่มีรายละเอียดดังนี้

1) ชุดข้อมูลสำหรับฝึกฝน: จะมีจำนวนอีเมลสแปม 3 อีเมล และมีจำนวนอีเมลแฮม 1 อีเมล รวมเป็นอีเมลทั้งหมด 4 อีเมล
2) ชุดข้อมูลสำหรับทดสอบ: จะมีจำนวนอีเมลสแปม 4 อีเมล และมีจำนวนอีเมลแฮม 2 อีเมล รวมเป็นอีเมลทั้งหมด 6 อีเมล

เนื่องจากจำนวนอีเมลสแปมและจำนวนอีเมลแฮมทั้งหมดเป็น 7 และ 3 ตามลำดับ ซึ่งเป็นเลขคี่จึงหาร 2 ไม่ลงตัวทั้งคู่ จะนำส่วนที่ไม่ลงตัวไปเพิ่มไว้ในชุดข้อมูลสำหรับทดสอบดังตัวอย่าง ดังนั้นชุดข้อมูลสำหรับทดสอบจะมีจำนวนมากกว่าชุดข้อมูลสำหรับฝึกฝนอยู่เล็กน้อยในกรณีที่แต่ละคลาสของอีเมลมีจำนวนอีเมลเป็นจำนวนคี่ นี้คือการรักษาสมดุลของคลาสภายในชุดข้อมูล จากนั้นสุ่มหยิบอีเมลตามรายละเอียดของชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบ

1) ชุดข้อมูลสำหรับฝึกฝน: สุ่มหยิบอีเมลสแปม 3 อีเมล และอีเมลแฮม 1 อีเมล สมมุติสุ่มได้เซตของอีเมลสแปม = {อีเมล 7, อีเมล 1, อีเมล 9} และสุ่มได้เซตของอีเมลแฮม = {อีเมล 6}
2) ชุดข้อมูลสำหรับทดสอบ: เหลืออีเมลสแปม 4 อีเมล และเหลืออีเมลแฮม 2 อีเมล จากการสุ่มชุดข้อมูลสำหรับฝึกฝน ใช้อีเมลที่เหลือจากการแบ่ง จะได้เซตของอีเมลสแปม = { อีเมล 3, อีเมล 5, อีเมล 8 , อีเมล 10 } และเซตของอีเมลแฮม = { อีเมล 2, อีเมล 4 }

ตารางที่ 6.3 ชุดข้อมูลสำหรับฝึกฝนที่ได้จากการสุ่มแบ่งชุดข้อมูลตัวอย่าง

อีเมล	คลาส
อีเมลที่ 1	อีเมลสแปม
อีเมลที่ 6	อีเมลแฮม
อีเมลที่ 7	อีเมลสแปม
อีเมลที่ 9	อีเมลสแปม

ตารางที่ 6.4 ชุดข้อมูลสำหรับทดสอบที่ได้จากการสุ่มแบ่งชุดข้อมูลตัวอย่าง

อีเมล	คลาส
อีเมลที่ 2	อีเมลแฮม
อีเมลที่ 3	อีเมลสแปม
อีเมลที่ 4	อีเมลแฮม
อีเมลที่ 5	อีเมลสแปม
อีเมลที่ 8	อีเมลสแปม
อีเมลที่ 10	อีเมลสแปม

6.2.1 รูปแบบการทดลอง

อัลกอริทึม NB แบบดั้งเดิม, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB จะทำการทดลองกับชุดข้อมูล 15 ชุดข้อมูล โดยใช้ชุดข้อมูลสำหรับฝึกฝนและชุดข้อมูลสำหรับทดสอบเหมือนกันทั้ง 15 ชุดข้อมูล อัลกอริทึมทั้งหมดจะทดลองโดยใช้การทำลาปลาซสมูททิง เพื่อจัดการปัญหาความน่าจะเป็นที่มีค่าเป็นศูนย์ การทดลองในวิทยานิพนธ์นี้จะทดลองโดยปราศจากการสกัดคุณลักษณะ (Feature Extraction) อัลกอริทึม NB แบบดั้งเดิมและอัลกกอริทึม AWF-NB จะทำการทดลอง 1 ครั้งเนื่องจากอัลกอริทึม NB และอัลกอริทึม AWF-NB ไม่ว่าจะทดลองกี่ครั้งก็ให้ผลลัพธ์เหมือนเดิม ในขณะที่อัลกอริทึม RIWE-NB มีกระบวนการสุ่มอยู่ภายในอัลกอริทึม การทดลองแต่ละครั้งจะให้ผลที่ไม่เหมือนกัน ดังนั้นอัลกอริทึม RIWE-NB จะทำการทดลอง 1000 ครั้งแล้วหาค่าเฉลี่ยความแม่นยำในการจำแนกและเวลาในการดำเนินการออกมา เนื่องจากต้องการผลการทดลองที่เสถียรซึ่งการทดลอง 1000 นั้นเพียงพอ

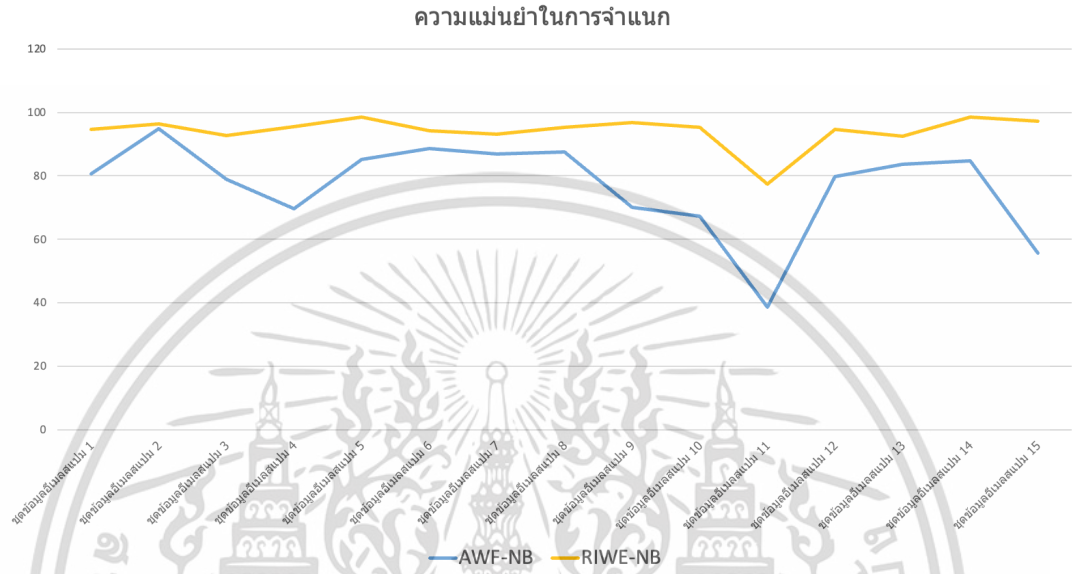
6.3 ผลการทดลองระหว่างอัลกอริทึมที่ต้องการปรับปรุง (AWF-NB) และอัลกอริทึมที่เสนอ (RIWE-NB)

6.3.1 ความแม่นยำในการจำแนก

ตารางที่ 6.5 ความแม่นยำในการจำแนกระหว่างอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

ชุดข้อมูล	ความแม่นยำในการจำแนก	
	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	80.63	**94.57**
ชุดข้อมูลอีเมลสแปม 2	94.90	**96.25**
ชุดข้อมูลอีเมลสแปม 3	78.96	**92.60**
ชุดข้อมูลอีเมลสแปม 4	69.54	**95.48**
ชุดข้อมูลอีเมลสแปม 5	85.19	**98.43**
ชุดข้อมูลอีเมลสแปม 6	88.51	**94.12**
ชุดข้อมูลอีเมลสแปม 7	86.80	**93.21**
ชุดข้อมูลอีเมลสแปม 8	87.56	**95.29**
ชุดข้อมูลอีเมลสแปม 9	70.12	**96.79**
ชุดข้อมูลอีเมลสแปม 10	67.32	**95.37**
ชุดข้อมูลอีเมลสแปม 11	38.59	**77.31**
ชุดข้อมูลอีเมลสแปม 12	79.81	**94.52**
ชุดข้อมูลอีเมลสแปม 13	83.56	**92.39**
ชุดข้อมูลอีเมลสแปม 14	84.73	**98.41**
ชุดข้อมูลอีเมลสแปม 15	55.58	**97.24**

จากตารางข้างบน ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB หรืองานวิจัยที่เสนอ มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB สำหรับชุดข้อมูลอีเมลสแปม 1 – 15 ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด



รูปที่ 6.1 แสดงกราฟความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าอัลกอริทึมความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมดเพื่อความชัดเจนในการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB วิทยานิพนธ์นี้จะแสดงร้อยละการเพิ่มขึ้นของของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB ในตารางต่อไปนี้

ตารางที่ 6.6 ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB

ชุดข้อมูล	ความแม่นยำในการจำแนก		ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB
	AWF-NB	RIWE-NB	
ชุดข้อมูลอีเมลสแปม 1	80.63	**94.57**	17.29
ชุดข้อมูลอีเมลสแปม 2	94.90	**96.25**	1.42
ชุดข้อมูลอีเมลสแปม 3	78.96	**92.60**	17.27
ชุดข้อมูลอีเมลสแปม 4	69.54	**95.48**	37.30
ชุดข้อมูลอีเมลสแปม 5	85.19	**98.43**	15.54
ชุดข้อมูลอีเมลสแปม 6	88.51	**94.12**	6.34
ชุดข้อมูลอีเมลสแปม 7	86.80	**93.21**	7.38
ชุดข้อมูลอีเมลสแปม 8	87.56	**95.29**	8.83
ชุดข้อมูลอีเมลสแปม 9	70.12	**96.79**	38.03
ชุดข้อมูลอีเมลสแปม 10	67.32	**95.37**	41.67
ชุดข้อมูลอีเมลสแปม 11	38.59	**77.31**	100.34
ชุดข้อมูลอีเมลสแปม 12	79.81	**94.52**	18.43
ชุดข้อมูลอีเมลสแปม 13	83.56	**92.39**	10.57
ชุดข้อมูลอีเมลสแปม 14	84.73	**98.41**	16.15
ชุดข้อมูลอีเมลสแปม 15	55.58	**97.24**	74.96

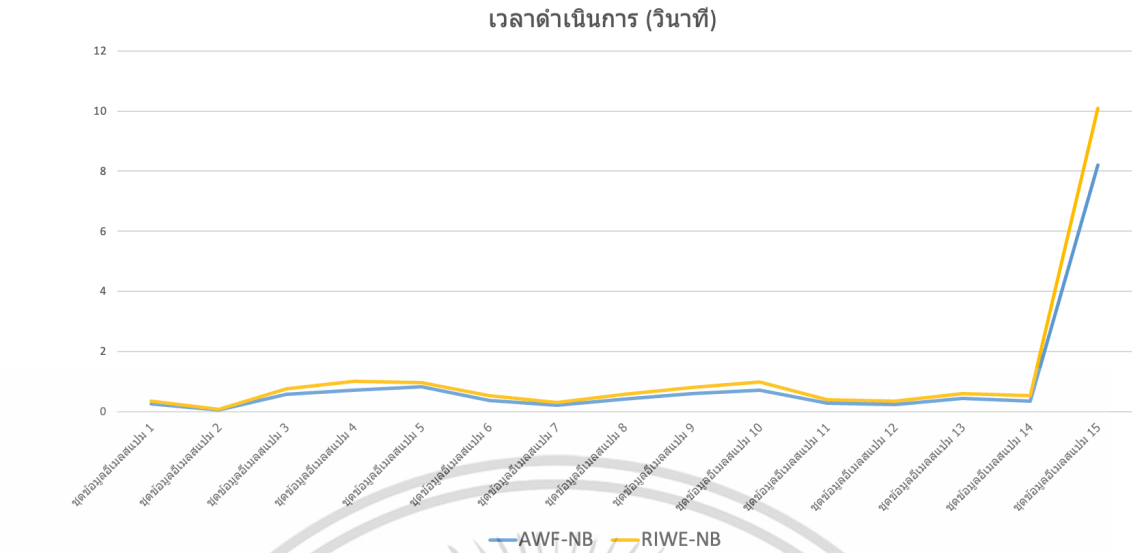
จากตารางข้างบน ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB มีการเพิ่มขึ้นของความแม่นยำอย่างเห็นได้ชัดในชุดข้อมูลส่วนมาก ได้แก่ ชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 6, ชุดข้อมูลอีเมลสแปม 7, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 10, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 12, ชุดข้อมูลอีเมลสแปม 13, ชุดข้อมูลอีเมลสแปม 14 และชุดข้อมูลอีเมลสแปม 15 โดยชุดข้อมูลเหล่านี้มีร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB ที่มากกว่า 6 เปอร์เซ็นต์ มีเพียงชุดข้อมูลอีเมลสแปม 2 เท่านั้นที่มีร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB เพียง 1.42 เปอร์เซ็นต์ อย่างไรก็ตามแม้ชุดข้อมูลอีเมลสแปม 2 จะมีร้อยละการเพิ่มขึ้นของความแม่นยำที่น้อย แต่ก็ยังเห็นว่าเพิ่มขึ้นอย่างชัดเจนเมื่อพิจารณาที่ตัวเลขความแม่นยำในการจำแนกระหว่างอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมดอย่างเห็นได้ชัด

