การจำแนกแนวโน้มของราคาหุ้นไทยรายวัน
โดยใช้การวิเคราะห์ความรู้สึกจากข่าว
CLASSIFICATION OF DAILY THAI STOCK PRICE TREND
USING NEWS SENTIMENT ANALYSIS

ณัฐิมา อภินาทเมธี

NUTTIMA APINARTMAYTEE

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูลและการวิเคราะห์
ศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าเจ้าลาดกระบัง
คณะวิทยาศาสตร์สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2567

KMITL-2024-SC-M-017-042

CLASSIFICATION OF DAILY THAI STOCK PRICE TREND USING NEWS SENTIMENT ANALYSIS

NUTTIMA APINARTMAYTEE

AN INDEPENDENT STUDY SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIRENMENT FOR THE DEGREE OF MASTER OF SCIENCE
IN DATA SCIENCE AND ANALYTICS
KMITL DIGITAL ANALYTICS AND INTELLIGENCE CENTER SCHOOL OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2024

KMITL-2024-SC-M-017-042

หัวข้อการค้นคว้าอิสระ	การจำแนกแนวโน้มของราคาหุ้นไทยรายวัน โดยใช้การวิเคราะห์ความรู้สึกจากข่าว
ชื่อนักศึกษา	นางสาวณัฐิมา อภินาทเมธี
รหัสประจำตัว	64605043
ปริญญา	วิทยาศาสตรมหาบัณฑิต (วิทยาการข้อมูลและการวิเคราะห์) ศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าลาดกระบัง
พ.ศ.	2567
อาจารย์ที่ปรึกษาการค้นคว้าอิสระ	ผู้ช่วยศาสตราจารย์ ดร.กนกกรรณ์ ลี้โรจนาประภา

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์ในสร้างโมเดลการจำแนกการความรู้สึกจากหัวข้อข่าวหุ้นไทยรายวัน โดยทำการเปรียบเทียบการรูปแบบการเก็บ 3 แบบ (A - C) และการเตรียมข้อความข่าวด้วยตัวตัดคำภาษาไทย 2 แบบ คือ PyThaiNLP และ DeepCut (I - II) ทำให้สามารถแบ่งการทดลองเป็น 6 การทดลอง ก่อนเลือกรูปแบบการเตรียมข้อความที่เหมาะสม เพื่อนำมาสร้างตัวแบบการจำแนกทั้ง 10 อัลกอลิทึม ในการศึกษาครั้งนี้เลือกหุ้น 2 ตัวจากตัวอย่างที่สุ่มจากหุ้นในกลุ่ม SET50 ซึ่งได้แก่ หุ้นของบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) และบริษัท ราช กรุ๊ป จำกัด (มหาชน) เก็บรวบรวมข้อมูลทั้งหัวข้อข่าวและราคาหุ้นรายวัน ตั้งแต่เดือนมกราคม พ.ศ. 2561 ถึงเดือนธันวาคม พ.ศ. 2565 นำหัวข้อข่าวที่รวบรวมมาเข้ากระบวนการเตรียมข้อมูลด้วยการลบเครื่องหมายวรรคตอน สัญลักษณ์ และตัวเลข การนอร์มอลไลซ์ตัวอักษร การตัดคำ และการสกัดคุณลักษณะ สำหรับการกำหนดแนวโน้มราคาหุ้นเป็นบวก เป็นลบ และเป็นกลางจากผลต่างของราคาปิดรายวัน

ผลการวิจัยในขั้นตอนการเตรียมข้อความพบว่า การเก็บข้อความเฉพาะหัวข้อข่าวของหุ้นที่สนใจ (A) และเลือกการใช้ตัวตัดคำ PyThaiNLP (I) ให้ค่าประสิทธิภาพการจำแนกในโมเดลตามพารามิเตอร์พื้นฐานดีที่สุดจึงเลือกการเตรียมข้อความที่ให้ผลดีที่สุดมาแบ่งเป็นชุดข้อมูลเรียนรู้ (ร้อยละ 70) เพื่อการปรับจูนพารามิเตอร์ โดยใช้ GridSearchCV และชุดข้อมูลทดสอบ (ร้อยละ 30) เพื่อเปรียบเทียบประสิทธิภาพการจำแนก พบว่าชุดสำหรับหุ้น บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) แบบจำลองซัพพอร์ตเวกเตอร์แมชชีนมีค่าประสิทธิภาพสูงสุด โดยมีค่าความถูกต้องร้อยละ 62.50 สำหรับหุ้น บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) แบบจำลองป่าสุ่ม มีค่าประสิทธิภาพสูงสุดโดยมีค่าความถูกต้องร้อยละ 65

คำสำคัญ : การวิเคราะห์ความรู้สึก การจำแนกข่าว ควอลิตี้เฮ้าส์ ราช กรุ๊ป การตัดคำ

Independent Study Title	Classification of Daily Thai Stock Price Trend Using News Sentiment Analysis
Student Name	Nuttima Apinartmaytee
Student ID	64605043
Degree	Master of Science (Data Science and Analytics)
	KMITL Digital Analytics and Intelligence Center
Year	2024
Independent Study Advisor	Assist. Prof. Dr. Kanogkan Leerojanaprapa

Abstract

This research aims to create a sentiment classification model from daily Thai stock news headlines. By comparing 3 scatching formats (A - C) and 2 formats of Thai word tokens, PyThaiNLP and DeepCut (I - II), there are 6 experiments. The best text preparation format was selected to build a classification model. A performance comparison will identify the best algorithm for each stock. Two stocks from the SET50, ware randomly selected Quality Houses Public Company Limited (QH) and RATCH Group Public Company Limited (RATCH), Daily headline news and stock prices during January 2018 to December 2022 ware collected. Headline news was prepared by deleting punctuation marks, symbols and numbers, normalizing them, tokenization, and feature extractive. Stock price trends are classified as positive, negative or neutral based on the difference in daily closing prices.

The research results during the text preparation phase indicate that collecting headlines specifically related to the stocks of interest (A) and using the PyThaiNLP tokenizer (I) yielded the highest classification performance based on the default parameters of the model. Therefore, the text preparation method that produced the best results was selected to be divided into a training dataset (70%) for parameter tuning using GridSearchCV, and a test dataset (30%) for evaluating classification performance. For the stock of RATCH, the Support Vector Machine (SVM) model achieved the highest performance, with an accuracy of 62.50%. For the stock of QH, the Random Forest model achieved the highest performance, with an accuracy of 65%.

Keywords : Sentiment Analysis, News Classification, Quality House (QH), RATCH Group (RATCH), Tokenization

กิตติกรรมประกาศ

งานวิจัยฉบับนี้สำเร็จลุล่วงด้วยดีเนื่องจากความกรุณาและความช่วยเหลืออย่างดียิ่งจาก ผศ. ดร.กนกกรรณ์ ลี้โรจนาประภา อาจารย์ที่ปรึกษาการค้นความอิสระ ที่ท่านได้เสียสละเวลาอันมีค่ายิ่งในการให้คำปรึกษาการดำเนินงานวิจัย ตลอดจนได้ช่วยตรวจสอบ แก้ไขปัญหา ข้อบกพร่องต่างๆ และเป็นกำลังใจในการทำงานหลายต่อหลายครั้งอันเป็นประโยชน์อย่างยิ่งในการจัดทำงานวิจัยนี้ ตั้งแต่เริ่มดำเนินการจนกระทั่งดำเนินการเสร็จสมบูรณ์ ผู้วิจัยขอกราบ ขอบพระคุณเป็น อย่างสูงไว้ ณ โอกาสนี้

ขอขอบคุณอาจารย์ ศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าลาดกระบังทุกท่าน ที่ให้ความรู้ในการทำงานวิจัยนี้ ตลอดจนเพื่อนๆ คณะวิทยาศาสตร์ สาขาวิทยาการข้อมูลและ การวิเคราะห์ที่ให้ความช่วยเหลือ ให้คำปรึกษาในการทำงานวิจัยครั้งนี้

ขอกราบขอบพระคุณคุณกุลธิดา กิติโกเศศ และครอบครัวที่คอยให้คำปรึกษาในเรื่องต่างๆ รวมทั้งเป็นกำลังใจที่ดีเสมอมา ในการทำงานวิจัยนี้ให้สมบูรณ์

สุดท้ายนี้ผู้วิจัยหวังว่างานวิจัยฉบับนี้คงเป็นประโยชน์สำหรับหน่วยงานที่เกี่ยวข้อง และผู้ที่สนใจศึกษาต่อไป

นางสาวณัฐิมา อภินาทเมธี

สารบัญ

บทที่ 1 บทนำ **1**

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของงานวิจัย 1

1.3 ขอบเขตของงานวิจัย 2

1.4 ประโยชน์ที่คาดว่าจะได้รับ 2

1.5 คำนิยาม 2

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง **4**

2.1 การเตรียมข้อมูลข้อความ 4

2.1.1 การตัดคำ (Tokenization) 4

2.1.2 การลบเครื่องหมายวรรคตอน (Punctuation Removal) 5

2.1.3 การนอร์มอลไลซ์ตัวอักษร (Character Normalization) 5

2.1.4 การลบคำฟุ่มเฟือย (Stop Word Removal) 6

2.1.5 การวิเคราะห์ความถี่ของคำในแต่ละกลุ่ม (Analysis Word Frequency in Groups) 6

2.1.6 การสกัดคุณลักษณะ (Feature Extraction) 7

2.2 การตัดคำด้วย PYTHAINLP และ DEEPCUT 8

2.2.1 PythaiNLP 8

2.2.2 DeepCut 8

2.3 การจำแนกความรู้สึก 9

2.4 การเรียนรู้แบบมีผู้สอนเพื่อจำแนกความรู้สึกของข่าวหุ้นรายวัน 9

2.4.1 การถดถอยเชิงโลจิสติค (Logistic Regression) 10

2.4.2 ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) 11

2.4.3 ป่าสุ่ม (Random Forest) 12

2.4.4 ต้นไม้ตัดสินใจ (Decision Tree) 13

2.4.5 นาอีฟเบย์ (Naïve Bayes) 14

2.4.6 เพอร์เซปตรอน (Perceptron) 15

2.4.7 เพื่อนบ้านใกล้สุด k ตัว (K-nearest Neighbor Algorithm) 15

2.4.8 สโตแคสติกการ์เดียนดิเซนท์ (Stochastic Gradient Descent) 16

2.5 การเปรียบเทียบประสิทธิภาพการทำนาย 16

2.5.1 เมทริกซ์ความสับสน (Confusion Matrix) 16

สารบัญ

หน้า

2.5.2 ค่าความถูกต้อง (Accuracy) 18
2.5.3 ค่าความแม่นยำ (Precision) 19
2.5.4 ค่าความระลึก (Recall) 19
2.5.5 ค่าความถ่วงดุล (F-Measure) 19
2.5.6 การเลือกโมเดลที่ดีที่สุด (Model Selection) 20
2.5 งานวิจัยที่เกี่ยวข้อง 20
บทที่ 3 วิธีดำเนินการวิจัย 22
3.1 การเก็บข้อมูลจากเว็บไซต์ 22
3.1.1 วิธีการเก็บข้อความหัวข้อข่าวเพื่อใช้ในการจำแนกความรู้สึก 23
3.1.2 วิธีการเก็บข้อมูลราคาปิดรายวันของหุ้นทั้ง 2 ตัว 24
3.1.3 การสร้างคำตอบเพื่อใช้ในการฝึกสอนแบบจำลองประเภทการจำแนก 25
3.2 การเตรียมข้อมูล 27
3.2.1 การลบเครื่องหมายวรรคตอนและสัญลักษณ์ออก (Punctuation Removal) 28
3.2.2 การนอล์มอลไลซ์ตัวอักษร (Character Normalization) 28
3.2.3 การตัดคำ (Tokenization) 29
3.2.4 การกำจัดคำหยุด (Stop Word Removal) 29
3.2.5 การวิเคราะห์คำในแต่ละกลุ่ม (Analysis Word in Group) 29
3.2.6 การสกัดคุณลักษณะ (Feature Extraction) 30
3.3 การจำแนกความรู้สึกจากข่าวหุ้นรายวัน 30
3.3.1 การถดถอยเชิงโลจิสติค (Logistic Regression) 30
3.3.2 นาอีฟเบย์ (Naïve Bayes) 31
3.3.3 การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) 31
3.3.4 ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) 31
3.3.5 ป่าสุ่ม (Random Forest) 32
3.3.6 ต้นไม้ตัดสินใจ (Decision Tree) 32
3.3.7 ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) 32
3.3.8 เพอร์เซปตรอน (Perceptron) 33
3.3.9 สโตแคสติกการ์เดียนดิเซนท์ (SGD) 33