6.3.2 เวลาดำเนินการ

ตารางที่ 6.7 เวลาการดำเนินการระหว่างอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

ชุดข้อมูล	เวลาดำเนินการ (วินาที)	
	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	**0.2483**	0.3549
ชุดข้อมูลอีเมลสแปม 2	**0.0456**	0.0657
ชุดข้อมูลอีเมลสแปม 3	**0.5804**	0.7613
ชุดข้อมูลอีเมลสแปม 4	**0.7199**	1.0013
ชุดข้อมูลอีเมลสแปม 5	**0.8288**	0.9741
ชุดข้อมูลอีเมลสแปม 6	**0.3805**	0.5417
ชุดข้อมูลอีเมลสแปม 7	**0.2093**	0.2945
ชุดข้อมูลอีเมลสแปม 8	**0.4123**	0.5797
ชุดข้อมูลอีเมลสแปม 9	**0.6055**	0.7982
ชุดข้อมูลอีเมลสแปม 10	**0.7134**	0.9904
ชุดข้อมูลอีเมลสแปม 11	**0.2778**	0.3859
ชุดข้อมูลอีเมลสแปม 12	**0.2385**	0.3464
ชุดข้อมูลอีเมลสแปม 13	**0.4303**	0.5937
ชุดข้อมูลอีเมลสแปม 14	**0.3509**	0.5246
ชุดข้อมูลอีเมลสแปม 15	**8.2080**	10.0891

จากตารางข้างบน เวลาดำเนินการของอัลกอริทึม RIWE-NcB หรืองานวิจัยที่เสนอมีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม AWF-NB ในชุดข้อมูลอีเมลสแปม 1 – 15 เนื่องจากอัลกอริทึม RIWE-NB ทำการปรับปรุงอัลกอริทึม AWF-NB โดยการเพิ่มการคำนวณค่าเอนโทรปีและการสุ่ม ดั้งนั้นเวลาดำเนินการของอัลกอริทึม RIWE-NB จึงมีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม AWF-NB ซึ่งเวลาดำเนินการที่เพิ่มขึ้นมากที่สุดของอัลกอริทึม RIWE-NB คือเวลาดำเนินการในชุดข้อมูลอีเมลสแปม 15 โดยเพิ่มขึ้นจากอัลกอริทึม AWF-NB 1.88 วินาที ในความเป็นจริงนั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก และคุ้มค่าที่อัลกอริทึม RIWE-NB จะใช้เวลาเพิ่มจากอัลกอริทึม AWF-NB อีกเล็กน้อยเพื่อให้ได้ความแม่นยำที่ดีขึ้นอย่างชัดเจนดังผลการทดลองในตารางที่ 6.5 ซึ่งสามารถสรุปได้ว่า เวลาดำเนินการโดยเฉลี่ยของอัลกอริทึม RIWE-NB มีค่ามากขึ้นจากอัลกอริทึม AWF-NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด



รูปที่ 6.2 แสดงกราฟเวลาดำเนินการของอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบเวลาดำเนินการของอัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าเวลาดำเนินการโดยเฉลี่ยของอัลกอริทึม RIWE-NB มีค่ามากขึ้นจากอัลกอริทึม AWF-NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด ตารางต่อไปนี้แสดงการเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB

ตารางที่ 6.8 การเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB
ชุดข้อมูลอีเมลสแปม 1	0.1066	42.93
ชุดข้อมูลอีเมลสแปม 2	0.0201	44.08
ชุดข้อมูลอีเมลสแปม 3	0.1809	31.17
ชุดข้อมูลอีเมลสแปม 4	0.2814	39.09
ชุดข้อมูลอีเมลสแปม 5	0.1453	17.53
ชุดข้อมูลอีเมลสแปม 6	0.1612	42.37
ชุดข้อมูลอีเมลสแปม 7	0.0852	40.71
ชุดข้อมูลอีเมลสแปม 8	0.1674	40.60
ชุดข้อมูลอีเมลสแปม 9	0.1927	31.82
ชุดข้อมูลอีเมลสแปม 10	0.2770	38.83
ชุดข้อมูลอีเมลสแปม 11	0.1081	38.91

ตารางที่ 6.8 (ต่อ)

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB
ชุดข้อมูลอีเมลสแปม 12	0.1079	45.24
ชุดข้อมูลอีเมลสแปม 13	0.1634	37.97
ชุดข้อมูลอีเมลสแปม 14	0.1737	49.50
ชุดข้อมูลอีเมลสแปม 15	1.8811	22.92

จากตารางข้างบน ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB มีการเพิ่มขึ้นของเวลาดำเนินการอย่างเห็นได้ชัดในชุดข้อมูลทั้งหมด โดยชุดข้อมูลทั้งหมดมีร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB มากกว่า 17 เปอร์เซ็นต์ ตามที่ได้อธิบายผลการทดลองเวลาดำเนินการ ถึงแม้ว่าอัลกอริทึม RIWE-NB จะใช้เวลาดำเนินการมากขึ้นจากอัลกอริทึม AWF-NB เป็นเปอร์เซ็นต์ที่สูง แต่เมื่อพิจารณาการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB แล้วจะเห็นว่าเวลาดำเนินการที่เพิ่มขึ้นของอัลกอริทึม RIWE-NB จากอัลกอริทึม AWF-NB เพิ่มขึ้นสูงสุดเพียง 1.88 วินาที เท่านั้นในชุดข้อมูลอีเมลสแปม 15 ในความเป็นจริงนั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก และคุ้มค่าที่อัลกอริทึม RIWE-NB จะใช้เวลาเพิ่มจากอัลกอริทึม AWF-NB อีกเล็กน้อยเพื่อให้ได้ความแม่นยำที่ดีขึ้นอย่างชัดเจนดังผลการทดลองในตารางที่ 6.5

6.4 ผลการทดลองระหว่างอัลกอริทึม NB และอัลกอริทึมที่เสนอ (RIWE-NB)

6.4.1 ความแม่นยำในการจำแนก

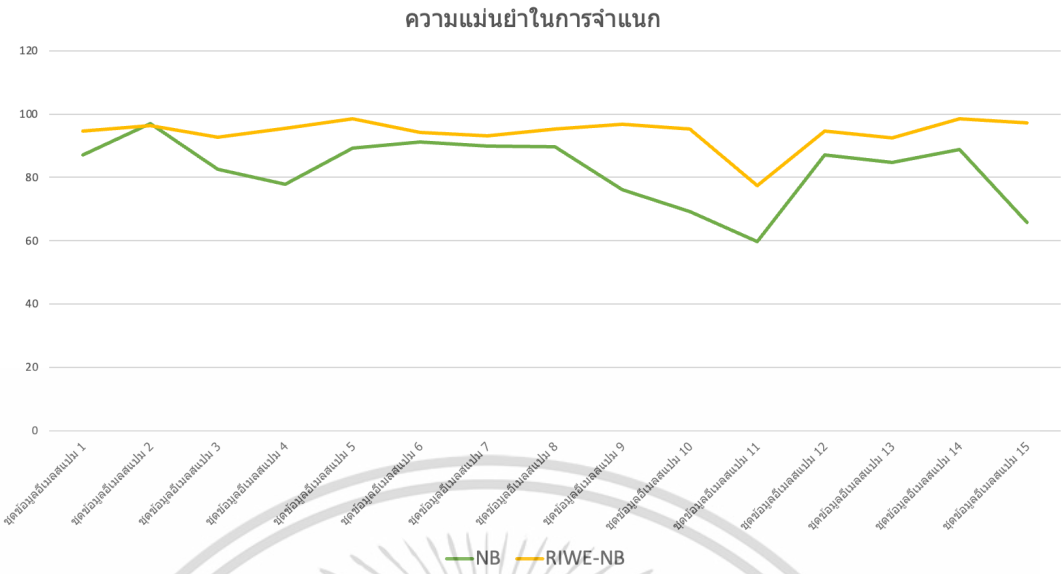
ตารางที่ 6.9 ความแม่นยำในการจำแนกระหว่างอัลกอริทึม NB และอัลกอริทึม RIWE-NB

ชุดข้อมูล	ความแม่นยำในการจำแนก	
	NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	87.16	**94.57**
ชุดข้อมูลอีเมลสแปม 2	**96.95**	96.25
ชุดข้อมูลอีเมลสแปม 3	82.47	**92.60**
ชุดข้อมูลอีเมลสแปม 4	77.86	**95.48**
ชุดข้อมูลอีเมลสแปม 5	89.18	**98.43**
ชุดข้อมูลอีเมลสแปม 6	91.13	**94.12**
ชุดข้อมูลอีเมลสแปม 7	89.87	**93.21**

ตารางที่ 6.9 (ต่อ)

ชุดข้อมูล	ความแม่นยำในการจำแนก	
	NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 8	89.75	**95.29**
ชุดข้อมูลอีเมลสแปม 9	76.12	**96.79**
ชุดข้อมูลอีเมลสแปม 10	69.25	**95.37**
ชุดข้อมูลอีเมลสแปม 11	59.72	**77.31**
ชุดข้อมูลอีเมลสแปม 12	87.12	**94.52**
ชุดข้อมูลอีเมลสแปม 13	84.68	**92.39**
ชุดข้อมูลอีเมลสแปม 14	88.74	**98.41**
ชุดข้อมูลอีเมลสแปม 15	65.81	**97.24**

จากตารางข้างบน จะเห็นว่าความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB หรืองานวิจัยที่เสนอ มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดั้งเดิมในชุดข้อมูลส่วนมากได้แก่ ชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 6, ชุดข้อมูลอีเมลสแปม 7, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 10, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 12, ชุดข้อมูลอีเมลสแปม 13, ชุดข้อมูลอีเมลสแปม 14 และชุดข้อมูลอีเมลสแปม 15 และมีเพียงชุดข้อมูลอีเมลสแปม 2 เท่านั้นที่ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่าน้อยกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดั้งเดิมเพียงเล็กน้อยไม่ถึง 1 เปอร์เซ็นต์ เนื่องจากชุดข้อมูลอีเมลสแปม 2 มีคำที่มีความสำคัญระหว่างคลาสใกล้เคียงกันเป็นจำนวนมากและทุกๆคำมีบทบาทต่อการจำแนกที่สำคัญพอๆกัน ดังนั้นการลดความสำคัญของคำในอัลกอริทึม RIWE-NB พิจารณาจากค่าเอนโทรปีและการสุ่ม ถ้าเอนโทรปีมีค่ามากโอกาสที่จะลดความสำคัญจะน้อย ถึงแม้อย่างนั้นเพราะมันคือการสุ่ม ซึ่งจะมีบ้างที่ค่าเอนโทรปีมีค่ามากแล้ว แต่ก็ยังเกิดการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าอยู่ เป็นสาเหตุให้ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB ในชุดข้อมูลอีเมลสแปม 2 ลดลงจากอัลกอริทึม NB แบบดั้งเดิมเล็กน้อย ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดังเดิมในชุดข้อมูลส่วนมาก



รูปที่ 6.3 แสดงกราฟความแม่นยำในการจำแนกของอัลกอริทึม NB และอัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบความแม่นยำในการจำแนกของอัลกอริทึม NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดังเดิมในชุดข้อมูลส่วนมาก เพื่อให้ชัดเจนในการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB วิทยานิพนธ์นี้จะแสดงร้อยละการเพิ่มขึ้นของของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB ในตารางต่อไปนี้