สารบัญ

หน้า

3.3.10 พาสซีฟ อากัสซีฟ (Passive Aggressive) 33
3.3.11 สรุปผลการปรับค่าพารามิเตอร์ 34
3.4 การวัดประสิทธิภาพ 36
บทที่ 4 ผลการวิจัยและการอภิปรายผล 37
4.1 ลักษณะของข้อมูลที่ใช้ในการทดลอง 37
4.2 การเปรียบเทียบผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันตามตัวตัดคำ 46
4.3 ผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจากแบบจำลอง 53
4.3.1.ผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจากแบบจำลองชุดข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นและใช้ตัวตัดคำ PythaiNLP 54
4.3.2 ผลจากแบบจำลองชุดข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นด้วยตัวตัดคำ PythaiNLP 62
4.4 ผลการเปรียบเทียบประสิทธิภาพของแบบจำลอง 70
4.5 ผลการนำแบบจำลองไปใช้งาน 71
4.5.1 การตรวจสอบค่าความสำคัญของคุณลักษณะ (Feature Importances) 72
4.5.2 การตรวจสอบข้อความจากผู้ใช้ 72
4.6 อภิปรายผล 73
บทที่ 5 สรุปผลการวิจัยและข้อเสนอแนะ 74
5.1 สรุปผลการวิจัย 74
5.2 ข้อเสนอแนะ 75
เอกสารอ้างอิง 77
ภาคผนวก ก 80
ประวัติผู้เขียน 130

สารบัญตาราง

หน้า

ตารางที่

2.1 Confusion Matrix ของ Class A 17
2.2 Confusion Matrix ของ Class B 17
2.3 Confusion Matrix ของ Class C 18
3.1 รูปแบบที่ใช้ดึงข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) 24
3.2 รูปแบบที่ใช้ดึงข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) 24
3.3 ตัวอย่างข้อความที่รวบรวมจากเว็บข่าวหุ้น 24
3.4 ตัวอย่างข้อมูลสถานะการจำแนกจากราคาปิด 27
3.5 ตัวอย่างขั้นตอนการการลบเครื่องหมายวรรคตอนสัญลักษณ์และตัวเลขออก 28
3.6 ตัวอย่างการนอล์มอลไลซ์ตัวอักษร 29
3.7 ตัวอย่างการเปรียบเทียบผลของตัวตัดคำ 29
3.8 ตัวอย่างการกำจัดคำหยุด 29
3.9 ตัวอย่างตัวแทนเอกสารที่ได้จากการสกัดคุณลักษณะ 30
3.10 พารามิเตอร์ที่นำมาใช้ในการตั้งค่าสำหรับเพิ่มประสิทธิภาพของแบบจำลอง (RATCH) 34
3.11 ค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลอง (RATCH) 35
3.12 พารามิเตอร์ที่นำมาใช้ในการตั้งค่าสำหรับเพิ่มประสิทธิภาพของแบบจำลอง (QH) 35
3.13 ค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลอง (QH) 36
4.1 จำนวนข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) 38
4.2 จำนวนข้อมูลข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) 38
4.3 จำนวนข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) 39
4.4 จำนวนข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) 43
4.5 ผลจากตัวตัดคำ PythaiNLP บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ 46
4.6 ผลจากตัวตัดคำ DeepCut บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ 48
4.7 ผลจากตัวตัดคำ PythaiNLP บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ชุดข้อมูลทดสอบ 50
4.8 ผลจากตัวตัดคำ DeepCut บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ชุดข้อมูลทดสอบ 51
4.9 ผลการทดสอบแบบจำลองการถดถอยเชิงโลจิสติคด้วยข้อมูลทดสอบ 54

สารบัญตาราง

หน้า

ตารางที่

4.10 ผลการทดสอบแบบจำลองนาอีฟเบย์ด้วยข้อมูลทดสอบ 56
4.11 ผลการทดสอบแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด ด้วยข้อมูลทดสอบ 57
4.12 ผลการทดสอบแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบ 58
4.13 ผลการทดสอบแบบจำลองป่าสุ่ม ด้วยข้อมูลทดสอบ 58
4.14 ผลการทดสอบแบบจำลองต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบ 59
4.15 ผลการทดสอบแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลทดสอบ 60
4.16 ผลการทดสอบแบบจำลองเพอร์เซปตรอน ด้วยข้อมูลทดสอบ 61
4.17 ผลการทดสอบแบบจำลองการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบ 61
4.18 ผลการทดสอบแบบจำลองพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบ 62
4.19 ผลการทดสอบแบบจำลองการถดถอยเชิงโลจิสติคด้วยข้อมูลทดสอบ 63
4.20 ผลการทดสอบแบบจำลองนาอีฟเบย์ ด้วยข้อมูลทดสอบ 64
4.21 ผลการทดสอบแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด ด้วยข้อมูลทดสอบ 64
4.22 ผลการทดสอบแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบ 65
4.23 ผลการทดสอบแบบจำลองป่าสุ่ม ด้วยข้อมูลทดสอบ 66
4.24 ผลการทดสอบแบบจำลองต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบ 67
4.25 ผลการทดสอบแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลทดสอบ 67
4.26 ผลการทดสอบแบบจำลองเพอร์เซปตรอน ด้วยข้อมูลทดสอบ 68
4.27 ผลการทดสอบแบบจำลองการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบ 69
4.28 ผลการทดสอบแบบจำลองพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบ 70
4.29 ค่าประสิทธิภาพของแบบจำลองตัวตัดคำ PythaiNLP บริษัท ราช กรุ๊ป จำกัด (มหาชน) 71
4.30 ค่าประสิทธิภาพของแบบจำลองตัวตัดคำ PythaiNLP บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) 71

สารบัญรูป

หน้า

รูปที่

2.1 ตัวอย่างการใช้งานแสดงคำฟุ่มเฟือย 6
2.2 ตัวอย่างผลของคำฟุ่มเฟือย 6
2.3 ตัวอย่างการใช้งานการสร้างภาพแสดงความถี่ของคำที่พบ 6
2.4 ตัวอย่างภาพแสดงความถี่ของคำที่พบ 7
2.5 ตัวอย่างการใช้งานตัวตัดคำ PythaiNLP 8
2.6 ตัวอย่างผลของตัวตัดคำ PythaiNLP 8
2.7 ตัวอย่างการใช้งานตัวตัดคำ DeepCut 9
2.8 ตัวอย่างผลของตัวตัดคำ DeepCut 9
2.9 ตัวอย่างผล SVM Classification 11
2.10 การแบ่งข้อมูลด้วยวิธี SVM Classification ของข้อมูลที่มี 2 กลุ่ม 12
2.11 การทำงานของ Random Forest modeling 13
2.12 ส่วนประกอบของต้นไม้ตัดสินใจ (Decision Tree) 14
2.13 การทำงานของเพอร์เซปตรอน (Perceptron) 15
3.1 ขั้นตอนการดำเนินงานวิจัย 22
3.2 Google Colaboratory หรือ Google Colab 22
3.3 คำสั่งในการดึงข้อมูลจาก เว็บข่าวหุ้น ด้วย Beautiful Soup 23
3.4 คำสั่งในการดึงข้อมูลจาก Yahoo Finance 25
3.5 ตัวอย่างข้อมูลจาก Yahoo Finance 25
3.6 ตัวอย่างข้อมูลที่เชื่อมต่อกันแล้วผ่านวันที่ 25
3.7 ขั้นตอนการจำแนกความรู้สึกจากข่าว 26
3.8 ขั้นตอนการเตรียมข้อมูล 27
4.1 Word Cloud ของข้อความที่เป็น Positive (Pos) ของหุ้น RATCH 40
4.2 Word Cloud ของข้อความที่เป็น Negative (Neg) ของหุ้น RATCH 41

สารบัญรูป

หน้า

รูปที่

4.3 Word Cloud ของข้อความที่เป็น Neutral (Neu) ของหุ้น RATCH 42
4.4 Word Cloud ของข้อความที่เป็น Positive (Pos) ของหุ้น QH 43
4.5 Word Cloud ของข้อความที่เป็น Negative (Neg) ของหุ้น QH 44
4.6 Word Cloud ของข้อความที่เป็น Neutral (Neu) ของหุ้น QH 45
4.7 ค่าความสำคัญของคุณลักษณะจากแบบจำลอง 72
4.8 การตรวจสอบข้อความจากผู้ใช้ 72

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันการลงทุนในตลาดหลักทรัพย์แห่งประเทศไทยได้รับความนิยมอย่างมาก เนื่องจากเป็นช่องทางการลงทุนที่ให้ผลตอบแทนสูงกว่าการลงทุนในสินทรัพย์อื่นๆ เช่น ที่ดิน ทองคำแท่ง หรือการฝากเงินกับธนาคารพาณิชย์ (ท๊อปไลน์เนอร์, 2564) ซึ่งทำให้นักลงทุนทั้งรายย่อยและรายใหญ่หันมาสนใจการลงทุนในตลาดหลักทรัพย์เพื่อสร้างความมั่งคั่งให้กับตนเอง อย่างไรก็ตาม การลงทุนในตลาดหลักทรัพย์มีความอ่อนไหวต่อการเปลี่ยนแปลงจากปัจจัยภายนอก เช่น นโยบายรัฐบาล สภาวะเศรษฐกิจ และข่าวสารต่าง ๆ ที่ส่งผลกระทบต่อราคาหุ้นอย่างมีนัยสำคัญ

การศึกษาการลงทุนในตลาดหลักทรัพย์ไทยพบว่าเหตุการณ์ทางเศรษฐกิจและการเมืองที่สำคัญ เช่น เหตุการณ์ “Black Monday” ในปี พ.ศ. 2530 วิกฤตการณ์อ่าวเปอร์เซียในปี พ.ศ. 2533 และวิกฤตการณ์ทางการเงินในภูมิภาคเอเชียในปี พ.ศ. 2540 ส่งผลกระทบต่อความเชื่อมั่นของนักลงทุนอย่างมาก (ลงทุนศาสตร์, 2564) นอกจากนี้ ข่าวสารเกี่ยวกับโรคระบาด เช่น การระบาดของโรค SARS และโควิด19 ยังทำให้นักลงทุนเกิดความตื่นตระหนกและเทขายหลักทรัพย์ ส่งผลให้ดัชนีตลาดหลักทรัพย์ลดลงอย่างมาก (ฟีโนมีน่า, 2563)

จากความอ่อนไหวของตลาดหลักทรัพย์ที่ได้รับผลกระทบจากข่าวสารและเหตุการณ์ต่าง ๆ ทำให้การวิเคราะห์ความรู้สึกจากข่าวหุ้นกลายเป็นเครื่องมือที่มีประสิทธิภาพในการช่วยนักลงทุนในการตัดสินใจลงทุน การวิเคราะห์ความรู้สึกจากข่าวสามารถช่วยให้ผู้ลงทุนรับรู้ถึงความเชื่อมั่นและทัศนคติของตลาดต่อหุ้นแต่ละตัว ซึ่งอาจมีผลต่อการเปลี่ยนแปลงของราคาหุ้นและความผันผวนของตลาดได้ การวิเคราะห์นี้จึงมีความสำคัญในการช่วยนักลงทุนประเมินสถานการณ์และตัดสินใจได้อย่างมีข้อมูลรองรับ (การ์ตูน, 2566)

ในการศึกษาผลกระทบของข่าวสารต่อราคาหุ้นในครั้งนี้ ผู้วิจัยได้เลือกศึกษาหุ้นจากสองบริษัทในตลาดหลักทรัพย์แห่งประเทศไทย ได้แก่ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ซึ่งอยู่ในอุตสาหกรรมอสังหาริมทรัพย์ และบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ซึ่งอยู่ในอุตสาหกรรมพลังงาน เนื่องจากทั้งสองบริษัทนี้มีลักษณะทางการเงินที่เสถียรและเป็นที่นิยมในการลงทุนจัดอยู่ในกลุ่มหุ้นปลอดภัย

การวิจัยนี้มุ่งหวังที่จะจำแนกและวิเคราะห์ความรู้สึกจากข่าวหุ้นของบริษัททั้งสอง เพื่อประเมินผลกระทบที่มีต่อราคาหุ้นและความผันผวนของตลาด ซึ่งจะช่วยให้นักลงทุนสามารถตัดสินใจลงทุนได้อย่างมีประสิทธิภาพและมั่นใจมากยิ่งขึ้น นอกจากนี้ยังสามารถใช้เป็นแนวทางในการปรับกลยุทธ์การลงทุนให้เหมาะสมกับสภาวะตลาดที่เปลี่ยนแปลงไปได้อย่างทันท่วงที

1.2 วัตถุประสงค์ของงานวิจัย

1) เปรียบเทียบตัวตัดคำภาษาไทยที่เหมาะสมกับข่าวหุ้นรายวันภาษาไทยระหว่าง PythaiNLP กับ DeepCut
2) เปรียบเทียบวิธีการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันภาษาไทยด้วยแบบจำลอง 10 แบบ ได้แก่ การถดถอยเชิงโลจิสติค (Logistic Regression) นาอีฟเบย์ (Naïve Bayes) การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ป่าสุ่ม (Random Forest) ต้นไม้ตัดสินใจ (Decision Tree) ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) เพอร์เซปตรอน (Perceptron) สโตแคสติกการ์เดียนดิเซนท์ (SGD) พาสซีฟ อากัสซีฟ (Passive Aggressive)

1.3 ขอบเขตของงานวิจัย

1) งานวิจัยนี้ทำนายทิศทางราคารายวันด้วยวิธีการวิเคราะห์ความรู้สึกจากหัวข้อข่าวของบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH)
2) การวิจัยจะใช้ข้อมูลข่าวรายวันภาษาไทย และ ภาษาอังกฤษ จาก จากเว็บไซต์ “ข่าวหุ้น” ย้อนหลัง 5 ปี (พ.ศ. 2561 - 2565)

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ทำให้นักลงทุนที่สนใจลงทุนในหุ้น QH และ RATCH สามารถนำวิธีการไปสร้างรูปแบบกลยุทธ์ในการลงทุน เพื่อให้เกิดผลตอบแทนจากการลงทุนเพิ่มขึ้นและลดความเสี่ยงที่จะได้รับในการลงทุน
2) เป็นแนวทางสำหรับผู้สนใจที่นำข่าวมาทำ Natural Language Processing ภาษาไทยเพื่อพยากรณ์หุ้นไทยในตัวอื่น ๆ ที่สนใจ

1.5 คำนิยาม

หุ้นปลอดภัย เป็นหุ้นที่มีความทนทานในทุกสภาพตลาด ทั้งตลาดขาขึ้นและขาลง หุ้นกลุ่มนี้จะเป็นหุ้นที่มีพื้นฐานค่อนข้างแข็งแกร่ง ความเสี่ยงต่ำ และมีการจ่ายเงินปันผลค่อนข้างสม่ำเสมอ แต่ก็อาจเป็นหุ้นที่กำไรเติบโตไม่หวือหวา หรืออาจไม่ค่อยมีเรื่องราวการเติบโตที่จะนำมาเป็นจุดขายของหุ้นสักเท่าไหร่ (ฐิติเมธ, 2565) ตลาดหลักโดยจะใช้เงื่อนไขนี้ในการพิจารณาดังนี้

ก. มาร์เก็ตแคปมากกว่า 10,000 ล้านบาท
ข. ค่า Beta ต่ำกว่า 1
ค. D/E Ratio ต่ำกว่า 1 เท่า
ง. กำไรสุทธิมากกว่า 0 (ห้ามขาดทุน) ตลอด 5 ปี (พ.ศ. 2560 - 2564)
จ. P/E Ratio ต่ำกว่า 15 เท่า

ฉ. P/BV Ratio ไม่เกิน 5 เท่า

ช. จ่ายเงินปันผล 10 ปีติดต่อกัน

QH คือ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) ลักษณะธุรกิจพัฒนาอสังหาริมทรัพย์เพื่อขายและให้เช่า บ้านพร้อมที่ดิน หน่วยในอาคารชุดพักอาศัย อาคารที่พักอาศัยให้เช่า (ธุรกิจเซอร์วิส อพาร์ตเมนต์ โรงแรม) อาคารสำนักงาน รวมทั้งรับจ้างบริหาร และร่วมลงทุนในธุรกิจอื่นๆ (ตลาดหลักทรัพย์แห่งประเทศไทย, 2565)

RATCH คือ บริษัท ราช กรุ๊ป จำกัด (มหาชน) ลักษณะธุรกิจบริษัทฯ ประกอบธุรกิจในรูปแบบบริษัทโฮลดิ้ง โดยลงทุนถือหุ้นในบริษัทอื่น ซึ่งมีสถานะเป็นบริษัทหลัก บริษัทย่อย และ/หรือ บริษัทร่วมค้าของบริษัทฯ ขึ้นอยู่กับสัดส่วนการถือหุ้นของบริษัทฯ ที่ผ่านมาบริษัทฯ ได้ลงทุนในบริษัทพัฒนาโครงการโรงไฟฟ้าที่ใช้เชื้อเพลิงหลักประเภทต่างๆ โครงการพลังงานทดแทน ตลอดจนธุรกิจเกี่ยวเนื่องกับการผลิตไฟฟ้าและธุรกิจพลังงานด้านอื่นๆ ทั้งในประเทศและต่างประเทศ รายได้หลักของบริษัทฯ มาจากเงินปันผลและส่วนแบ่งกำไร (ตลาดหลักทรัพย์แห่งประเทศไทย, 2565)

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการศึกษาครั้งนี้ผู้วิจัยได้นำเสนอเนื้อหาที่เน้นถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องโดยมีรายละเอียดของเนื้อหาประกอบ ดังต่อไปนี้

2.1 การเตรียมข้อมูลข้อความ

2.2 การตัดคำด้วย PythaiNLP และ DeepCut

2.3 การจำแนกความรู้สึก

2.4 การเรียนรู้แบบมีผู้สอนเพื่อจำแนกความรู้สึกของข่าวหุ้นรายวัน

2.5 การเปรียบเทียบผลการทำนาย

2.6 งานวิจัยที่เกี่ยวข้อง

2.1 การเตรียมข้อมูลข้อความ

การเตรียมข้อมูลในงานวิจัยฉบับนี้จะเป็นการเตรียมข้อมูลสำหรับข้อความซึ่งขั้นตอนที่ใช้ในการเตรียมข้อมูลมีดังนี้

2.1.1 การตัดคำ (Tokenization)

การตัดคำเป็นหนึ่งในงานที่สำคัญของการประมวลผลภาษาธรรมชาติ เป็นงานส่วนใหญ่ที่ใช้งานเกี่ยวกับการประมวลผลภาษาธรรมชาติ จะต้องทำการตัดคำก่อนที่จะดำเนินการในส่วนอื่นต่อไป จะต้องทำการตัดคำก่อนที่จะดำเนินการวิเคราะห์ตามหลักไวยากรณ์ในการตัดคำในภาษาไทย จีน ญี่ปุ่น จะไม่ง่ายเหมือนกับ ภาษาอังกฤษ ภาษาสเปนเพราะคำภาษาไทยไม่ได้ถูกแบ่งส่วนเหมือนในภาษาอังกฤษ เช่น การเว้นวรรค, การจุลภาค เป็นต้นวิธีการตัดคำซึ่งแบ่งออกเป็น 2 ประเภทที่แตกต่างกัน ได้แก่

การตัดคำโดยใช้พจนานุกรม (Dictionary-based) วิธีการตัดคำโดยใช้ จะใช้ชุดข้อมูลจากพจนานุกรมสำหรับการวิเคราะห์และตัดคำวิธีการนี้จะเข้าไปค้นหาลำดับของอักขระในพจนานุกรมเพื่อจับคู่คำที่ถูกต้อง แต่การตัดคำโดยใช้พจนานุกรมจะมีปัญหาคำที่ไม่มีให้พจนานุกรมจะไม่สามารถตัดคำได้ ประสิทธิภาพของวิธีการตัดคำโดยใช้พจนานุกรมขึ้นอยู่กับคุณภาพและขนาดของพจนานุกรม และสามารถปรับปรุงประสิทธิภาพโดยการเพิ่มคำใหม่หรือคำเฉพาะที่ไม่มีในพจนานุกรมลงในพจนานุกรม ที่ใช้สำหรับกระบวนการตัดคำได้ ยกตัวอย่างการตัดคำจากพจนานุกรม เช่น Maximal matching

การตัดคำโดยใช้วิธีการ Dictionary-based ขั้นตอนวิธี Maximal matching วิธีนี้จะพิจารณาข้อความทั้งหมดทีละคำจากซ้ายไปขวา การตัดคำจะเข้าไปค้นหาคำใน พจนานุกรม ถ้าคำที่ค้นหาเจอในพจนานุกรมให้เลือกจำนวนคำที่ตัดได้จำนวนน้อยที่สุดยกตัวอย่างเช่น “ไปหามเหสี” พจนานุกรมตัวอย่างคือ [1 า, ม, เ, ส, ี, ไป, หา, หาม, เห, สี, มเหสี] การเข้าไปค้นหาในพจนานุกรม