ตารางที่ 6.10 ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB

ชุดข้อมูล	ความแม่นยำในการจำแนก		ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม **RIWE-NB** จากอัลกอริทึม **NB**
	NB	RIWE-NB	
ชุดข้อมูลอีเมลสแปม 1	87.16	**94.57**	8.50
ชุดข้อมูลอีเมลสแปม 2	**96.95**	96.25	-0.72
ชุดข้อมูลอีเมลสแปม 3	82.47	**92.60**	12.28
ชุดข้อมูลอีเมลสแปม 4	77.86	**95.48**	22.63
ชุดข้อมูลอีเมลสแปม 5	89.18	**98.43**	10.37
ชุดข้อมูลอีเมลสแปม 6	91.13	**94.12**	3.28
ชุดข้อมูลอีเมลสแปม 7	89.87	**93.21**	3.72
ชุดข้อมูลอีเมลสแปม 8	89.75	**95.29**	6.17
ชุดข้อมูลอีเมลสแปม 9	76.12	**96.79**	27.15
ชุดข้อมูลอีเมลสแปม 10	69.25	**95.37**	37.72

ตารางที่ 6.10 (ต่อ)

ชุดข้อมูล	ความแม่นยำในการจำแนก		ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม **RIWE-NB** จากอัลกอริทึม **NB**
	NB	RIWE-NB	
ชุดข้อมูลอีเมลสแปม 11	59.72	**77.31**	29.45
ชุดข้อมูลอีเมลสแปม 12	87.12	**94.52**	8.49
ชุดข้อมูลอีเมลสแปม 13	84.68	**92.39**	9.10
ชุดข้อมูลอีเมลสแปม 14	88.74	**98.41**	10.90
ชุดข้อมูลอีเมลสแปม 15	65.81	**97.24**	47.76

จากตารางข้างบน ร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB มีการเพิ่มขึ้นของความแม่นยำอย่างเห็นได้ชัดในชุดข้อมูลส่วนมากได้แก่ ชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 10, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 12, ชุดข้อมูลอีเมลสแปม 13, ชุดข้อมูลอีเมลสแปม 14 และชุดข้อมูลอีเมลสแปม 15 โดยชุดข้อมูลเหล่านี้มีร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB มากกว่า 6 เปอร์เซ็นต์ ส่วนชุดข้อมูลอีเมลสแปม 6 และชุดข้อมูลอีเมลสแปม 7 มีร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB ประมาณ 3 เปอร์เซ็นต์ มีเพียงชุดข้อมูลอีเมลสแปม 2 เท่านั้นที่มีร้อยละการเพิ่มขึ้นของความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB ลดลง 0.72 เปอร์เซ็นต์ อย่างไรก็ตามแม้ชุดข้อมูลอีเมลสแปม 2 จะมีความแม่นยำในการจำแนกที่ลดลง แต่ก็ลดลงเพียงเล็กน้อยเท่านั้นไม่ถึง 1 เปอร์เซ็นต์ เมื่อพิจารณาที่ตัวเลขความแม่นยำในการจำแนกระหว่างอัลกอริทึม NB และอัลกอริทึม RIWE-NB ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB ในชุดข้อมูลส่วนมากอย่างเห็นได้ชัด

6.4.2 เวลาดำเนินการ

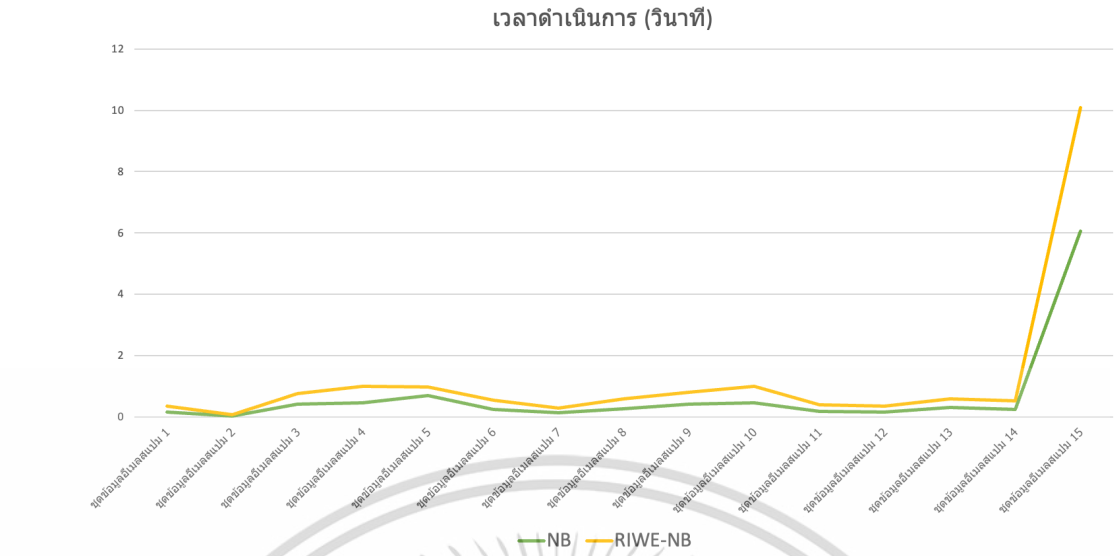
ตารางที่ 6.11 เวลาการดำเนินการระหว่างอัลกอริทึม NB และอัลกอริทึม RIWE-NB

ชุดข้อมูล	เวลาดำเนินการ (วินาที)	
	NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	**0.1605**	0.3549
ชุดข้อมูลอีเมลสแปม 2	**0.0310**	0.0657
ชุดข้อมูลอีเมลสแปม 3	**0.4209**	0.7613
ชุดข้อมูลอีเมลสแปม 4	**0.4683**	1.0013

ตารางที่ 6.11 (ต่อ)

ชุดข้อมูล	เวลาดำเนินการ (วินาที)	
	NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 5	**0.6901**	0.9741
ชุดข้อมูลอีเมลสแปม 6	**0.2418**	0.5417
ชุดข้อมูลอีเมลสแปม 7	**0.1356**	0.2945
ชุดข้อมูลอีเมลสแปม 8	**0.2659**	0.5797
ชุดข้อมูลอีเมลสแปม 9	**0.4266**	0.7982
ชุดข้อมูลอีเมลสแปม 10	**0.4641**	0.9904
ชุดข้อมูลอีเมลสแปม 11	**0.1827**	0.3859
ชุดข้อมูลอีเมลสแปม 12	**0.1551**	0.3464
ชุดข้อมูลอีเมลสแปม 13	**0.3182**	0.5937
ชุดข้อมูลอีเมลสแปม 14	**0.2345**	0.5246
ชุดข้อมูลอีเมลสแปม 15	**6.0690**	10.0891

จากตารางข้างบน เวลาดำเนินการของอัลกอริทึม RIWE-NB หรืองานวิจัยที่เสนอ มีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม NB ในชุดข้อมูลอีเมลสแปม 1 – 15 เนื่องจากอัลกอริทึม RIWE-NB ทำการปรับปรุงอัลกอริทึม AWF-NB โดยการเพิ่มการคำนวณค่าเอนโทรปีและการสุ่ม และอัลกอริทึม AWF-NB ก็ทำการปรับปรุงอัลกอริทึม NB แบบดั้งเดิมโดยการเพิ่มขั้นตอนการตรวจสอบเงื่อนไขและขั้นตอนการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า ดังนั้นเวลาดำเนินการของอัลกอริทึม RIWE-NB จึงมีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม NB แบบดั้งเดิม ซึ่งเวลาดำเนินการที่เพิ่มขึ้นมากที่สุดของอัลกอริทึม RIWE-NB คือเวลาดำเนินการในชุดข้อมูลอีเมลสแปม 15 โดยเพิ่มขึ้นจากอัลกอริทึม NB 4.02 วินาที ในความเป็นจริงนั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก และคุ้มค่าที่อัลกอริทึม RIWE-NB จะใช้เวลาเพิ่มจากอัลกอริทึม NB อีกเล็กน้อยเพื่อให้ได้ความแม่นยำที่ดีขึ้นอย่างชัดเจนดังผลการทดลองในตารางที่ 6.9 ซึ่งสามารถสรุปได้ว่า เวลาดำเนินการของอัลกอริทึม RIWE-NB มีค่ามากขึ้นจากอัลกอริทึม NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด



รูปที่ 6.4 แสดงกราฟเวลาดำเนินการของอัลกอริทึม NB และอัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบเวลาดำเนินการของอัลกอริทึม NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าเวลาดำเนินการของอัลกอริทึม RIWE-NB มีค่ามากขึ้นจากอัลกอริทึม NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด ตารางต่อไปนี้แสดงการเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB

ตารางที่ 6.12 การเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB
ชุดข้อมูลอีเมลสแปม 1	0.1944	121.12
ชุดข้อมูลอีเมลสแปม 2	0.0347	111.94
ชุดข้อมูลอีเมลสแปม 3	0.3404	80.87
ชุดข้อมูลอีเมลสแปม 4	0.5330	113.82
ชุดข้อมูลอีเมลสแปม 5	0.2840	41.15
ชุดข้อมูลอีเมลสแปม 6	0.2999	124.03
ชุดข้อมูลอีเมลสแปม 7	0.1589	117.18
ชุดข้อมูลอีเมลสแปม 8	0.3138	118.01
ชุดข้อมูลอีเมลสแปม 9	0.3716	87.11
ชุดข้อมูลอีเมลสแปม 10	0.5263	113.40
ชุดข้อมูลอีเมลสแปม 11	0.2032	111.22

ตารางที่ 6.12 (ต่อ)

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB
ชุดข้อมูลอีเมลสแปม 12	0.1913	123.34
ชุดข้อมูลอีเมลสแปม 13	0.2755	86.58
ชุดข้อมูลอีเมลสแปม 14	0.2901	123.71
ชุดข้อมูลอีเมลสแปม 15	4.0201	66.24

จากตารางข้างบน ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB มีการเพิ่มขึ้นของเวลาดำเนินการอย่างเห็นได้ชัดในชุดข้อมูลทั้งหมด โดยชุดข้อมูลทั้งหมดมีร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB มากกว่า 40 เปอร์เซ็นต์ ตามที่ได้อธิบายผลการทดลองเวลาดำเนินการ ถึงแม้ว่าอัลกอริทึม RIWE-NB จะใช้เวลาดำเนินการมากขึ้นจากอัลกอริทึม NB เป็นเปอร์เซ็นต์ที่สูง แต่เมื่อพิจารณาการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB แล้วจะเห็นว่าเวลาดำเนินการที่เพิ่มขึ้นของอัลกอริทึม RIWE-NB จากอัลกอริทึม NB เวลาเพิ่มขึ้นสูงสุดเพียง 4.02 วินาที เท่านั้นในชุดข้อมูลอีเมลสแปม 15 ในความเป็นจริงนั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก และคุ้มค่าที่อัลกอริทึม RIWE-NB จะใช้เวลาเพิ่มจากอัลกอริทึม NB อีกเล็กน้อยเพื่อให้ได้ความแม่นยำที่ดีขึ้นอย่างชัดเจนดังผลการทดลองในตารางที่ 6.9

6.5 ผลการทดลองระหว่างอัลกอริทึม NB และอัลกอริทึมที่ต้องการปรับปรุง (AWF-NB)

6.5.1 ความแม่นยำในการจำแนก

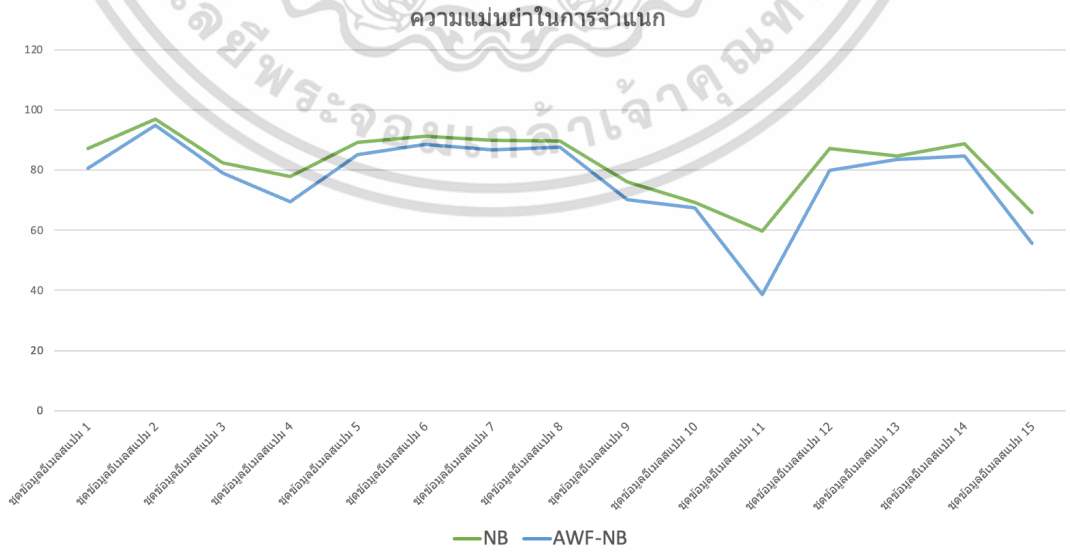
ตารางที่ 6.13 ความแม่นยำในการจำแนกระหว่างอัลกอริทึม NB และอัลกอริทึม AWF-NB