จะเริ่มด้วย ตำแหน่งที่ 1 แถวที่ 1 ตัว คือ “ไ” ถูกพบในพจนานุกรมจะถูกระบุเป็น 1 ในตำแหน่ง 2 แถวที่ 1 คือ “ป” เมื่อรวมกับตำแหน่งที่ 1 และ ตำแหน่งที่ 2 จะเป็นคำว่า “ไป” พบในพจนานุกรม จะถูกระบุเป็น 1 ในตำแหน่งที่ 3 คือคำว่า “ไปห” ซึ่งไม่มีคำนี้ในพจนานุกรมจึงถูกระบุเป็น “inf” หมายถึงไม่พบคำนี้ในพจนานุกรม ต่อมาในแถวที่ 2 จะเริ่มต้นด้วยตำแหน่งที่ 2 แต่จะดูค่าย้อนกลับไปในตำแหน่งที่ 1 แถวที่ 1 แล้วเลือกค่าที่น้อยที่สุดซึ่งคือ 1 เป็นตัวอักษร “ไ” เมื่อนำมารวมกันได้เป็น {“ไ”, “ป”} 2 คำนี้เจอในพจนานุกรมในตำแหน่งที่ 2 แถวที่ 2 จึงถูกระบุเป็น 2 ในตำแหน่งต่อมาจะเป็น {“ไ”, “ปห”} ซึ่ง “ปห” ไม่มีในพจนานุกรมจึงถูกระบุเป็น “inf” โดยจะเข้าไปค้นหาจนครบทุกตำแหน่ง เช่น คำในตำแหน่งที่ 3 เริ่มที่แถวที่ 3 โดยผลจะเป็น {“ไป”, “ห”} , {“ไป”, “หา”}, {“ไป”,“หาม”} ทั้งหมดถูกพบในพจนานุกรมจึงถูกระบุเป็น 2 ในตำแหน่งที่ 3, 4, 5 ตามลำดับเมื่อเข้าไปค้นหาคำทั้งหมดในพจนานุกรมแล้ว ต้องทำการคำนวณย้อนกลับโดยเริ่มต้นที่ ตำแหน่งที่ 9 เลือกค่าที่น้อยที่สุดที่ถูกพบในพจนานุกรม คือ 3 อยู่ในแถวที่ 5 เป็นจุดเริ่มต้น ย้อนกลับไปจนไม่พบค่าในตำแหน่งนั้นหรือไม่เป็นค่า “inf” คือในตำแหน่งที่ 5 รวมกันเป็นคำว่า “มเหสี” ต่อมาเลือกจุดเริ่มต้นที่ไม่พบค่าในตำแหน่งนั้น คือ ตำแหน่งที่ 4 โดยจะคำนวณย้อนกลับไปจนเสร็จสิ้นจะได้คำว่า “หา”, “ไป” เมื่อนำมารวมกัน {“ไป”, “หา”, “มเหสี”}

โดยสรุปว่า “ไปหามเหสี” จะถูกตัดคำเป็น ไป|หา|มเหสี การตัดคำโดยใช้เทคนิคการเรียนรู้เครื่อง (Machine Learning-based) การตัดคำโดย Machine learning-based จะอาศัยแบบจำลองทางสถิติจากเทคนิคการเรียนรู้ด้วยเครื่อง โดยเทคนิคนี้ใช้การติดแท็กคำโดยมีการระบุอักขระที่เริ่มต้นของคำ และอักขระภายในคำข้อดีของ Machine Learning-based ไม่ต้องมีพจนานุกรมรองรับในการตัดคำ แต่จะขึ้นอยู่คลังข้อมูลที่นำมาฝึกฝนแบบจำลองและปัญหาคำที่ไม่รู้จักและคำที่กำกวมจะได้รับการจัดการโดยการเตรียมชุดตัวอย่างการฝึกฝนของแบบจำลองมากเพียงพอเพื่อให้การตัดคำได้แม่นยำ

2.1.2 การลบเครื่องหมายวรรคตอน (Punctuation Removal)

ในการวิจัยนี้มีการแบ่งแยกการทำงานในส่วนนี้คือ ถ้าเป็นข้อความข่าวภาษาอังกฤษ จะไม่ทำการลบเครื่องหมายเว้นวรรคตอน ลบเครื่องหมายต่างๆ แต่ถ้าเป็นข้อความภาษาไทยจะทำการการลบเครื่องหมายต่างๆ ที่ไม่จำเป็นในการวิเคราะห์เช่น เครื่องหมายลูกน้ำ เครื่องหมายอัศเจรีย์ และเครื่องหมายเว้นวรรค เช่น ประโยค "ปิดดีลซื้อ NexifEnergy กว่า2หมื่นล้านบาท!" จะถูกแปลงเป็น "ปิด", "ดีล", "ซื้อ", "NexifEnergy", "2", "หมื่น", "ล้าน", "บาท"

2.1.3 การนอร์มอลไลซ์ตัวอักษร (Character Normalization)

การนอร์มอลไลซ์ตัวอักษรคือ การแปลงคำต่างๆ ให้หลายเป็นตัวอักษรในแบบเดียวกันเพื่อความสะดวกในการวิเคราะห์ เช่น การแปลงเป็นตัวอักษรตัวเล็กในภาษาอังกฤษเนื่องจากคอมพิวเตอร์จะมองตัวอักษรใหญ่และเล็กเป็นคนละตัวกัน เช่น ประโยค "ปิดดีลซื้อ NexifEnergy กว่า2หมื่นล้านบาท!" จะถูกแปลงเป็น "ปิด", "ดีล", "ซื้อ", "nexifenergy", "2", "หมื่น", "ล้าน", "บาท"

2.1.4 การลบคำฟุ่มเฟือย (Stop Word Removal)

การลบคำฟุ่มเฟือย เป็นคำที่ตัดออกได้โดยที่ข้อความยังสื่อความหมายเดิม ในประโยคบทวิจารณ์เพื่อลดความซ้ำซ้อนก่อนในไปใช้วิเคราะห์ เช่น "เฮ!ศาลฯยกฟ้องปมถูกกล่าวหาผิดข้อตกลงเข้าประมูลโรงไฟฟ้า" จะถูกตัดคำฟุ่มเฟือยออกจะเป็น "เฮ", "ศาล", "ยกฟ้อง", "ปม", "กล่าวหา", "ข้อตกลง", "ประมูล", "โรงไฟฟ้า"

```python
from pythainlp.corpus.common import thai_stopwords
# โหลดคำที่ไม่สื่อความหมายในภาษาไทย
thai_stopwords2 = thai_stopwords()

# แสดงรายการคำที่ไม่สื่อความหมาย
print(thai_stopwords2)
```

รูปที่ 2.1 ตัวอย่างการใช้งานแสดงคำฟุ่มเฟือย

```
frozenset({'แต่ทว่า', 'ขาว', 'เข้าใจ', 'มีแต่', 'ล้วนแต่', 'ช่วงที่', 'รวมทั้ง',
```

รูปที่ 2.2 ตัวอย่างผลของคำฟุ่มเฟือย

2.1.5 การวิเคราะห์ความถี่ของคำในแต่ละกลุ่ม (Analysis Word Frequency in Groups)

การวิเคราะห์ความถี่ของคำในแต่ละกลุ่มโดยใช้จะใช้เครื่องมือในการสร้างภาพ Word Cloud ของ Python โดยกระบวนการสร้างภาพแสดงความถี่ของคำที่พบจะแสดงถึงความถี่ของคำที่ปรากฏในข้อความหรือกลุ่มข้อมูล โดยคำที่ปรากฏบ่อยจะมีขนาดใหญ่กว่า และคำที่ปรากฏน้อยจะมีขนาดเล็กกว่า วิธีนี้ช่วยให้สามารถมองเห็นคำที่สำคัญและบ่อยในข้อมูลได้ง่ายขึ้นเพื่อนำไปวิเคราะห์คำต่างๆ ที่พบ โดยการวิเคราะห์คำเหล่านี้ช่วยให้เราเข้าใจถึงข้อมูลที่มีความรู้สึกในด้านต่างๆ ในข่าวหุ้น เช่น การรายงานมูลค่าทางการเงิน การดำเนินงานของโครงการ และกิจกรรมการซื้อขายหุ้น การใช้ Word Cloud ช่วยให้เราสามารถมองเห็นภาพรวมของข้อมูลได้อย่างรวดเร็วและง่ายดาย

```python
path = 'THSarabunNew.ttf'
regexp = r"[ก-๙a-zA-Z']+"
wordcloud = WordCloud(
                      font_path='/content/drive/MyDrive/Data/data project/THsarabunNew.ttf',
                      min_font_size=1,
                      background_color="white",
                      width=400,
                      height=200,
                      max_words=1000,
                      colormap='plasma',
                      scale=3,
                      font_step=4,
                    #   contour_width=3,
                      contour_color='steelblue',
                      collocations=False,
                      regexp=regexp,
                      margin=2
                      ).generate(text_str)
```

```python
fig, ax = plt.subplots(1, 1, figsize=(16, 12))
ax.imshow(wordcloud, interpolation='bilinear')
ax.axis("off")
fig.show()
```

รูปที่ 2.3 ตัวอย่างการใช้งานการสร้างภาพแสดงความถี่ของคำที่พบ

รูปที่ 2.4 ตัวอย่างภาพแสดงความถี่ของคำที่พบ

2.1.6 การสกัดคุณลักษณะ (Feature Extraction)

การสกัดคุณลักษณะ (Feature Extraction) เป็นกระบวนการแปลงข้อมูลให้อยู่ในรูปแบบที่คอมพิวเตอร์สามารถประมวลผลได้เรียกว่าการทำดัชนี (Indexing) เพื่อสร้างตัวแทนเอกสาร (Document Terms Matrix) ที่อยู่ในรูปของเวกเตอร์ของน้ำหนักคำ (Term Weighting) ซึ่งการทำดัชนี (Indexing) สามารถทำได้หลายวิธี เช่น การนับจำนวนคำหรือนับความถี่ของคำที่ปรากฏในเอกสารแล้วทำการแทนค่าด้วยเวกเตอร์ เป็นต้น สำหรับเทคนิคที่นำมาใช้ในการสกัดคุณลักษณะมีหลายเทคนิค แต่ในงานวิจัยนี้จะประยุกต์ใช้เทคนิคการสกัดคุณลักษณะด้วยวิธีการทางสถิติ โดยใช้การให้น้ำหนักของคำในเอกสารแบบ TF-IDF (Term Frequency-Inverse Document Frequency) ซึ่งเป็นวิธีที่ใช้ประเมินความสำคัญของคำหรือคุณลักษณะ สามารถพิจารณาได้จากค่าน้ำหนัก TF-IDF คุณลักษณะใดที่มีค่าน้ำหนัก TF-IDF สูงก็จะหมายความว่าคุณลักษณะนั้นมีประสิทธิภาพในการจำแนกเอกสารสูงเช่นกัน จากนั้นจึงนำค่าคุณลักษณะที่สกัดได้มาแปลงให้อยู่ในรูปของตัวแทนเอกสาร (Document Terms Matrix) ก่อนที่จะนำเข้าสู่ขั้นตอนการพัฒนาแบบจำลองในลำดับถัดไป โดยสูตรการคำนวณหาค่าน้ำหนัก TF-IDF สามารถแสดงได้ดังสมการที่ (2.1)

$$TFIDF_{f,d} = TF_{f,d} \times \log \frac{|D|}{|DF_f|} \qquad (2.1)$$

โดยที่ $TFIDF_{f,d}$ คือ ค่าน้ำหนักของคุณลักษณะ (f) ที่ปรากฏในเอกสาร (d)

$TF_{f,d}$ คือ ความถี่ของคุณลักษณะ (f) ที่ปรากฏในเอกสาร (d)

$|D|$ คือ จำนวนเอกสารทั้งหมดในชุดข้อมูล

$|DF_f|$ คือ จำนวนของเอกสารทั้งหมดที่มีคุณลักษณะ (f) ปรากฏอยู่

2.2 การตัดคำด้วย PythaiNLP และ DeepCut

การตัดคำ (Tokenization) เป็นกระบวนการที่สำคัญในงานประมวลผลภาษาธรรมชาติ (Natural Language Processing - NLP) โดยเฉพาะในภาษาไทยที่ไม่มีการเว้นวรรคระหว่างคำ การตัดคำจะช่วยให้ระบบสามารถแยกและเข้าใจแต่ละคำในประโยคได้ง่ายขึ้น

2.2.1 PythaiNLP

PyThaiNLP เป็นไลบรารีสำหรับการประมวลผลภาษาธรรมชาติในภาษาไทย โดยมีฟังก์ชันหลายอย่าง เช่น การตัดคำ การวิเคราะห์คำ การแปลงข้อความ และการตรวจสอบการสะกดคำ และแก้ไขคำผิด สำหรับการตัดคำ PyThaiNLP ใช้โมเดลการตัดคำแบบดั้งเดิมและการตัดคำตามกฎไวยากรณ์ของภาษาไทย โมเดลเหล่านี้ถูกพัฒนาขึ้นมาเพื่อให้สามารถตัดคำในภาษาไทยได้อย่างแม่นยำและรวดเร็ว ด้วยการใช้เทคนิค Maximum Matching โดยอาศัยฐานข้อมูลคำศัพท์และกฎเกณฑ์ต่าง ๆ ในภาษาไทย (Wannaphong et al., 2023)

PyThaiNLP ถูกพัฒนาโดยชุมชนผู้พัฒนาที่สนใจในการประมวลผลภาษาธรรมชาติในภาษาไทย ซึ่งเป็นโครงการโอเพ่นซอร์ส (Open Source) ที่มีการร่วมมือจากนักพัฒนาหลายคน โดยมีการสนับสนุนและปรับปรุงอย่างต่อเนื่องจากผู้ใช้และนักพัฒนาในชุมชน

```python
from pythainlp.tokenize import word_tokenize

text = "ฉันรักเมืองไทย"
tokens = word_tokenize(text, engine='newmm')
print(tokens)
```

รูปที่ 2.5 ตัวอย่างการใช้งานตัวตัดคำ PythaiNLP

```
['ฉัน', 'รัก', 'เมือง', 'ไทย']
```

รูปที่ 2.6 ตัวอย่างผลของตัวตัดคำ PythaiNLP

2.2.2 DeepCut

DeepCut เป็นโมเดลการตัดคำภาษาไทยที่ใช้เทคนิค Deep Learning โดยอาศัยโครงข่ายประสาทเทียมแบบลึก (Deep Neural Network) ในการตัดคำ โมเดลนี้ถูกฝึกด้วยข้อมูลขนาดใหญ่ที่มีการติดป้ายกำกับ (Labeled Data) เพื่อให้สามารถเรียนรู้การตัดคำได้อย่างแม่นยำ DeepCut ใช้การเรียนรู้แบบลึก (Deep Learning) โดยใช้เทคนิคต่างๆ เช่น Long Short-Term Memory (LSTM) หรือ Convolutional Neural Network (CNN) ในการสร้างโมเดล ทำให้สามารถจัดการกับความซับซ้อนของภาษาไทยได้ดีและมีความยืดหยุ่นสูงในการตัดคำ (Rakpong et al., 2019)

```
import deepcut

text = "ฉันรักเมืองไทย"
tokens = deepcut.tokenize(text)
print(tokens)
```

รูปที่ 2.7 ตัวอย่างการใช้งานตัวตัดคำ DeepCut

```
1/1 [==============================] - 0s 27ms/step
['ฉัน', 'รัก', 'เมืองไทย']
```

รูปที่ 2.8 ตัวอย่างผลของตัวตัดคำ DeepCut

2.3 การจำแนกความรู้สึก

การจำแนกความรู้สึก (Sentiment Analysis) คือกระบวนการที่ใช้ในการวิเคราะห์ข้อความเพื่อหาความรู้สึกหรือทัศนคติที่แฝงอยู่ในข้อความนั้น ๆ ว่าเป็นบวก (Positive) ลบ (Negative) หรือเป็นกลาง (Neutral) การจำแนกความรู้สึกนี้มักจะใช้ในหลากหลายด้าน เช่น การวิเคราะห์ความคิดเห็นของลูกค้า การวิเคราะห์ความรู้สึกในโซเชียลมีเดีย และการวิเคราะห์ข่าวหรือบทความ ตัวอย่างการจำแนกความรู้สึก (วรเทพ, 2560) ยกตัวอย่างเช่น

"สินค้านี้ดีมาก คุณภาพเยี่ยม!" (บวก)

"บริการแย่มาก ส่งของช้า" (ลบ)

"วันนี้อากาศดีมาก เหมาะกับการออกไปเดินเล่น" (บวก)

"เบื่อหน่ายกับการจราจรติดขัดทุกวัน" (ลบ)

การจำแนกความรู้สึกสามารถทำได้หลายวิธี ทั้งการใช้กฎเกณฑ์แบบดั้งเดิม (Rule-Based Approach) การใช้เทคนิคการเรียนรู้ของเครื่อง (Machine Learning) สำหรับการใช้งานในเชิงธุรกิจและวิจัย การจำแนกความรู้สึกสามารถช่วยให้ผู้ประกอบการหรือผู้วิจัยเข้าใจความคิดเห็นและความรู้สึกของกลุ่มเป้าหมายได้ดียิ่งขึ้น ซึ่งจะเป็นประโยชน์ต่อการปรับปรุงผลิตภัณฑ์ บริการ หรือกลยุทธ์ให้สอดคล้องกับความต้องการ

2.4 การเรียนรู้แบบมีผู้สอนเพื่อจำแนกความรู้สึกของข่าวหุ้นรายวัน

ในการจำแนกความรู้สึกของข่าวหุ้นรายวันนั้นเราจำเป็นต้องใช้ข่าวและคำตอบ (Label) โดยในงานวิจัยนี้มีการกำหนด คำตอบ (Label) เป็น 3 สถานะคือ Positive (Pos) Negative (Neg) และ Neutral (Nue) เพื่อที่จะมาฝึกสอน (Train) และทดสอบ (Test) ให้แก่แบบจำลองประเภทการเรียนรู้แบบมีผู้สอน โดยในงานวิจัยฉบับได้ใช้โมเดลทั้งหมด 10 แบบจำลองได้แก่ การถดถอยเชิงโลจิสติค (Logistic Regression) นาอีฟเบย์ (Naïve Bayes) การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ป่าสุ่ม (Random

Forest) ต้นไม้ตัดสินใจ (Decision Tree) ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) เพอร์เซปตรอน (Perceptron) สโตแคสติกการ์เดียนดิเซนท์ (SGD) พาสซีฟ อากัสซีฟ (Passive Aggressive) แบบจำลองแต่ละตัวใช้วิธีการเรียนรู้และการประมวลผลข้อมูลที่แตกต่างกัน อย่างแบบจำลองเชิงเส้น ใช้การถดถอยเชิงโลจิสติค (Logistic Regression) การใช้แนวคิดของทฤษฎีความน่าจะเป็น นาอีฟเบย์ (Naïve Bayes) และการตัดสินใจของต้นไม้ใช้ป่าสุ่ม (Random Forest) ต้นไม้ตัดสินใจ (Decision Tree) อีกทั้งการใช้หลายแบบจำลองยังช่วยให้สามารถเปรียบเทียบประสิทธิภาพระหว่างแบบจำลองได้ เพื่อเลือกแบบจำลองที่มีประสิทธิภาพดีที่สุดในด้านความแม่นยำ ความรวดเร็วในการประมวลผลและการจัดการกับข้อมูลขนาดใหญ่หรือขนาดเล็ก (Kotsiantis, 2007)

2.4.1 การถดถอยเชิงโลจิสติค (Logistic Regression)

ขั้นตอนวิธีการจำแนกประเภทข้อมูล (Classification) ด้วยการประมาณค่าความน่าจะเป็นของข้อมูลเมื่อข้อมูลชุดนั้นมีคุณลักษณะที่สามารถจำแนกออกได้เป็น 2 ประเภท เช่น มีหรือไม่มี ใช่หรือไม่ใช่ เกี่ยวข้องหรือไม่เกี่ยวข้อง เป็นต้น ยกตัวอย่างการประมาณค่าความน่าจะเป็นของข้อมูลว่าเป็นข้อมูลประเภท 0 หรือ 1 จากค่าอัตราส่วนความน่าจะเป็น (Odds Ratio) ดังสมการที่ (2.2)

$$\text{Odds ratio} = \frac{p}{(1 - p)} \tag{2.2}$$

เมื่อ p คือ ความน่าจะเป็นของข้อมูลประเภทที่ 1

$(1 - p)$ คือ ความน่าจะเป็นของข้อมูลประเภทที่ 0

เนื่องจากความน่าจะเป็นมีค่าระหว่าง 0 ถึง 1 จึงต้องทำการปรับค่าผลของสมการที่ (2.1) ให้มีค่าอยู่ระหว่าง 0 ถึง 1 และใช้รูปแบบของ การถดถอยเชิงโลจิสติค (Logistic Regression) ดังสมการที่ (2.3)

$$ln(\text{Odds ratio}) = ln\left(\frac{p}{(1 - p)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n \tag{2.3}$$

เมื่อ β คือ สัมประสิทธิ์ของ X และ X คือ ตัวแปรที่ใช้ในการประมาณค่าความน่าจะเป็นในการจำแนกข้อมูลจำนวน n ตัว โดยการประมาณค่าความน่าจะเป็น (p) สามารถทำการประมาณได้จากสมการที่ (2.3) ซึ่งได้มาจากการปรับแก้สมการที่ (2.4) ดังนี้

$$p = \frac{1}{1 - e^{\beta X}} \tag{2.4}$$

โดยที่ $\beta X = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$

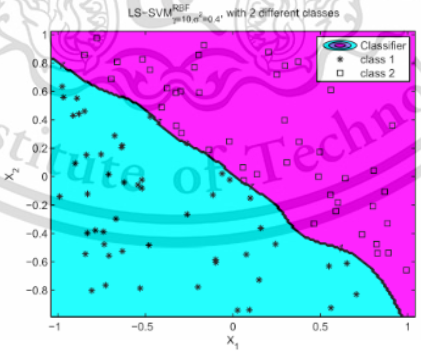
ซึ่งค่าความน่าจะเป็นที่ประมาณได้จากสมการที่ (2.4) จะถูกใช้ในการจำแนกข้อมูลว่าเป็นข้อมูลประเภทที่ 0 เมื่อค่าความน่าจะเป็นที่ได้ออกมามีค่าน้อยกว่า 0.5 และจะจำแนกว่าเป็นข้อมูลประเภทที่ 1 เมื่อค่าความน่าจะเป็นที่ได้ออกมามีค่ามากกว่าหรือเท่ากับ 0.5