ชุดข้อมูล	ความแม่นยำในการจำแนก	
	NB	AWF-NB
ชุดข้อมูลอีเมลสแปม 1	**87.16**	80.63
ชุดข้อมูลอีเมลสแปม 2	**96.95**	94.90
ชุดข้อมูลอีเมลสแปม 3	**82.47**	78.96
ชุดข้อมูลอีเมลสแปม 4	**77.86**	69.54
ชุดข้อมูลอีเมลสแปม 5	**89.18**	85.19
ชุดข้อมูลอีเมลสแปม 6	**91.13**	88.51
ชุดข้อมูลอีเมลสแปม 7	**89.87**	86.80

ตารางที่ 6.13 (ต่อ)

ชุดข้อมูล	ความแม่นยำในการจำแนก	
	NB	AWF-NB
ชุดข้อมูลอีเมลสแปม 8	**89.75**	87.56
ชุดข้อมูลอีเมลสแปม 9	**76.12**	70.12
ชุดข้อมูลอีเมลสแปม 10	**69.25**	67.32
ชุดข้อมูลอีเมลสแปม 11	**59.72**	38.59
ชุดข้อมูลอีเมลสแปม 12	**87.12**	79.81
ชุดข้อมูลอีเมลสแปม 13	**84.68**	83.56
ชุดข้อมูลอีเมลสแปม 14	**88.74**	84.73
ชุดข้อมูลอีเมลสแปม 15	**65.81**	55.58

จากตารางข้างบน ความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB มีค่าน้อยกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดั้งเดิมในชุดข้อมูลอีเมลสแปม 1 – 15 ที่ความแม่นยำของอัลกอริทึม AWF-NB ลดลงจากอัลกอริทึม NB แบบดั้งเดิมเพราะการกระทำที่ไปลดความสำคัญของคำที่มีความสำคัญระหว่างคลาสแตกต่างกันเพียงเล็กน้อยลงอย่างมาก เมื่อทดลองอัลกอริทึม AWF-NB ที่ปราศจากการสกัดคุณลักษณะ (Feature Extraction) เช่นการทดลองในวิทยานิพนธ์นี้ จึงส่งผลให้ความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB ลดลงจากอัลกอริทึม NB แบบดั้งเดิม ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB มีค่าน้อยกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB ในชุดข้อมูลทั้งหมด



รูปที่ 6.5 แสดงกราฟความแม่นยำในการจำแนกของอัลกอริทึม NB และอัลกอริทึม AWF-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบความแม่นยำในการจำแนกของอัลกอริทึม NB และอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB มีค่าน้อยกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB ในชุดข้อมูลทั้งหมด เพื่อให้ชัดเจนในการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB วิทยานิพนธ์นี้จะแสดงร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB ในตารางต่อไปนี้

ตารางที่ 6.14 ร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB

ชุดข้อมูล	ความแม่นยำในการจำแนก		ร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม **AWF-NB** จากอัลกอริทึม **NB**
	NB	AWF-NB	
ชุดข้อมูลอีเมลสแปม 1	**87.16**	80.63	7.49
ชุดข้อมูลอีเมลสแปม 2	**96.95**	94.90	2.11
ชุดข้อมูลอีเมลสแปม 3	**82.47**	78.96	4.26
ชุดข้อมูลอีเมลสแปม 4	**77.86**	69.54	10.69
ชุดข้อมูลอีเมลสแปม 5	**89.18**	85.19	4.47
ชุดข้อมูลอีเมลสแปม 6	**91.13**	88.51	2.88
ชุดข้อมูลอีเมลสแปม 7	**89.87**	86.80	3.42
ชุดข้อมูลอีเมลสแปม 8	**89.75**	87.56	2.44
ชุดข้อมูลอีเมลสแปม 9	**76.12**	70.12	7.88
ชุดข้อมูลอีเมลสแปม 10	**69.25**	67.32	2.79
ชุดข้อมูลอีเมลสแปม 11	**59.72**	38.59	35.38
ชุดข้อมูลอีเมลสแปม 12	**87.12**	79.81	8.39
ชุดข้อมูลอีเมลสแปม 13	**84.68**	83.56	1.32
ชุดข้อมูลอีเมลสแปม 14	**88.74**	84.73	4.52
ชุดข้อมูลอีเมลสแปม 15	**65.81**	55.58	15.54

จากตารางข้างบน ร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB มีการลดลงของความแม่นยำอย่างเห็นได้ชัดในชุดข้อมูลส่วนมากได้แก่ ชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 12 และชุดข้อมูลอีเมลสแปม 15 โดยชุดข้อมูลเหล่านี้มีร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB มากกว่า 7 เปอร์เซ็นต์ ส่วนชุดข้อมูลอีเมลสแปม 2, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 6, ชุดข้อมูลอีเมลสแปม 7, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 10, ชุดข้อมูลอีเมลสแปม 13 และ

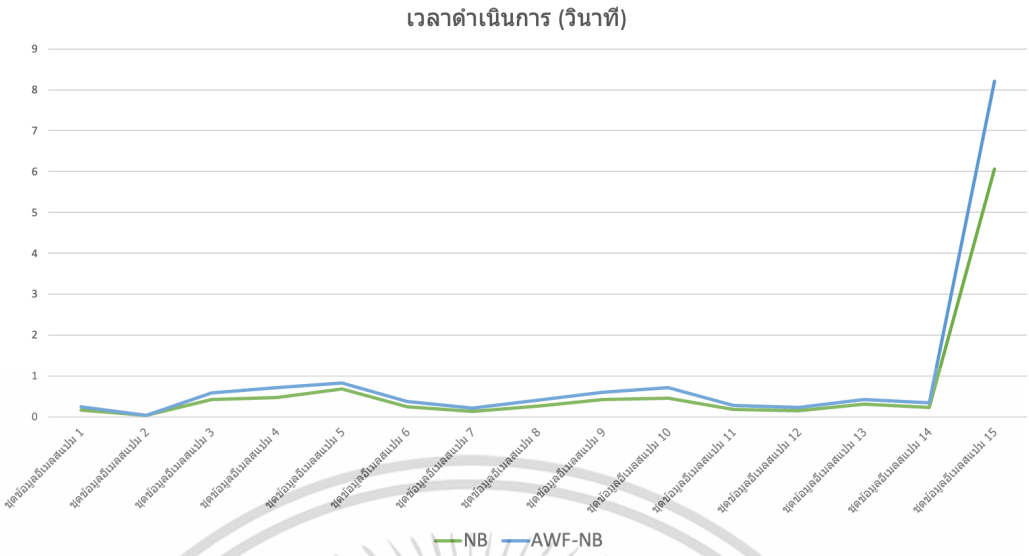
ชุดข้อมูลอีเมลสแปม 14 มีร้อยละการลดลงของความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB จากอัลกอริทึม NB ประมาณ 1 - 5 เปอร์เซ็นต์ ซึ่งสามารถสรุปได้ว่า ความแม่นยำในการจำแนกของอัลกอริทึม AWF-NB มีค่าน้อยกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB ในชุดข้อมูลทั้งหมดอย่างเห็นได้ชัด

6.5.2 เวลาในการดำเนินการ

ตารางที่ 6.15 เวลาในการดำเนินการระหว่างอัลกอริทึม NB และอัลกอริทึม AWF-NB

ชุดข้อมูล	เวลาในการดำเนินการ (วินาที)	
	NB	AWF-NB
ชุดข้อมูลอีเมลสแปม 1	0.1605	0.2483
ชุดข้อมูลอีเมลสแปม 2	0.0310	0.0456
ชุดข้อมูลอีเมลสแปม 3	0.4209	0.5804
ชุดข้อมูลอีเมลสแปม 4	0.4683	0.7199
ชุดข้อมูลอีเมลสแปม 5	0.6901	0.8288
ชุดข้อมูลอีเมลสแปม 6	0.2418	0.3805
ชุดข้อมูลอีเมลสแปม 7	0.1356	0.2093
ชุดข้อมูลอีเมลสแปม 8	0.2659	0.4123
ชุดข้อมูลอีเมลสแปม 9	0.4266	0.6055
ชุดข้อมูลอีเมลสแปม 10	0.4641	0.7134
ชุดข้อมูลอีเมลสแปม 11	0.1827	0.2778
ชุดข้อมูลอีเมลสแปม 12	0.1551	0.2385
ชุดข้อมูลอีเมลสแปม 13	0.3182	0.4303
ชุดข้อมูลอีเมลสแปม 14	0.2345	0.3509
ชุดข้อมูลอีเมลสแปม 15	6.0690	8.2080

จากตารางข้างบน เวลาดำเนินการของอัลกอริทึม AWF-NB มีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม NB แบบดั้งเดิมในชุดข้อมูลอีเมลสแปม 1 – 15 เนื่องจากอัลกอริทึม AWF-NB ทำการปรับปรุงอัลกอริทึม NB โดยการเพิ่มขั้นตอนการตรวจสอบเงื่อนไขและขั้นตอนการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า ดังนั้นเวลาดำเนินการของอัลกอริทึม AWF-NB จึงมีค่ามากกว่าเวลาดำเนินการของอัลกอริทึม NB ซึ่งเวลาดำเนินการที่เพิ่มขึ้นมากที่สุดของอัลกอริทึม AWF-NB คือเวลาดำเนินการในชุดข้อมูลอีเมลสแปม 15 โดยเพิ่มขึ้นจากอัลกอริทึม NB 2.14 วินาที ถ้ามองในความเป็นจริงแล้ว นั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก ซึ่งสามารถสรุปได้ว่า เวลาดำเนินการของอัลกอริทึม AWF-NB มีค่ามากขึ้นจากอัลกอริทึม NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด



รูปที่ 6.6 แสดงกราฟเวลาดำเนินการของอัลกอริทึม NB และอัลกอริทึม AWF-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบเวลาดำเนินการของอัลกอริทึม NB และอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด จากกราฟจะเห็นได้ว่าเวลาดำเนินการของอัลกอริทึม AWF-NB มีค่ามากขึ้นจากอัลกอริทึม NB เพียงเล็กน้อยในชุดข้อมูลทั้งหมด ตารางต่อไปนี้แสดงการเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB

ตารางที่ 6.16 การเพิ่มขึ้นของเวลาดำเนินการและร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB
ชุดข้อมูลอีเมลสแปม 1	0.0878	54.70
ชุดข้อมูลอีเมลสแปม 2	0.0146	47.10
ชุดข้อมูลอีเมลสแปม 3	0.1595	37.89
ชุดข้อมูลอีเมลสแปม 4	0.2516	53.73
ชุดข้อมูลอีเมลสแปม 5	0.1387	20.10
ชุดข้อมูลอีเมลสแปม 6	0.1387	57.36
ชุดข้อมูลอีเมลสแปม 7	0.0737	54.35
ชุดข้อมูลอีเมลสแปม 8	0.1464	55.06
ชุดข้อมูลอีเมลสแปม 9	0.1789	41.94
ชุดข้อมูลอีเมลสแปม 10	0.2493	53.72
ชุดข้อมูลอีเมลสแปม 11	0.0951	52.05

ตารางที่ 6.16 (ต่อ)

ชุดข้อมูล	การเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB (วินาที)	ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB
ชุดข้อมูลอีเมลสแปม 12	0.0834	53.77
ชุดข้อมูลอีเมลสแปม 13	0.1121	35.23
ชุดข้อมูลอีเมลสแปม 14	0.1164	49.64
ชุดข้อมูลอีเมลสแปม 15	2.1390	35.24