2.4.2 ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines)

ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ตัวย่อคือ "SVM" หรืออีกชื่อเรียกว่า Kernel Machines เป็นเทคนิคหนึ่งของ Machine Learning ที่สามารถนำมาช่วยแก้ปัญหาการจำแนกข้อมูล (Classification) และ การถดถอย (Regression) สามารถ Classify Pattern ที่มีความสลับซับซ้อน (Complex) ได้อย่างมีประสิทธิภาพ เป็นเทคนิควิธีการเรียนรู้ของเครื่อง (Machine Leaning) ในรูปแบบการเรียนรู้เชิงสถิติ เป็นการเรียนรู้แบบมีผู้สอน (Supervised learning) โดย SVM ได้สร้างคิดค้นโดย Vladimir Vapnik และปัจจุบันได้ถูกนำมาใช้เป็นรูปแบบมาตรฐาน โดย Vapnik and Corinna Cortes ในปี 1995 (Zack, 2023)

พื้นฐานของ SVM รูปแบบของการใช้งาน SVM จะมี 2 ลักษณะ ดังนี้ Large Margin Separation หาแถบที่กว้างที่สุดที่จะแบ่งข้อมูลทั้ง 2 คลาสออกจากกัน (Linearly Separable Plane) โดยจะหาค่าที่มี Margin กว้างที่สุดระหว่างตัว Separator กับ Positive และ Negative Samples และได้ Feature Vector ที่อยู่ใกล้ Separator ที่สุด เรียกว่า Support Vectors ในการใช้งาน SVM จะต้องกำหนด Kernel Functions ซึ่งเป็นการเลือก Function ในการแปลงค่า Feature เดิมเป็น Feature ใหม่เพื่อให้ข้อมูลแยกจากกันโดยใช้ Linearly Separable Plane โดย Kernel ฟังก์ชัน

รูปแบบข้อมูลของ SVM Linearly Separation เป็นข้อมูลในมุมมอง 2 มิติ หรือเส้นตรง และ Hyperplane เป็นมุมมองแบบ 3 มิติ ซึ่งจะมองเห็นเป็นแผ่น



รูปที่ 2.9 ตัวอย่างผล SVM Classification

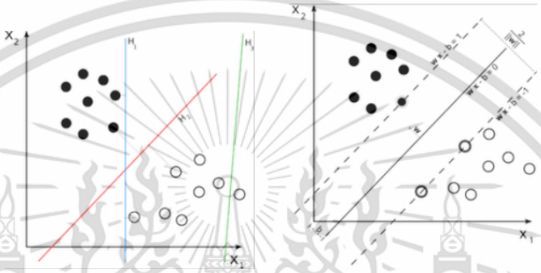
ที่มา : De Brabanter (2011)

จากรูปที่ 2.1 แสดงผลการใช้ SVM (LSSVM) ฝึกการเรียนรู้ (Training) ด้วย RBF Kernel ในการจัดกลุ่มของข้อมูลแยกออกเป็นสองกลุ่ม โดยป้อนข้อมูลสำหรับการเรียนรู้จากนั้นทดสอบด้วย

ข้อมูลสำหรับการทดสอบ (Testing) ก็จะได้คำตอบจาก SVM กลับมาว่า ข้อมูลทดสอบแต่ละรายการอยู่ในกลุ่มใดสมการของ Linear SVM เป็นดังนี้

$$D = \{(x_i, y_i) \mid x_i \in \mathbb{R}^p,\ y_i \in \{-1,1\}\}_{i=1}^{n} \quad (2.5)$$

จากสมการ Linear SVM เมื่อกำหนดค่า D เป็น กลุ่มของ Training Data เมื่อ y เป็น 1 หรือ -1 เท่านั้น โดยมีค่า x_i เป็นตัวบ่งบอกกลุ่ม โดยเราต้องการหาตำแหน่งที่จะแบ่งแยกที่ให้ค่า y_i = 1 จาก ค่า y_i เท่ากับ -1



ก **ข**

รูปที่ 2.10 การแบ่งข้อมูลด้วยวิธี SVM Classification ของข้อมูลที่มี 2 กลุ่ม

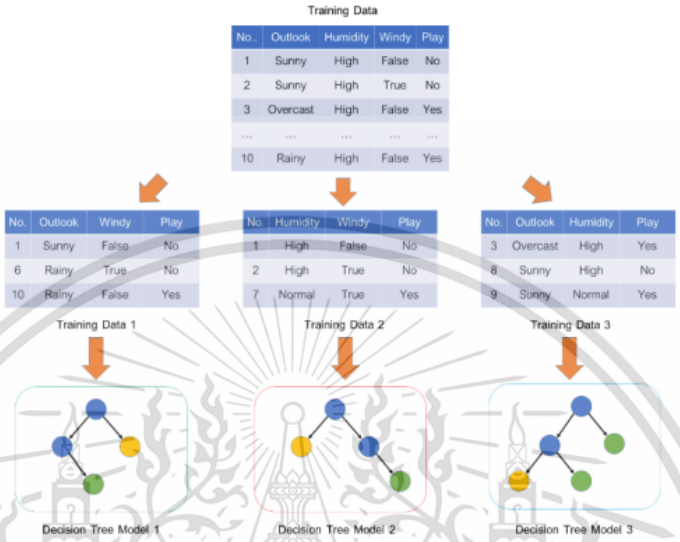
ที่มา : Zack (2023)

จากรูปที่ 2.2 ก แสดงผลของการแบ่งกลุ่มโดยใช้ SVM จะเห็นว่า ทั้งสองกลุ่มที่มาจาก Input Data เดียวกันจะถูกแยกออกจากกัน ในขณะที่เส้นที่ใช้แบ่งแยกมี H_1, H_2 และ H_3 โดยจะเห็นว่า H_1 แม้จะสามารถแบ่งทั้งสองกลุ่มออกได้เช่นกัน แต่ระยะในการแบ่งจากเส้นแบ่งไปถึงตัวอย่างที่ใกล้ที่สุดนั้นมีขนาดน้อย แต่จากเส้น H_2 จะเป็นเส้นที่แบ่งกลุ่มของสมาชิกที่กว้างมากที่สุดของทั้งสองกลุ่มคือให้ค่า Maximum Margin ซึ่งแสดงให้เห็นในรูปที่ 2.2 ข และเราเรียกตัวอย่างที่อยู่บน Margin นี้ว่า Support Vector โดยค่า Margin คือระยะห่างระหว่างไฮเปอร์เพลน (Hyperplane) ที่ใช้แยกกลุ่มข้อมูลต่างๆ กับตัวอย่างข้อมูลที่ใกล้ที่สุดในแต่ละคลาส

2.4.3 ป่าสุ่ม (Random Forest)

แบบจำลองการเรียนรู้ด้วยเครื่องป่าสุ่มเป็นเทคนิคหนึ่งของการเรียนรู้แบบรวมกลุ่ม (Ensemble Leaning) ซึ่งเป็นเทคนิคที่ใช้แบบจำลองการเรียนรู้ด้วยเครื่องชนิดจำแนกประเภทข้อมูล (Classification) หลายๆ แบบจำลองมาใช้ในการหาคำตอบ เนื่องจากแบบจำลองจำแนกประเภทข้อมูลเพียงแบบจำลองเดียวไม่ สามารถแยกแยะระหว่างค่าผิดปกติและรูปแบบได้ การเรียนรู้แบบรวมกลุ่มสามารถอธิบาย และแบ่งออกเป็น 2 ลักษณะกว้างๆ ดังนี้ 1) Vote Ensemble เป็นการใช้ข้อมูลสอน (Training Data) ชุดเดียวกันแต่สร้างแบบจำลองด้วยเทคนิคจำแนกประเภทข้อมูลที่แตกต่างกัน และ 2) Bootstrap Aggregating (Bagging) เป็นการสุ่มข้อมูลสอน ให้เป็นหลายชุด แต่

สร้างแบบจำลองด้วยเทคนิคเดียวกันทั้งหมด โดย Random Forest เป็นเทคนิคที่มีลักษณะคล้ายกับ Bagging แต่แทนที่จะสุ่มข้อมูลสอน (Training data) ให้เป็นหลายชุดเพียงอย่างเดียว ก็ทำการสุ่มคุณลักษณะของข้อมูล (Attribute) ด้วย และนำมาสร้างแบบจำลองด้วยเทคนิคต้นไม้ตัดสินใจ (Decision Tree) เพียงเทคนิคเดียว



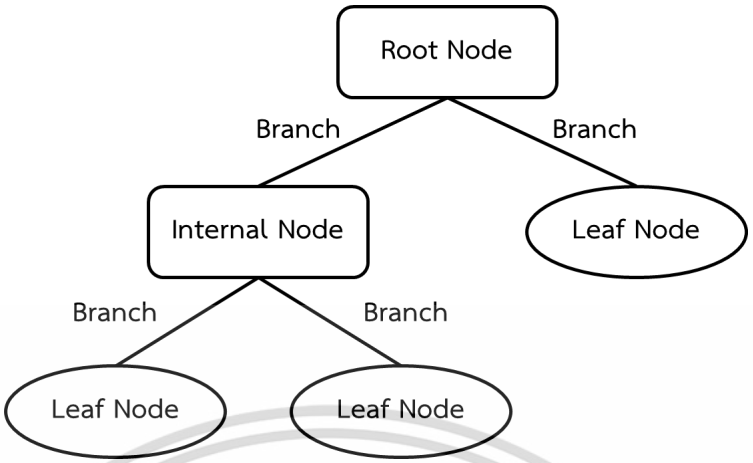
รูปที่ 2.11 การทำงานของ Random Forest modeling

ที่มา : Quinlan (1986)

จากการที่ต้นไม้ได้รับข้อมูลแบบสุ่มรวมถึงคุณลักษณะของข้อมูลที่ได้รับมีความแตกต่างกัน ดังนั้น ผลจากการตัดสินใจของต้นไม้จึงแตกต่างกัน แบบจำลอง Random Forest จะเลือกผลที่ต้นไม้ส่วนใหญ่ตอบเหมือนกัน (Voting) เป็นคำตอบของแบบจำลอง วิธีการนี้เป็นการแก้ปัญหาค่าผิดปกติของข้อมูล (Noise) และปัญหาที่แบบจำลองปรับตัวเข้ากันข้อมูลสอนมากจนเกินไป (Overfitting)

2.4.4 ต้นไม้ตัดสินใจ (Decision Tree)