จากตารางข้างบน ร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB มีการเพิ่มขึ้นของเวลาดำเนินการอย่างเห็นได้ชัดในชุดข้อมูลทั้งหมด โดยชุดข้อมูลทั้งหมดมีร้อยละการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB มากกว่า 20 เปอร์เซ็นต์ ตามที่ได้อธิบายผลการทดลองเวลาดำเนินการ ถึงแม้ว่าอัลกอริทึม AWF-NB จะใช้เวลาดำเนินการมากขึ้นจากอัลกอริทึม NB เป็นเปอร์เซ็นต์ที่สูง แต่เมื่อพิจารณาการเพิ่มขึ้นของเวลาดำเนินการของอัลกอริทึม AWF-NB จากอัลกอริทึม NB แล้วจะเห็นว่าเวลาดำเนินการที่เพิ่มขึ้นของอัลกอริทึม AWF-NB จากอัลกอริทึม NB เพิ่มขึ้นสูงสุดเพียง 2.14 วินาที ในชุดข้อมูลอีเมลสแปม 15 ในความเป็นจริงนั้นถือว่าเป็นเวลาที่ห่างกันไม่มาก อย่างไรก็ตามไม่คุ้มค่าที่จะใช้อัลกอริทึม AWF-NB โดยปราศจากการสกัดคุณลักษณะ เนื่องจากการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าเพียงเล็กน้อย ส่งผลให้ความแม่นยำลดลง

6.6 ผลการทดลองระหว่างอัลกอริทึม NB, อัลกอริทึมที่ต้องการปรับปรุง (AWF-NB) และอัลกอริทึมที่เสนอ (RIWE-NB)

6.6.1 ความแม่นยำในการจำแนก

ตารางที่ 6.17 ความแม่นยำในการจำแนกระหว่างอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

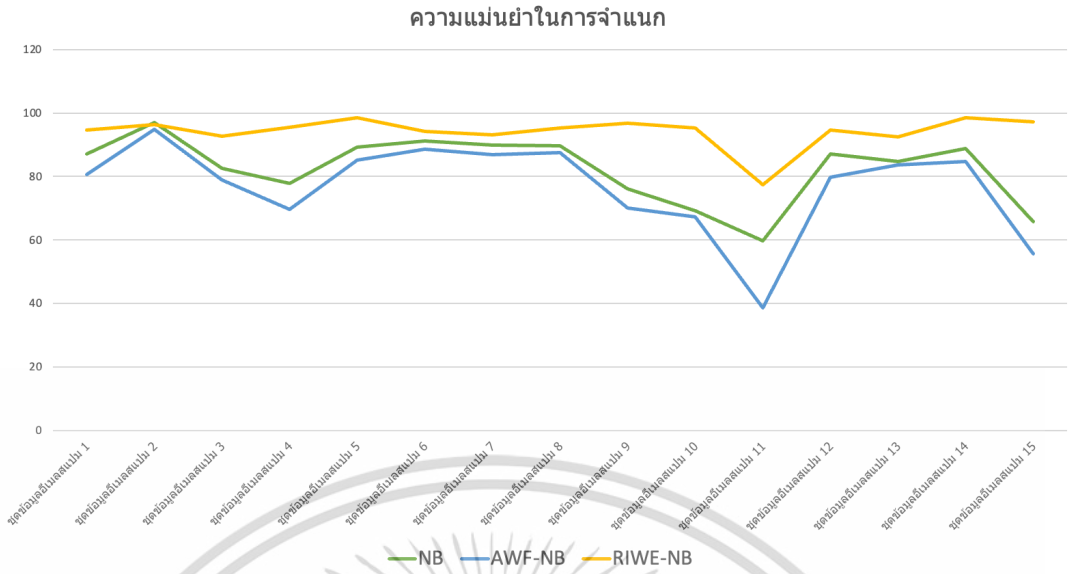
ชุดข้อมูล	ความแม่นยำในการจำแนก		
	NB	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	87.16	80.63	**94.57**
ชุดข้อมูลอีเมลสแปม 2	**96.95**	94.90	96.25
ชุดข้อมูลอีเมลสแปม 3	82.47	78.96	**92.60**
ชุดข้อมูลอีเมลสแปม 4	77.86	69.54	**95.48**

ตารางที่ 6.17 (ต่อ)

ชุดข้อมูล	ความแม่นยำในการจำแนก		
	NB	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 5	89.18	85.19	**98.43**
ชุดข้อมูลอีเมลสแปม 6	91.13	88.51	**94.12**
ชุดข้อมูลอีเมลสแปม 7	89.87	86.80	**93.21**
ชุดข้อมูลอีเมลสแปม 8	89.75	87.56	**95.29**
ชุดข้อมูลอีเมลสแปม 9	76.12	70.12	**96.79**
ชุดข้อมูลอีเมลสแปม 10	69.25	67.32	**95.37**
ชุดข้อมูลอีเมลสแปม 11	59.72	38.59	**77.31**
ชุดข้อมูลอีเมลสแปม 12	87.12	79.81	**94.52**
ชุดข้อมูลอีเมลสแปม 13	84.68	83.56	**92.39**
ชุดข้อมูลอีเมลสแปม 14	88.74	84.73	**98.41**
ชุดข้อมูลอีเมลสแปม 15	65.81	55.58	**97.24**

จากตารางข้างบนแสดงให้เห็นว่า ความแม่นยำในการจำแนกของอัลกอริทึม RIWE-NB มีค่ามากกว่าความแม่นยำในการจำแนกของอัลกอริทึม NB แบบดั้งเดิมและอัลกอริทึม AWF-NB ในชุดข้อมูลอีเมลสแปม 1, ชุดข้อมูลอีเมลสแปม 3, ชุดข้อมูลอีเมลสแปม 4, ชุดข้อมูลอีเมลสแปม 5, ชุดข้อมูลอีเมลสแปม 6, ชุดข้อมูลอีเมลสแปม 7, ชุดข้อมูลอีเมลสแปม 8, ชุดข้อมูลอีเมลสแปม 9, ชุดข้อมูลอีเมลสแปม 10, ชุดข้อมูลอีเมลสแปม 11, ชุดข้อมูลอีเมลสแปม 12, ชุดข้อมูลอีเมลสแปม 13, ชุดข้อมูลอีเมลสแปม 14 และชุดข้อมูลอีเมลสแปม 15 มีเพียงชุดข้อมูลอีเมลสแปม 2 ที่อัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกน้อยกว่าอัลกอริทึม NB แบบดั้งเดิม ในส่วนของอัลกอริทึม AWF-NB อัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกมากกว่าอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด ซึ่งสามารถสรุปได้ว่า อัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกที่มากกว่าอัลกอริทึม NB แบบดั้งเดิมและอัลกอริทึม AWF-NB อย่างเห็นได้ชัดในชุดข้อมูลส่วนมาก

เมื่อเปรียบเทียบความแม่นยำในการจำแนกระหว่างอัลกอริทึม AWF-NB และอัลกอริทึม NB แบบดั้งเดิม สามารถสรุปได้ว่า อัลกอริทึม AWF-NB มีความแม่นยำในการจำแนกน้อยว่าอัลกอริทึม NB แบบดั้งเดิมอย่างเห็นได้ชัดในชุดข้อมูลทั้งหมด ตามที่ได้อธิบายผลการทดลองระว่างอัลกอริทึม AWF-NB และอัลกอริทึม NB ในตารางที่ 6.13 ที่ความแม่นยำของอัลกอริทึม AWF-NB ลดลงจากอัลกอริทึม NB แบบดั้งเดิมเพราะการกระทำที่ไปลดความสำคัญของคำที่มีความสำคัญระหว่างคลาสแตกต่างกันเพียงเล็กน้อยลงอย่างมาก เมื่อทดลองอัลกอริทึม AWF-NB ที่ปราศจากการสกัดคุณลักษณะ จึงส่งผลให้ความแม่นยำในการจำแนกของอัลกอรึทึม AWF-NB ลดลงจากอัลกอริทึม NB แบบดั้งเดิม



รูปที่ 6.7 แสดงกราฟความแม่นยำในการจำแนกของอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นภาพรวมการเปรียบเทียบความแม่นยำในการจำแนกของอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด ซึ่งสามารถสรุปได้ว่า อัลกอริทึม RIWE-NB มีความแม่นยำมากที่สุดในชุดข้อมูลส่วนมาก และมีเพียงชุดข้อมูลอีเมลสแปม 2 เท่านั้นที่มีความแม่นยำในการจำแนกน้อยกว่าอัลกอริทึม NB เล็กน้อย สำหรับการเปรียบเทียบอัลกอริทึม RIWE-NB กับอัลกอริทึม AWF-NB จากกราฟคืออัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกมากกว่าอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด และสำหรับการเปรียบเทียบอัลกอริทึม NB กับอัลกอริทึม AWF-NB จากกราฟคืออัลกอริทึม AWF-NB มีความแม่นยำในการจำแนกน้อยกว่าอัลกอริทึม NB แบบดั้งเดิมในชุดข้อมูลทั้งหมด

6.6.2 เวลาในการดำเนินการ

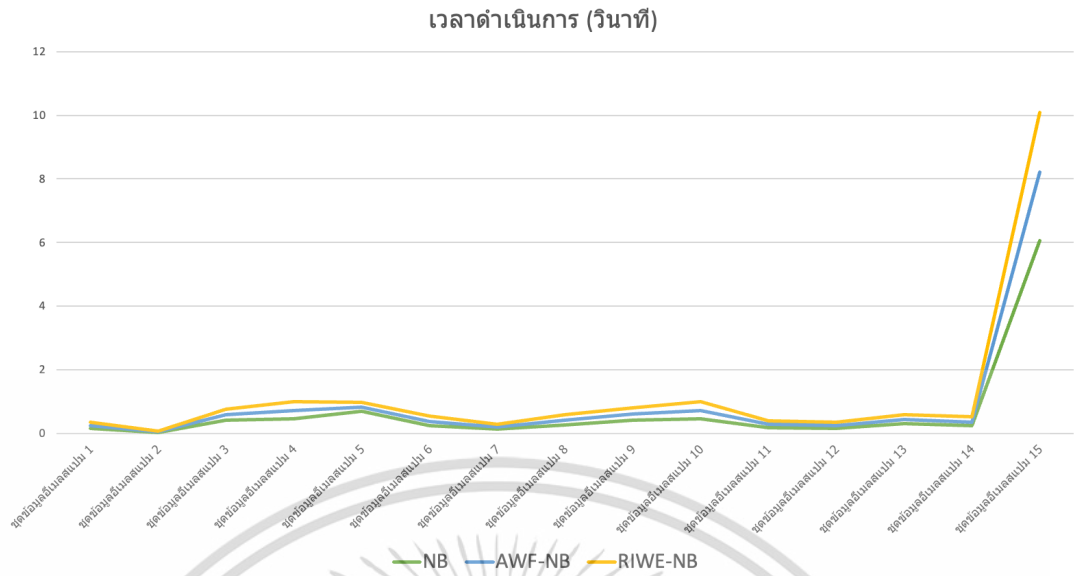
ตารางที่ 6.18 เวลาในการดำเนินการระหว่างอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB

ชุดข้อมูล	เวลาในการดำเนินการ (วินาที)		
	NB	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 1	**0.1605**	0.2483	0.3549
ชุดข้อมูลอีเมลสแปม 2	**0.0310**	0.0456	0.0657
ชุดข้อมูลอีเมลสแปม 3	**0.4209**	0.5804	0.7613
ชุดข้อมูลอีเมลสแปม 4	**0.4683**	0.7199	1.0013
ชุดข้อมูลอีเมลสแปม 5	**0.6901**	0.8288	0.9741

ตารางที่ 6.18 (ต่อ)

ชุดข้อมูล	เวลาในการดำเนินการ (วินาที)		
	NB	AWF-NB	RIWE-NB
ชุดข้อมูลอีเมลสแปม 6	**0.2418**	0.3805	0.5417
ชุดข้อมูลอีเมลสแปม 7	**0.1356**	0.2093	0.2945
ชุดข้อมูลอีเมลสแปม 8	**0.2659**	0.4123	0.5797
ชุดข้อมูลอีเมลสแปม 9	**0.4266**	0.6055	0.7982
ชุดข้อมูลอีเมลสแปม 10	**0.4641**	0.7134	0.9904
ชุดข้อมูลอีเมลสแปม 11	**0.1827**	0.2778	0.3859
ชุดข้อมูลอีเมลสแปม 12	**0.1551**	0.2385	0.3464
ชุดข้อมูลอีเมลสแปม 13	**0.3182**	0.4303	0.5937
ชุดข้อมูลอีเมลสแปม 14	**0.2345**	0.3509	0.5246
ชุดข้อมูลอีเมลสแปม 15	**6.0690**	8.2080	10.0891