ต้นไม้ตัดสินใจ (Decision Tree) เป็นขั้นตอนวิธีการจำแนกประเภทข้อมูล (Classification) ที่ได้รับความนิยม โดย Decision Tree หรือต้นไม้ตัดสินใจเป็นแบบจำลองที่มีโครงสร้างข้อมูลแบบเป็นลำดับชั้น (Hierarchy) เพื่อช่วยในการสนับสนุนการตัดสินใจเพื่อหาทางเลือกที่ดีที่สุด มีลักษณะคล้ายกับต้นไม้กลับหัวที่มีรากอยู่ด้านบนสุดและใบอยู่ล่างสุด ประกอบด้วยโหนด (Node) ซึ่งโหนดแรกสุดจะถูกเรียกว่าโหนดราก (Root Node) ในแต่ละโหนดก็จะแสดงคุณลักษณะ (Attribute) ที่ใช้จำแนกประเภทข้อมูลว่าจะให้เป็นไปในทิศทางใด แยกออกด้วยกิ่ง (Branch) ในจำนวนที่เท่ากับคุณลักษณะของแต่ละโหนด จนได้ผลในการจำแนกข้อมูลออกมาเป็นโหนดใบ (Leaf Node) โดยการสร้างต้นไม้ตัดสินใจจะถูกสร้างจากบนลงล่าง ทำการตัดสินใจว่าควรใช้คุณลักษณะ (Attribute) ใดเป็นโหนดราก (Root Node) ส่วนโหนดอื่น ๆ ในต้นไม้ที่ไม่ใช่โหนดรากและโหนดใบจะเรียกว่าโหนดภายใน (Internal Node) ซึ่งขั้นตอนวิธีการสร้างแบบจำลองต้นไม้ตัดสินใจนั้นสามารถสร้างด้วยอัลกอริทึม (Algorithm) ได้หลายแบบ เช่น ID3, ID4.5, ID5.0, CART และ CHARID เป็นต้น



รูปที่ 2.12 ส่วนประกอบของต้นไม้ตัดสินใจ (Decision Tree)

2.4.5 นาอีฟเบย์ (Naïve Bayes)

นาอีฟเบย์ (Naïve Bayes) เป็นขั้นตอนวิธีการจำแนกประเภทข้อมูล (Classification) ที่มีพื้นฐานมาจากทฤษฎีของ Bayes (Bayes Theorem) และสมมติฐานที่ให้การเกิดของเหตุการณ์ต่าง ๆ เป็นอิสระต่อกัน ทำการจำแนกประเภทข้อมูลโดยอาศัยหลักการทางสถิติและความน่าจะเป็น (Probability) ดังสมการที่ (2.6)

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

โดยที่ P(A) คือ ความน่าจะเป็นที่จะเกิดเหตุการณ์ A

P(B) คือ ความน่าจะเป็นที่จะเกิดเหตุการณ์ B

P(A|B) คือ ความน่าจะเป็นที่เหตุการณ์ A จะเกิดขึ้นถ้าเหตุการณ์ B เกิดขึ้นแล้ว

P(B|A) คือ ความน่าจะเป็นที่เหตุการณ์ B จะเกิดขึ้นถ้าเหตุการณ์ A เกิดขึ้นแล้ว

จากสมการที่ (2.5) สามารถพัฒนาเป็นสมการคำนวณความน่าจะเป็นของการจำแนกประเภทข้อมูลด้วยขั้นตอนวิธีการ Naïve Bayes ได้ดังสมการที่ (2.7)

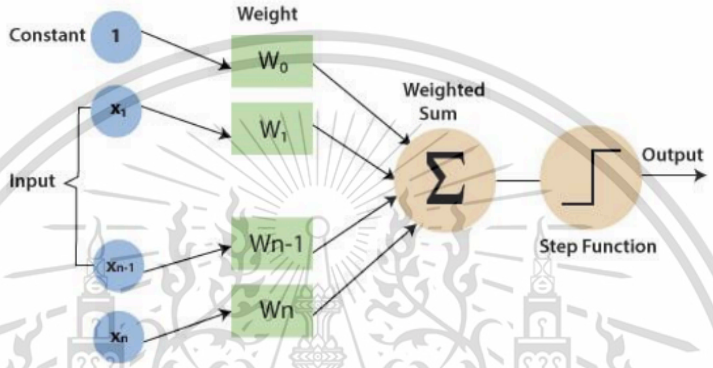
$$P(a_1 + a_2 + \ldots + a_n|v_j) = \prod_{i=1}^{n} P(a_i|v_j) \tag{2.7}$$

โดยที่ $\prod$ คือ ผลคูณของของค่า $P(a_i|v_j)$ ทั้งหมด i = 1, 2, 3, ..., n และ j = 1, 2, 3, ..., n เนื่องจาก Naïve Bayes เป็นวิธีการจำแนกข้อมูลที่ง่าย รวดเร็ว และมีประสิทธิภาพอีกวิธีหนึ่ง เหมาะกับกรณีที่มีชุดข้อมูลตัวอย่างจำนวนไม่มาก และคุณสมบัติของชุดข้อมูลไม่ขึ้นต่อกัน จึงสามารถ

ประยุกต์ใช้ Naïve Bayes ได้กับการจำแนกประเภทข้อมูลที่หลากหลาย เช่น การจำแนกประเภทข้อความ (Text Classification) หรือการวินิจฉัย (Diagnosis) เป็นต้น

2.4.6 เพอร์เซปตรอน (Perceptron)

การทำงานของเพอร์เซปตรอน (Perceptron) ดังรูปที่ 2.11 คล้ายคลึงกับการทำงานของเซลล์ประสาทสมองมนุษย์ประกอบด้วย 5 ส่วนหลักคือส่วนรับข้อมูลเข้า (Input) ค่าถ่วงน้ำ หนัก (Weight) ส่วนรวมผลคุณของค่าข้อมูลนำเข้าและค่าถ่วงน้ำหนัก (Weighted Sum) ค่าเบี่ยงเบน (Bias) และส่วนขจัดความเป็นเชิงเส้น (Activation Function)



รูปที่ 2.13 การทำงานของเพอร์เซปตรอน (Perceptron)
ที่มา : https://www.javatpoint.com/pytorch-perceptron

1. ค่าถ่วงน้ำหนัก (Weight)

ค่าถ่วงน้ำหนัก เป็นค่าพารามิเตอร์ที่เป็นส่วนประกอบของเพอร์เซปตรอน (Perceptron) ซึ่งจะมีการปรับเปลี่ยนไปตลอดจนกวา จะได้ค่าที่ดีที่สุด โดยค่าถ่วงน้ำหนัก จะถูกคูณเข้ากับค่าข้อมูลสำหรับแต่ละหน่วยของขอ้มูลนำเข้าก่อนที่จะถูกรวมและส่งผ่านส่วนขจัดเชิงเส้น

2. ค่าเบี่ยงเบน (Bias)

ค่าเบี่ยงเบนเป็นค่าพารามิเตอร์ที่มี 1 ตัวต่อ 1 เพอร์เซปตรอน (Perceptron) ซึ่งจะมีการปรับเปลี่ยนไปจนกว่าจะได้ค่าที่ดีที่สุดเหมือนกับค่าถ่วงน้ำหนัก โดยค่าเบี่ยงเบนจะถูกรวมเข้ากับผลรวมของผลคูณระหว่างค่าถ่วงน้ำหนักกับค่าข้อมูลนำเข้าทั้งหมดที่เข้าสู่เพอร์เซปตรอน (Perceptron)

3. ส่วนขจัดความเป็นเชิงเส้น (Activation Function)

ส่วนขจัดความเป็นเชิงเส้นเป็นฟังก์ชั่นที่รับผลรวมการประมวลผลทั้งหมด แล้วนำมาคำนวณในสมการที่ไม่เป็นเชิงเส้น หลังจากผ่านส่วนรวมผลคุณของค่าข้อมูลนำเข้าและค่าถ่วงน้ำหนัก

2.4.7 เพื่อนบ้านใกล้สุด k ตัว (K-nearest Neighbor Algorithm)

ขั้นตอนวิธีการค้นหาเพื่อนบ้านใกล้สุด k ตัว (K-nearest Neighbor Algorithm) ขั้นตอนวิธีการค้นหาเพื่อนบ้านใกล้สุด k ตัว หรือเรียกแบบย่อว่า KNN เป็นอัลกอริทึมการ เรียนรู้ของ

เครื่องแบบหนึ่งที่ใช้ต้นทุนการคำนวณต่ำและนำไปใช้งานได้ง่ายซึ่งรองรับการนำไปใช้งาน ทั้งปัญหาแบบการจำแนก (Classification) และแบบการถดถอย (Regression) เมื่อทำการทำนายจะ มีการเก็บชุดข้อมูลฝึกสอนทั้งหมดและค้นหาตำแหน่งของจุดข้อมูลจำนวน k จุดในชุดข้อมูลฝึกสอนที่ใกล้เคียงจุดข้อมูลที่ต้องการจำแนกมากที่สุด อย่างไรก็ตามไม่มีตัวแบบอื่นนอกจากที่มาจากชุดข้อมูลฝึกสอนและทำการคำนวณระยะทางระหว่างจุดคือการค้นหาจากชุดข้อมูลฝึกสอนเท่านั้น ในการใช้วิธี KNN ในปัญหาการถดถอยเพื่อทำการพยากรณ์ ค่าของตัวแปรตามจะถูกคำนวณ จากผลรวมถ่วงน้ำหนักของค่าตัวแปรตามของเพื่อนบ้านใกล้เคียง k ทั้งหมด โดยที่น้ำหนักจะ แปรผกผันกับระยะห่างระหว่างจุดข้อมูลป้อนเข้า (Input Record) และระยะห่างจะใช้การวัด ระยะทางแบบยูคลิด (Euclidean) ฟังก์ชันระยะทางแบบยูคลิดเขียนเป็นสมการได้ดังนี้ (AL-Dosary et al., 2019)

$$\mathrm{E}(x,p) = \sqrt{\sum_{i=1}^{n}(x_i - p_i)^2} \tag{2.8}$$

โดยที่ x และ p คือ จุดที่จะใช้หาระยะห่างจากกัน

n คือ จำนวนข้อมูลของตัวแปรป้อนเข้า (Attributes)

2.4.8 สโตแคสติกการ์เดียนดิเซนท์ (Stochastic Gradient Descent)

วิธีสโตแคสติกการ์เดียนดิเซนท์ (Stochastic Gradient Descent) เป็นวิธีที่มีประสิทธิภาพในการจำแนกการเรียนรู้ของตัว ภายใต้ฟังก์ชันการสูญเสียโค้งนูนจำแนกเชิงเส้นได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน(Support Vector Machines) และการถดถอยโลจิสติก (Logistic Regression) ใช้การเรียนรู้ตัวแบบเชิงเส้นต่าง ๆ ได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ที่มีคำตอบเป็นทวิภาคการถดถอยโลจิสติก (Logistic Regression) ที่มีคำตอบเป็นทวิภาค และการถดถอยเชิงเส้นแทนที่ข้อมูลสูญหายและแปลงคุณลักษณะเชิงกลุ่มให้เป็น ทวิภาค แปลงคุณลักษณะต่าง ๆ ให้อยู่ในรูปปรกติ (Normalization) ดังนั้นค่าสัมประสิทธิ์ของผล เป็นข้อมูลที่อยู่ในรูปปรกติ (Nektarios, 2013) สำหรับคุณลักษณะที่มีคำตอบเป็นนามบัญญัติจะใช้ฟังก์ชันสูญเสียไฮนด์ (Hinge Loss Function) หรือฟังก์ชันการสูญเสียล็อก (Log Loss Function) ส่วนคุณลักษณะที่มีคำตอบเป็นเชิงตัวเลขจะใช้ฟังก์ชัน การสูญเสียกำลังสอง (Squared Loss Function) ฟังก์ชันการสูญเสียเอฟซิลอน (Epsilon-Insensitive Loss Function) หรือฟังก์ชันการสูญเสียฮูเบอร์ (Huber Loss Function)

2.5 การเปรียบเทียบประสิทธิภาพการทำนาย

2.5.1 เมทริกซ์ความสับสน (Confusion Matrix)