จากตารางข้างบน เวลาดำเนินการที่มากที่สุดคืออัลกอริทึม RIWE-NB, อัลกอริทึม AWF-NB และอัลกอริทึม NB ตามลำดับ ซึ่งสมเหตุสมผลเนื่องจากอัลกอริทึม AWF-NB ทำการปรับปรุงอัลกอริทึม NB โดยการเพิ่มขั้นตอนการตรวจสอบเงื่อนไขและขั้นตอนการลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่า ต่อมาอัลกอริทึม RIWE-NB ทำการปรับปรุงอัลกอริทึม AWF-NB ต่อด้วยการเพิ่มขั้นตอนการคำนวณค่าเอนโทรปีและการสุ่มเข้าไปอีก ดังนั้นเวลาดำเนินการจึงเพิ่มขึ้นตามลำดับ โดยอัลกอริทึม RIWE-NB มีเวลาดำเนินการที่เพิ่มขึ้นมากที่สุดจากอัลกอริทึม AWF-NB เท่ากับ 1.88 วินาที และมีเวลาดำเนินการที่เพิ่มขึ้นมากที่สุดจากอัลกอริทึม NB เท่ากับ 4.02 วินาที ในชุดข้อมูลสแปมอีเมล 15 พิจารณาตัวเลขเวลาที่เพิ่มขึ้นสูงสุดในอัลกอริทึม RIWE-NB ซึ่งเป็นอัลกอริทึมที่ใช้เวลาดำเนินการมากที่สุด จะเห็นว่าในความเป็นจริงแล้วเวลานั้นแตกต่างกันไม่มากและค่อนข้างใกล้เคียงกัน ซึ่งสรุปได้ว่าอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB มีเวลาในการดำเนินการที่ใกล้เคียงกัน



รูปที่ 6.8 แสดงกราฟเวลาดำเนินการของอัลกอริทึม NB, อัลกอริทึม AWF-NB และ อัลกอริทึม RIWE-NB

จากกราฟด้านบนแสดงให้เห็นถึงภาพรวมของการเปรียบเทียบเวลาดำเนินการของอัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ในชุดข้อมูลทั้งหมด ซึ่งอัลกอริทึมที่ใช้เวลาดำเนินการมากที่สุดคืออัลกอริทึม RIWE-NB, อัลกอริทึม AWF-NB และอัลกอริทึม NB ตามลำดับ จากกราฟจะเห็นได้ว่าอัลกอริทึมทั้งสามใช้เวลาในการดำเนินการแตกต่างกันเพียงเล็กน้อยเท่านั้น ซึ่งสามารถสรุปได้ว่า อัลกอริทึม NB, อัลกอริทึม AWF-NB และอัลกอริทึม RIWE-NB ใช้เวลาดำเนินการใกล้เคียงกันในชุดข้อมูลทั้งหมด

บทที่ 7
บทสรุปและข้อเสนอแนะ

7.1 สรุป

อัลกอริทึมนาอีฟเบย์ (NB) เป็นอัลกอริทึมที่นิยมใช้กันมากในการจำแนกประเภทอีเมลสแปม (Spam email) และการจำแนกประเภทข้อความ จุดแข็งของอัลกอริทึม NB คือมีการฝึกฝนอย่างรวดเร็วโดยใช้เทคนิคอย่างง่ายและยังให้ความแม่นยำสูง อีกทั้งยังใช้ชุดข้อมูลสำหรับฝึกฝนขนาดเล็กและสามารถจัดการกับปัญหาการจำแนกประเภทแบบหลายคลาสได้ จุดอ่อนของอัลกอริทึม NB คือ ปัญหาความน่าจะเป็นมีค่าเป็นศูนย์ เกิดขึ้นในกรณีที่อีเมลที่ต้องการจำแนกมีคำบางคำที่ไม่ได้อยู่ในชุดข้อมูลสำหรับฝึกฝน จึงทำให้ความน่าจะเป็นมีค่าเป็นศูนย์และไม่สามารถจำแนกได้ ส่งผลให้ความแม่นยำลดลง โดยทั่วไปปัญหานี้สามารถแก้ได้ด้วยวิธีการทำลาปลาซสมูททิง (Laplace smoothing) อัลกอริทึม AWF-NB [8] หรืองานวิจัยที่ต้องการปรับปรุง เสนอวิธีการปรับปรุงอัลกอริทึม NB แบบดั้งเดิม โดยการลดความสำคัญของคำในคลาสที่มีความถี่ต่ำกว่า ด้วยแนวคิดที่ว่าอัลกอริทึม NB จะมองว่าคำทุกคำมีความสำคัญเท่ากัน ซึ่งในความเป็นจริงไม่ได้เป็นแบบนั้นทุกกรณี อย่างไรก็ตามการกระทำนี้ส่งผลให้ความแม่นยำลดลงในกรณีที่มีคำที่มีความสำคัญในแต่ละคลาสต่างกันเพียงเล็กน้อย นี้คือข้อเสียที่ร้ายแรงของอัลกอริทึม AWF-NB ยิ่งชุดข้อมูลสำหรับทดสอบมีคำที่มีความสำคัญระหว่างคลาสและคำที่มีบทบาทสูงใกล้เคียงกันหลายคำ ความแม่นยำของอัลกอริทึม AWF-NB จะยิ่งลดลง ดังนั้นวิทยานิพนธ์นี้จึงเสนอวิธีการปรับปรุงอัลกอริทึม AWF-NB เพื่อจัดการกับข้อเสียที่ร้ายแรงนี้โดยเรียกอัลกอริทึมที่ปรับปรุงใหม่นี้ว่าอัลกอริทึม RIWE-NB วิธีการแก้ปัญหาของอัลกอริทึม RIWE-NB คือจะใช้การคำนวณค่าเอนโทรปีของคำ เพื่อเป็นเกณฑ์ในการตัดสินใจว่าควรจะลดความสำคัญของคำในคลาสที่มีความสำคัญน้อยกว่าหรือไม่ ยิ่งค่าเอนโทรปีมีค่ามาก หมายถึงความสำคัญของคำในแต่คลาสมีค่าใกล้เคียงกันมาก ดังนั้นจึงมีโอกาสน้อยที่คำนั้นจะถูกลดความสำคัญในคลาสที่มีความสำคัญน้อยกว่า

ผลการทดลองแสดงให้เห็นว่าอัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกที่มากกว่าอัลกอริทึม AWF-NB และอัลกอริทึม NB ในชุดข้อมูลส่วนมากอย่างเห็นได้ชัด มีเพียงชุดข้อมูลเดียวเท่านั้นที่อัลกอริทึม RIWE-NB มีค่าน้อยกว่าอัลกอริทึม NB เล็กน้อย เนื่องจากชุดข้อมูลนั้นมีจำนวนคำที่มีความสำคัญระหว่างคลาสใกล้เคียงกันเป็นจำนวนมาก และคำเหล่านั้นมีบทบาทในการจำแนกสูง ซึ่งอัลกอริทึม RIWE-NB จะลดความสำคัญด้วยวิธีการสุ่มตัวเลขระหว่าง 0 – 1 ถ้าตัวเลขที่ได้มีค่ามากกว่าค่าเอนโทรปีจะทำการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่า ต่อให้เอนโทรปีจะมีค่ามากแค่ไหน แต่เนื่องจากใช้วิธีการสุ่ม ดังนั้นจึงมีโอกาสที่ความสำคัญของคำในคลาสที่ต่ำกว่าจะถูกลด ในขณะที่ค่าเอนโทรปีมีค่ามาก ถ้าเปรียบเทียบอัลกอริทึม RIWE-NB กับอัลกอริทึม AWF-NB

ผลที่ได้คืออัลกอริทึม RIWE-NB มีความแม่นยำมากกว่าอัลกอริทึม AWF-NB ในชุดข้อมูลทั้งหมด เมื่อพิจารณาเวลาดำเนินการ อัลกอริทึมที่ใช้เวลาดำเนินการมากที่สุดคืออัลกอริทึม RIWE-NB, อัลกอริทึม AWF-NB และอัลกอริทึม NB ตามลำดับ อย่างไรก็ตามจากผลการทดลองตัวเลขเวลาต่างกันไม่เกิน 4 วินาทีเท่านั้น ในความเป็นจริงถือว่าแทบจะไม่แตกต่าง ดังนั้นสรุปได้ว่าแม้อัลกอริทึม RIWE-NB จะใช้เวลาดำการมากที่สุด แต่ก็มากขึ้นเพียงเล็กน้อย ซึ่งคุ้มค่าที่อัลกอริทึม RIWE-NB จะใช้เวลาดำเนินการเพิ่มขึ้นเล็กน้อย เพื่อให้ได้ความแม่นยำในการจำแนกมากขึ้นอย่างเห็นได้ชัด ดังนั้นอัลกอริทึม RIWE-NB มีความแม่นยำในการจำแนกมากกว่าอัลกอริทึม AWF-NB และอัลกอริทึม NB แบบดั้งเดิมอย่างเห็นได้ชัด ในขณะที่เวลาดำเนินการยังคงถูกรักษาไว้ในระดับที่ใกล้เคียงกัน

7.2 ข้อเสนอแนะ

อัลกอริทึม RIWE-NB ที่เสนอสามารถนำไปปรับปรุงพัฒนาต่อได้ โดยวิธีการกำหนดเทรชโฮลด์ (Threshold) ตามที่ได้อธิบายไว้อัลกอริทึม RIWE-NB จะใช้วิธีการสุ่มตัวเลขระหว่าง 0 – 1 ถ้าตัวเลขที่ได้มีค่ามากกว่าค่าเอนโทรปีของคำ ถึงจะทำการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่า อย่างไรก็ตามเนื่องจากมันคือวิธีการสุ่ม ต่อให้เอนโทรปีมีค่ามากแค่ไหน ถึงจะมีโอกาสน้อยที่จะเกิดการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่า แต่มันก็ยังสามารถเกิดกรณีนี้ขึ้นได้ ยิ่งในชุดข้อมูลที่มีจำนวนคำที่มีความสำคัญระหว่างคลาสใกล้เคียงกันเป็นจำนวนมากและคำเหล่านั้นยังมีบทบาทในการจำแนกสูง ความแม่นยำของอัลกอริทึม RIWE-NB จึงมีโอกาสที่จะลดลงได้ ซึ่งการเพิ่มเทรชโฮลด์เป็นการทำให้แน่ใจว่าถ้าค่าเอนโทรปีมีค่าที่มากกว่าเทรชโฮลด์ที่กำหนดไว้ จะไม่ทำการลดความสำคัญของคำในคลาสที่มีความสำคัญต่ำกว่าอย่างแน่นอน วิธีการนี้จะเป็นการเพิ่มความแม่นยำในการจำแนกปกติของอัลกอริทึม RIWE-NB ด้วย อย่างไรก็ตามต้องมีการทดลองเพื่อหาค่าเทรชโฮลด์ที่เหมาะสม หรือค่ากลางที่ดีที่สุดที่สามารถใช้ได้กับทุกชุดข้อมูล

เอกสารอ้างอิง

[1] W. Peng, L. Huang, J. Jia and E. Ingram. “Enhancing the Naive Bayes Spam Filter Through Intelligent Text Modification Detection” 17th IEEE Int. Conf. Trust, Security And Privacy In Computing And Communications / 12th IEEE Int. Conf. Big Data Science And Engineering (TrustCom/BigDataSE), New York, USA, 2018. pp. 849-854.

[2] K. Tretyakov. “Machine Learning Techniques in Spam Filtering” Data Mining Problem-oriented Seminar, 2004, pp. 60-79.

[3] J. Brownlee. “A Gentle Introduction to Bayes Theorem for Machine Learning.” [online]. Available: https://machinelearningmastery.com/bayes-theorem-for-machine-learning. 2019.

[4] C. Goyal. “Improve Naive Bayes Text classifier using Laplace Smoothing.” [online]. Available: https://www.analyticsvidhya.com/blog/2021/04/improve-naive-bayes-text-classifier-using-laplace-smoothing. 2021.

[5] S.K. Tuteja and N. Bogiri. “Email Spam filtering using BPNN classification algorithm” Int. Conf. Automatic Control and Dynamic Optimization Techniques (ICACDOT), Pune, India, 2016. pp. 915-919.

[6] V. Vishagini and A.K. Rajan. “An Improved Spam Detection Method with Weighted Support Vector Machine” Int. Conf. Data Science and Engineering (ICDSE), Kochi, India, 2018.

[7] K. Agarwal and T. Kumar. “Email Spam Detection Using Integrated Approach of Naive Bayes and Particle Swarm Optimization” 2nd Int. Conf. Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018. pp. 685-690.

[8] Z. Guo. “Text Classification Based on Naive Bayes with Adjusted Weights via Frequency Ratio of Feature Words” Int. Conf. Computer Technology and Media Convergence Design (CTMCD), Sanya, China, 2021. pp. 263-267.

[9] wikipedia. “Entropy (information theory).” [online]. Available: https://en.wikipedia.org/wiki/Entropy_(informatio n_theory).

[10] S. Ray. “6 Easy Steps to Learn Naive Bayes Algorithms with codes in Python and R.” [online]. Available:

https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained. 2017.

[11] R. Tipsena, C. Jareanpon and G. Somprasertsri. “Automatic Question Classification on Webboard Using Text Mining Techniques” Journal of Science and Technology Mahasarakham University, vol. 33, no. 5, 2013. pp. 493-502.

[12] U. Suttapakti. Text Classification Efficiency of Open Ended Questionnaire by Naïve-Bayes and Support Vector Machine. N.P. : Rajapruk University. 2011.

[13] V. Garnepudi. “Spam Mails Dataset.” [online]. Available: https://www.kaggle.com/datasets/venky73/spam-mails-dataset. 2019.

[14] F. Qureshi. “Spam Email.” [online]. Available: https://www.kaggle.com/datasets/mfaisalqureshi/spam-email. 2021.

[15] Nitisha. “Email Spam Dataset.” [online]. Available: https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset. 2020.

[16] H. Sinha. “Email Spam Classification.” [online]. Available: https://www.kaggle.com/datasets/harshsinha1234/email-spam-classification. 2020.

[17] H. Ozler. “Spam or Not Spam Dataset.” [online]. Available: https://www.kaggle.com/datasets/ozlerhakan/spam-or-not-spam-dataset. 2018.

[18] J. Aguilarand and K. Cureno “Spam email from Enron Dataset.” [online]. Available: https://www.kaggle.com/datasets/juanagsolano/spam-email-from-enron-dataset. 2021.

[19] C. Naidu. “Spam Classification for Basic NLP.” [online]. Available: https://www.kaggle.com/datasets/chandramoulinaidu/spam-classification-for-basic-nlp. 2021.

[20] G. Olalekan. “Email Classification.” [online]. Available: https://www.kaggle.com/datasets/ganiyuolalekan/spam-assassin-email-classification-dataset. 2021.

[21] V. Ch 22384. “spam and ham email dataset.” [online]. Available: https://www.kaggle.com/datasets/venkateshch22384/spam-and-ham-email-dataset. 2022.

[22] T. Anderson. “Ham and Spam Emails.” [online]. Available: https://www.kaggle.com/datasets/tobyanderson/ham-and-spam-emails. 2021.

[23] W. van Lit. “Email Spam.” [online]. Available: https://www.kaggle.com/datasets/veleon/ham-and-spam-dataset. 2019.

[24] M. Gu “Ling-Spam Dataset.” [online]. Available: https://www.kaggle.com/datasets/mandygu/lingspam-dataset. 2019.

[25] N. David “Email Spam Classification from SHANTANU DHAKAD.” [online]. Available: https://www.kaggle.com/datasets/neildavid/email-spam-classification-from-shantanu-dhakad. 2022.

ภาคผนวก ก.
งานวิจัยที่ได้รับการตีพิมพ์

ก.1 Improving Naive Bayes by Reducing the Importance of Low-Frequency Words Based on Entropy of Words for Spam Email Classification, ICCAS 2022

2022 The 22nd International Conference on Control, Automation and Systems (ICCAS 2022)
BEXCO, Busan, Korea, Nov. 27~Dec. 01, 2022

Improving Naive Bayes by Reducing the Importance of Low-Frequency Words Based on Entropy of Words for Spam Email Classification

Phaiboon Trikanjananun[1], Arjin Numsomran[1*], and Vittaya Tipsuwannaporn[1]

[1]Department of Instrumentation and Control Engineering, School of Engineering,
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand, (arjin.nu@kmitl.ac.th) * Corresponding author

Abstract: The Naive Bayes algorithm (NB algorithm) is a popular one for spam email classification due to fast training, using simple techniques and high accuracy. One of many research improving NB algorithms are the AWF-NB algorithm. In this paper, we call the research an AWF-algorithm for convenient mention. The AWF-NB algorithm focuses on solving the equally important word in each class because it is not always the case. Another problem of the NB algorithm to solve this problem, the AWF-NB extremely reduces the importance of words in the class that has lower importance. However, this action will lead to reducing the accuracy in cases that slightly differ among the importance of words in each class. Therefore, the goal of the research is to improve the AWF-NB algorithm by reducing the importance of words based on entropy of words. We compute the entropy of a word to decide if it should be reduced in importance. The experimental results on ten spam email datasets from Kaggle website indicated that the RIWE-NB algorithm can remarkably increase the classification accuracy of the NB algorithm and the AWF-NB algorithm in majority datasets while the execution time is still conserved.

Keywords: Naïve bayes, NB algorithm, Spam email classification

1. INTRODUCTION

The NB algorithm is widely used in classification tasks because of fast training, use of simple techniques, and high accuracy. This is a prominent one for spam email classification or text classification. In addition, the NB algorithm uses a small training set and can deal with multi-class prediction problems [1], [2]. However, in the Naive Bayes, there is a serious problem since it is derived from the Bayes theorem [3], which is a probability calculation. If it classifies an email containing the word that does not appear in the training set, the probability will be zero. As a result, the classification accuracy is reduced. This problem can be solved by Laplace smoothing [4], commonly used for the Naive Bayes algorithm. We picked up the NB algorithm to improve spam email classification because of its advantage as we mentioned before.

There are many researchers who have applied various algorithms for spam email classification. Tuteja and Bogiri [5] use BPNN algorithm (Back Propagation Neural Network) to classify spam email. Although the accuracy is so high, the BPNN rather consumes more training time and is complex to develop. Vishagini and Rajan [6] developed a weighted SVM (Support Vector Machine) using their weight from the KFCM algorithm. Their proposed method can reduce more misclassification rate than the conventional SVM and the weighted SVM. However, it is more complex and consumes time than the NB algorithm. Agarwal and Kumar [7] combined PSO (Particle Swarm Optimization) algorithm with NB algorithm to classify spam email. Their experimental results show that their algorithm has more accuracy than the conventional NB algorithm. The same with the algorithms mentioned above. Combining PSO with NB consumes a lot of training and testing time. It was a very complex model and difficult to modify.

The next research is about improving the NB algorithm for text classification. Guo [8] applied adjusting weights via frequentness in ratio of feature words with Naive Bayes algorithm, solving the equally important word in each class, because it is not always the case in reality [8]. In this paper, we will call the AWF-NB algorithm for convenient mention. The experimental results show that AWF-NB algorithm has more accuracy than the conventional NB algorithm in majority datasets. It can also be seen that it was a non-complex algorithm because it just uses adjusting weights with the NB algorithm. The AWF-NB also has a fast training and testing like the conventional NB. However, extremely reducing the importance of words in the class that has lower importance results in a drop in accuracy since the importance of words may be slightly different. Therefore, the AWF-NB algorithm is preferred to improve spam email classification. The goal of the research is to improve the AWF-NB algorithm by reducing the importance of words based on entropy of words. We add a new factor, the entropy of words, for deciding to adjust weight in AWF-NB algorithm. The proposed method can increase the classification accuracy in most datasets from the conventional NB and the AWF-NB while the execution time is conserved.

The rest of this paper is organized as the following sections. Section I defines the concept of the conventional NB algorithm. Section II explains the idea of AWF-NB algorithm. Section III illustrates the proposed method. In section IV, we explain the analysis of the experimental results. Lastly, section V concludes the performance of the recommended method.

978-89-93215-25-0/22/$31.00 ©ICROS

2. THE CONVENTIONAL NB ALGORITHM

The conventional NB algorithm is a supervised machine learning probabilistic model for classification based on Bayes Theorem with a hypothesis of independence among predictors [9]. This is a prominent one for spam email classification because of fast training, use of simple techniques and high accuracy. In addition, the conventional NB algorithm uses a small training set and can deal with multi-class prediction problems. The steps of the NB algorithm to classify spam email are described below.

Step 1: Calculate both probability of spam and non-spam (ham) email using Eq. (1), (2), and (3).

$$P(C|X) = P(C)\prod_{x\in X} P(x|C) \quad (1)$$

Where

P(C|X) is the probability of target class *C* (spam or ham) given every word (*X*) in email,

X is the set of attribute words,

P(C) is the probability of class *C*,

P(x|C) is the probability of attribute word *x* given class *C*.

$$P(C) = \frac{w_C}{W} \quad (2)$$

Where

w_C is the number of words in class *C*,

W is the number of every word.

$$P(x|C) = \frac{w_{x,C}}{w_C} \quad (3)$$

Where

$w_{x,C}$ is the number of words *x* in class *C*.

Step 2: Compare the probability of spam *P(S|X)* and probability of ham *P(H|X)*. If *P(S|X)* is greater than *P(H|X)*, classify this email as spam.

The serious problem of the NB algorithm, if the testing set of email contains the word that does not appear in the training set, the probability will be zero. Normally, Laplace smoothing is used to solve this problem. Eq. (4) shows representation *P(x|C)* with the Laplace smoothing method.

$$P(x|C) = \frac{W_{x,C} + \propto}{W_C + \propto * A} \quad (4)$$

Where

A is the number of attributes or unique words,

α is the smoothing parameter equal to 1.

3. THE AWF-NB ALGORITHM

The AWF-NB algorithm [8] applied adjusting weights via frequentness in ratio of feature words with the NB algorithm. This algorithm tries to solve the equally important word with each class of NB algorithm which is not always the case in reality. The AWF-NB algorithm steps are explained below.

Step 1: Calculate the probability of word *x* in spam email *P(x|S)* and the probability of word *x* in ham email *P(x|H)* using Eq. (3).

Step 2: If *P(x|S)* is greater than *P(x|H)*, represent *P(x|H)* with Eq. (5) and *P(x|S)* remains unchanged. It shows that the attribute word *x* in class *S* is more important than the word *x* in class *H*. Therefore, *P(x|H)* should be reduced in importance. This is the main concept of the AWF-NB algorithm.

$$P(x|H) = \frac{1}{w_C} \quad (5)$$

Where

P(x|H) is the probability of attribute word *x* given class *H*.

Step 3: Repeat step 1 - 2 to find the new *P(x|S)* and *P(x|H)* of every word.

Step 4: Use the new *P(x|S)* and *P(x|H)* in each word to calculate *P(S|X)* and *P(H|X)* using Eq. (1). If *P(S|X)* is greater than *P(H|X)*, classify this email as spam. If not, it's a ham email.

4. THE PROPOSED METHOD

As we explain the concept of AWF-NB algorithm in the previous section, The AWF-NB algorithm will extremely reduce the importance of words if it has the importance of a class less than another class. For example, assume that the number of words x equal to 100. The number of words x in spam email and ham email are 51 and 49, respectively. Therefore, P(x|S) is equal to 0.51, P(x|H) is equal to 0.49. In this case, the AWF-NB algorithm will extremely reduce P(x|H) to 0.01 using Eq (5). It may result in dropping the accuracy because the importance of word x in ham email is slightly less than the importance of word x in spam email. Hence, the proposed method will compute the Entropy of the word to decide if it should be reduced in importance. We will name the proposed method as the RIWE-NB algorithm. The Entropy value is explained below.

4.1 Entropy

The entropy [10] is the average level of uncertainty of attribute's data inherent to the possible class in information theory. The maximum entropy value is $\log_2(C)$ where *C* is the number of possible classes. Eq. (6) shows the entropy equation of a word *x* in the spam email and ham email class.

$$En(x) = -\sum_{c\in C} P(x|c)\log_2 P(x|c) \quad (6)$$

Where

En(x) is the entropy of word *x*,

C is the set of classes (spam or ham),

P(x|C) is the ratio of the number of word *x* in class *C* to the number of word x.

4.2 The step of the proposed algorithm

The proposed algorithm reforms the probability in Eq (1) by taking log, because the probability can be very small if the number of words *x* is very large. Eq (7) shows the changed probability of Eq (1).

$$P(C|X) = \log P(C) + \sum_{x \in X} \log P(x|C) \quad (7)$$

Step 1: Calculate the probability of word *x* in spam email *P(x|S)* and the probability of word *x* in ham email *P(x|H)* using Eq. (4). To deal with the zero probability problem, we need to do the Laplace smoothing. Therefore, we use Eq. (4) instead of Eq. (3).

Step 2: Calculate the Entropy of words *x* using Eq. (6).

Step 3: If *P(x|S)* is not equal to *P(x|H)*, check which one is less. Assume that *P(x|H)* is a smaller one.

Step 4: Random number *r* between 0 to 1 (maximum entropy value). If *r* is greater than the Entropy of words *x* in step 2 and *P(x|H)* is not equal to 0, represent *P(x|H)* with Eq. (8) and *P(x|S)* remains unchanged. Eq. (8) is the combination of Eq. (5) and the Laplace smoothing method.

$$P(x|C) = \frac{1 + \propto}{W_C + \propto * A} \quad (8)$$

In contrast, If *P(x|S)* is less than *P(x|H)*, *P(x|S)* is checked instead of *P(x|H)* if it should be decreasing. Reducing the importance of words is based on the entropy of words. The larger the entropy of a word, it means that there is more distribution of a word into each class. Therefore, the chance of reducing the importance of words is lower.

Step 5: Repeat step 1 - 4 to calculate the *P(x|S)* and *P(x|H)* of every word.

Step 6: Use the calculated *P(x|S)* and *P(x|H)* in each word to calculate *P(S|X)* and *P(H|X)* using Eq. (7). If *P(S|X)* is greater than *P(H|X)*, classify this email as spam. If not, it's a ham email. The pseudocode of the proposed algorithm is shown in Fig. 1.

5. EXPERIMENTAL CONDITIONS AND RESULTS

5.1 Datasets

We used ten datasets from the Kaggle website for testing. For convenient explanation, we will rename these datasets as spam email dataset 1 to 10 since many datasets don't have the unique name. In the references part, we will show the source of the URL of ten datasets [11], [12], [13], [14], [15], [16], [17], [18]. Table I shows the reference number of ten datasets and Table II shows characteristics of ten datasets.

```
function RIWE (email) returns classification
    words ← List of word in email
    pa_spam ← 0
    pa_ham ← 0
    for each value w of words do
        pw_spam ← Compute the probability of word w in spam email
        pw_ham ← Compute the probability of word w in ham email
        entropy ← Compute the entropy of word w
        r ← Random number between 0 to 1
        if pw_ham is less than pw_spam and pw_ham isn't equal to 0
            if r is greater than entropy,
                then reducing the importance of pw_ham
        else if pw_spam is less than pw_ham and pw_spam isn't equal to 0
            if r is greater than entropy,
                then reducing the importance of pw_spam
        pa_spam ← Accumulate pw_spam value
        pa_ham ← Accumulate pw_ham value
    PS ← Compute the probability of spam using pa_spam
    PH ← Compute the probability of ham using pa_ham
    if PS is greater than PH,
        then return spam class
    else
        return ham class
```

Fig. 1. The pseudocode of the RIWE-NB algorithm.

Table 1. Reference number of ten datasets

Datasets	Table Column Head
Spam email dataset 1	11
Spam email dataset 2	12
Spam email dataset 3, 4, 5	13
Spam email dataset 6	14
Spam email dataset 7	15
Spam email dataset 8	16
Spam email dataset 9	17
Spam email dataset 10	18

Table 2. Characteristics of ten datasets

Datasets	No. of emails	No. of spam emails	No. of ham emails
Spam email dataset 1	5171	1499	3672
Spam email dataset 2	5572	747	4825
Spam email dataset 3	6046	1896	4150
Spam email dataset 4	10000	5000	5000
Spam email dataset 5	2605	433	2172
Spam email dataset 6	5727	1368	4359
Spam email dataset 7	3000	500	2500
Spam email dataset 8	5854	1496	4358
Spam email dataset 9	5796	1896	3900
Spam email dataset 10	5796	1896	3900

5.2 Experimental conditions and results

Every dataset in Table II is randomly split into two parts for the training set and the test set by saving the balance of emails between both classes in each part. To receive the consistent results, we only proceeded with the experiment one thousand times with the same training and test sets for the RIWE-NB algorithm since the RIWE-NB has a random method. Other algorithms will just proceed one time because it's the same no matter how many times we test them. All of the algorithms proceeded with the Laplace smoothing method and didn't proceed extracting feature words before. Table III shows the experimental results of the conventional NB, AWF-NB and RIWE-NB.

Table 3. The experimental comparisons on the accuracy of the concertional NB, AWF-NB and RIWE-NB

Datasets	Accuracy Measurement		
	NB	*AWF-NB*	*RIWE-NB*
Spam email dataset 1	87.16	80.63	**94.57**
Spam email dataset 2	**96.95**	94.90	96.25
Spam email dataset 3	82.47	78.96	**92.60**
Spam email dataset 4	77.86	69.54	**95.48**
Spam email dataset 5	89.18	85.19	**98.43**
Spam email dataset 6	91.13	88.51	**94.12**
Spam email dataset 7	89.87	86.80	**93.21**
Spam email dataset 8	89.75	87.56	**95.29**
Spam email dataset 9	76.12	70.12	**96.79**
Spam email dataset 10	69.25	67.32	**95.37**

The experimental results in Table III illustrates that the classification accuracy of the RIWE-NB algorithm is more than the conventional NB algorithm and the AWF-NB algorithm in majority datasets. There is only spam email dataset 2 that has slightly lower accuracy. In contrast between the conventional NB and the AWF-NB, clearly, the AWF-NB has lower accuracy than the conventional NB in spam email classification without proceeding to extract feature words. It results from extremely reducing the important words within the class that have slightly lower importance. Therefore, the RIWE-NB algorithm can significantly increase the classification accuracy of the NB algorithm and the AWF-NB algorithm. TABLE IV shows the comparisons on the execution time of the conventional NB, AWF-NB and RIWE-NB.

The comparison results in Table IV illustrate that the algorithms consuming the most of execution time are RIWE-NB, AWF-NB and NB, respectively. The AWF-NB adds weighting to the conventional NB and the RIWE-NB improves the AWF-NB by adding entropy calculation so the experimental results are reasonable. In reality, we considered it no different because the difference in execution time is only less than a second. Therefore, the execution time is still conserved in the REWI-NB algorithm.

Table 4. The experimental comparisons on the execution time of the concertional NB, AWF-NB and RIWE-NB

Datasets	Execution Time Measurement (s)		
	NB	*AWF-NB*	*RIWE-NB*
Spam email dataset 1	**0.1605**	0.2483	0.3549
Spam email dataset 2	**0.0310**	0.0456	0.0657
Spam email dataset 3	**0.4209**	0.5804	0.7613
Spam email dataset 4	**0.4683**	0.7199	1.0013
Spam email dataset 5	**0.6901**	0.8288	0.9741
Spam email dataset 6	**0.2418**	0.3805	0.5417
Spam email dataset 7	**0.1356**	0.2093	0.2945
Spam email dataset 8	**0.2659**	0.4123	0.5797
Spam email dataset 9	**0.4266**	0.6055	0.7982
Spam email dataset 10	**0.4641**	0.7134	0.9904

6. CONCLUSION

This paper proposes a new approach to improve accuracy of the AWF-NB algorithm, called RIWE-NB algorithm, applying to spam email classification. To solve the extremely reducing importance of words within class that have lower importance. The main concept is reducing the importance of words within class that have lower importance based on the entropy of words. The RIWE-NB algorithm is determined by using ten datasets from Kaggle website. The experimental results indicate that RIWE-NB algorithm can remarkably increase the classification accuracy of the NB algorithm and the AWF-NB algorithm in majority datasets while the execution time is conserved.

REFERENCES

[1] W. Peng, L. Huang, J. Jia and E. Ingram, "Enhancing the Naive Bayes Spam Filter Through Intelligent Text Modification Detection," in 2018 17th IEEE Int. Conf. Trust, Security And Privacy In Computing And Communications / 12th IEEE Int. Conf. Big Data Science And Engineering (TrustCom/BigDataSE), New York, USA, pp.849-854.

[2] K. Tretyakov, "Machine Learning Techniques in Spam Filtering," in 2004 Data Mining Problem-oriented Seminar, pp.60-79.

[3] J. Brownlee, "A Gentle Introduction to Bayes Theorem for Machine Learning." machinelearningmastery.com. https://machinelearningmastery.com/bayes-theorem-for-machine-learning (accessed Oct. 4, 2019).

[4] C. Goyal, "Improve Naive Bayes Text classifier using Laplace Smoothing." analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2021/04/improve-naive-bayes-text-classifier-using-laplace-smoothing (accessed Apr. 16, 2021).

[5] S.K. Tuteja and N. Bogiri, "Email Spam filtering

using BPNN classification algorithm," in 2016 Int. Conf. Automatic Control and Dynamic Optimization Techniques (ICACDOT), Pune, India, pp.915-919.
[6] V. Vishagini and A.K. Rajan, "An Improved Spam Detection Method with Weighted Support Vector Machine," in 2018 Int. Conf. Data Science and Engineering (ICDSE), Kochi, India.
[7] K. Agarwal and T. Kumar, "Email Spam Detection Using Integrated Approach of Naive Bayes and Particle Swarm Optimization," in 2018 2nd Int. Conf. Intelligent Computing and Control Systems (ICICCS), Madurai, India, pp.685-690.
[8] Z. Guo, "Text Classification Based on Naive Bayes with Adjusted Weights via Frequency Ratio of Feature Words," in 2021 Int. Conf. Computer Technology and Media Convergence Design (CTMCD), Sanya, China, pp.263-267.
[9] S. Ray, "6 Easy Steps to Learn Naive Bayes Algorithms with codes in Python and R." analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained (accessed Sep. 11, 2017).
[10] "Entropy (information theory)" en.wikipedia.org. https://en.wikipedia.org/wiki/Entropy_(information_theory)
[11] V. Garnepudi, "Spam Mails Dataset" kaggle.com. https://www.kaggle.com/datasets/venky73/spam-mails-dataset (accessed Jan. 23, 2019).
[12] F. Qureshi, "Spam Email" kaggle.com. https://www.kaggle.com/datasets/mfaisalqureshi/spam-email (accessed Jun. 21, 2021).
[13] Nitisha, "Email Spam Dataset" kaggle.com. https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset (accessed Oct. 30, 2020).
[14] H. Sinha, "Email Spam Classification" kaggle.com. https://www.kaggle.com/datasets/harshsinha1234/email-spam-classification (accessed Jul. 6, 2020).
[15] H. Ozler, "Spam or Not Spam Dataset" kaggle.com. https://www.kaggle.com/datasets/ozlerhakan/spam-or-not-spam-dataset (accessed Dec. 15, 2018).
[16] J. Aguilar and K. Cureno, "Spam email from Enron Dataset" kaggle.com. https://www.kaggle.com/datasets/juanagsolano/spam-email-from-enron-dataset (accessed Nov. 19, 2021).
[17] C. Naidu, "Spam Classification for Basic NLP" kaggle.com. https://www.kaggle.com/datasets/chandramoulinaidu/spam-classification-for-basic-nlp (accessed Mar. 9, 2021).
[18] G. Olalekan, "Email Classification" kaggle.com. https://www.kaggle.com/datasets/ganiyuolalekan/spam-assassin-email-classification-dataset (accessed Aug. 6, 2021).

ประวัติผู้เขียน

ชื่อ-นามสกุล	ไพบูลย์ ตรีกาญจนานันท์
วัน เดือน ปีเกิด	12 มิถุนายน 2523
ที่อยู่	1050 ถนนพัฒนาการ แขวงสวนหลวง เขตสวนหลวง กรุงเทพฯ 10250
ประวัติการศึกษา	วศ.บ. (เกียรตินิยมอันดับ 1) วิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยมหิดล, 2545