เมทริกซ์ความสับสน (Confusion Matrix) เป็นเครื่องมือที่ใช้ในการประเมินประสิทธิภาพของแบบจำลองประเภทการจำแนก (Classification Model) โดยเฉพาะในกรณีที่มีหลายคลาส

(MulticlassClassification) เมทริกซ์นี้จะแสดงจำนวนการทำนายที่ถูกต้องและผิดพลาดในแต่ละคลาส ทำให้สามารถวิเคราะห์ข้อผิดพลาดและความแม่นยำของโมเดลได้อย่างละเอียด

2.5.1.1 โครงสร้างของเมทริกซ์ความสับสน (Confusion Matrix) ขนาด 3x3

งานวิจัยนี้ทำการวัดประสิทธิภาพของการแบบจำลองเพื่อจำแนกข้อความข่าวออกเป็น 3 ประเภท คือ Positive (Pos), Negative (Neg) และ Neutral (Nue) ทำให้ Confusion matrix มีขนาด 3 x 3 โดยจะมี 9 ช่อง (3 แถว x 3 คอลัมน์) โดยแต่ละช่องมีความหมายดังนี้

1. เมทริกซ์ความสับสน (Confusion Matrix) ของคลาส Positive (Pos)

ค่าแสดงความรู้สึกของคลาส Positive (Pos) ด้วย Confusion matrix ดังตารางที่ 2.1

ตารางที่ 2.1 Confusion matrix ของการจำแนกข้อความข่าว Pos

Confusion matrix		Predicted Class		
		Positive	Negative	Neutral
Actual Class	Positive	TP_{Pos}	FN_{Pos}	FN_{Pos}
	Negative	FP_{Pos}	TN_{Pos}	TN_{Pos}
	Neutral	FP_{Pos}	TN_{Pos}	TN_{Pos}

TP (True Positive) สำหรับ Pos จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive

FP (False Positive) สำหรับ Pos จำนวนข้อความที่ไม่เป็น Positive แต่ทำนายว่าเป็น Positive

FN (False Negative) สำหรับ Pos จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น

TN (True Negative) สำหรับ Pos จำนวนข้อความที่ไม่เป็น Positive และทำนายว่าไม่เป็น Positive

2. เมทริกซ์ความสับสน (Confusion Matrix) ของคลาส Negative (Neg)

ค่าแสดงความรู้สึกของคลาส Negative (Neg) ด้วย Confusion matrix ดังตารางที่ 2.2

ตารางที่ 2.2 Confusion matrix ของการจำแนกข้อความข่าว Neg

Confusion matrix		Predicted Class		
		Positive	Negative	Neutral
Actual Class	Positive	TN_{Neg}	FP_{Neg}	TN_{Neg}
	Negative	FN_{Neg}	TP_{Neg}	FN_{Neg}
	Neutral	TN_{Nue}	FP_{Neg}	TN_{Neg}

TP (True Positive) สำหรับ Neg จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative

FP (False Positive) สำหรับ Neg จำนวนข้อความที่ไม่เป็น Negative แต่ทำนายว่าเป็น Negative

FN (False Negative) สำหรับ Neg จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น

TN (True Negative) สำหรับ Neg จำนวนข้อความที่ไม่เป็น Negative และทำนายว่าไม่เป็น Negative

3. เมทริกซ์ความสับสน (Confusion Matrix) ของคลาส Neutral (Neu)

ค่าแสดงความรู้สึกของคลาส Neutral (Neu) ด้วย Confusion matrix ดังตารางที่ 2.3

ตารางที่ 2.3 Confusion matrix ของการจำแนกข้อความข่าว Nue

Confusion matrix		Predicted Class		
		Positive	Negative	Neutral
Actual Class	Positive	TN_{Nue}	TN_{Nue}	FP_{Nue}
	Negative	TN_{Nue}	TN_{Nue}	FP_{Nue}
	Neutral	FN_{Nue}	FN_{Nue}	TP_{Nue}

TP (True Positive) สำหรับ Nue จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral

FP (False Positive) สำหรับ Nue จำนวนข้อความที่ไม่เป็น Neutral แต่ทำนายว่าเป็น Neutral

FN (False Negative) สำหรับ Nue จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น

TN (True Negative) สำหรับ Nue จำนวนข้อความที่ไม่เป็น Neutral และทำนายว่าไม่เป็น Neutral

การคำนวณหาค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) และค่าความถ่วงดุล (F-Measure) สามารถคำนวณได้ดังสมการต่อไปนี้

2.5.2 ค่าความถูกต้อง (Accuracy)

ค่าความถูกต้องเป็นการวัดสัดส่วนของการทำนายที่ถูกต้องจากทั้งหมด

$$\text{ค่าความถูกต้อง (Accuracy)} = \frac{TP_{Pos} + TP_{Neg} + TP_{Nue}}{Total\ Sample} \quad (2.9)$$

2.5.3 ค่าความแม่นยำ (Precision)

ค่าความแม่นยำเป็นการวัดสัดส่วนของการทำนายที่ถูกต้องในแต่ละประเภทจากการทำนายทั้งหมดที่เป็นประเภทนั้น

ค่าความแม่นยำ (Precision) สำหรับ Pos $= \frac{TP_{Pos}}{TP_{Pos} + FP_{Pos} + FP_{Pos}}$ (2.10)

ค่าความแม่นยำ (Precision) สำหรับ Neg $= \frac{TP_{Neg}}{TP_{Neg} + FP_{Neg} + FP_{Neg}}$ (2.11)

ค่าความแม่นยำ (Precision) สำหรับ Nue $= \frac{TP_{Nue}}{TP_{Nue} + FP_{Nue} + FP_{Nue}}$ (2.12)

2.5.4 ค่าความระลึก (Recall)

ค่าความระลึกเป็นการวัดสัดส่วนของการทำนายที่ถูกต้องในแต่ละประเภทจากจำนวนข้อมูลที่เป็นจริงในประเภทนั้น

ค่าความระลึก (Recall) สำหรับ Pos $= \frac{TP_{Pos}}{TP_{Pos} + FN_{Pos} + FN_{Pos}}$ (2.13)

ค่าความระลึก (Recall) สำหรับ Neg $= \frac{TP_{Neg}}{TP_{Neg} + FN_{Neg} + FN_{Neg}}$ (2.14)

ค่าความระลึก (Recall) สำหรับ Nue $= \frac{TP_{Nue}}{TP_{Nue} + FN_{Nue} + FN_{Nue}}$ (2.15)

2.5.5 ค่าความถ่วงดุล (F-Measure)

ค่าความถ่วงดุลเป็นการวัดค่ากลางที่ถ่วงดุลระหว่าง Precision และ Recall

ค่าความถ่วงดุล (F-Measure) สำหรับ Pos $= 2 \text{ x } \frac{Percision_{Pos} \cdot Recall_{Pos}}{Percision_{Pos} + Recall_{Pos}}$ (2.16)

ค่าความถ่วงดุล (F-Measure) สำหรับ Neg $= 2 \text{ x } \frac{Percision_{Neg} \cdot Recall_{Neg}}{Percision_{Neg} + Recall_{Neg}}$ (2.17)

ค่าความถ่วงดุล (F-Measure) สำหรับ Nue $= 2 \text{ x } \frac{Percision_{Nue} \cdot Recall_{Nue}}{Percision_{Nue} + Recall_{Nue}}$ (2.18)

2.5.6 การเลือกโมเดลที่ดีที่สุด (Model Selection)

การเลือกโมเดลที่ดีที่สุดคือการเลือกโมเดลย่อยๆ จากโมเดลหลัก มาใช้ในการตัดสินใจร่วมกันแบบเสียงข้างมากโดยการเลือกโมเดลที่จะมาทำการตัดสินใจร่วมกันนั้นจะเลือกจากการพิจารณาค่าความถูกต้องที่สูงที่สุดของแต่ละโมเดลย่อย ซึ่งการนำโมเดลย่อยมาช่วยกันตัดสินใจจะทำให้สามารถเพิ่มประสิทธิภาพของการตัดสินใจของโมเดลได้ดีขึ้น (Salini and Jeyapriya, 2018)

2.6 งานวิจัยที่เกี่ยวข้อง

ในหัวข้อนี้จะกล่าวถึงงานวิจัยที่เกี่ยวข้องกับวิเคราะความรู้สึกจากข่าวและการรวบรวมข้อมูลข่าวและนำมาพยากรณ์โดยแบบจำลองต่าง ๆ

ศุภณัฐ ก้อนศิลา (2566) การพยากรณ์ทิศทางการเปลี่ยนแปลงราคาหุ้นในกลุ่ม SET50 โดยใช้ปัจจัยข่าว และการวิเคราะห์เชิงเทคนิค โดยมีการใช้หุ้นไทยในกลุ่ม SET50 ในการทำการทดลองและมีการใช้ข้อมูลสองชุดคือ ข้อมูลชุดที่ 1 ใช้ข่าวหุ้นรายวันของประเทศสหรัฐอเมริกา ช่วงเวลา 25/4/2561-15/01/2566 (ข่าวภาษาอังกฤษ) ข้อมูลชุดที่ 2 ใช้ราคาหุ้น จาก yfinance ของหุ้นแต่ละตัว และเอาตัวชี้วันทางการเงินอื่นๆ รายวันมาด้วย เช่น ราคา SET50 SET S&P500 ราคาดัชนีอุตสาหกรรมในประเทศการเตรียมข้อมูลส่วยปัจจัยข่าว เพื่อทำการแปลงข่าวให้เป็นตัวเลข โดยวิธีสกัดบทความข่าว โดยการหาความคล้ายคลึงกันของข้อความ แบ่งข่าวเป็น 7 ประเภท แล้วใช้ Library Fasttext จัดการจำแนกแล้วรวมคะแนนความคล้ายคลึงกันในแต่ละประเภทในแต่ละวันซึ่งในงานวิจัยนี้ไม่ได้ใช้การหาความคล้ายคลึงของข้อความ

วิกานดา และ อัญชนา (2563) โดยมีการศึกษาปัจจัยหนึ่งที่มีผลกระทบต่อราคาของหุ้นก็คือเรื่องของข่าวสาร ผู้วิจัยจึงเกิดแนวคิดที่จะพยากรณ์ทิศทางของราคาหุ้นรายวันจากข้อความข่าวโดยใช้วิธีการประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) เพื่อให้นักลงทุนสามารถคาดคะเนทิศทางของราคาหุ้นก่อนที่ตลาดหุ้นแห่งประเทศไทย โดยศึกษาข้อความข่าวจากแหล่งข่าวต่างๆ และใช้การตัดคำจาก Library PyhaiNLP แบบจำลองโดยใช้ตัวแบบการจำแนกเพื่อหาแบบจำลองที่มีค่าความถูกต้องแม่นยำสูงสุดเพื่อใช้พยากรณ์ทิศทางของราคาหุ้นรายวัน ผลสรุปก็มีความแตกต่างกันตามหุ้นแต่ละตัว

Bipin et al. (2021) เป็นงานวิจัยที่แสดงให้เห็นผลกระทบของปัจจัยหลายอย่างที่มีต่อราคาหุ้นโดยเอาข้อมูลการแสดงความคิดเห็นต่างๆ การตอบสนองต่อเหตุการณ์ใน Google Search Trends,

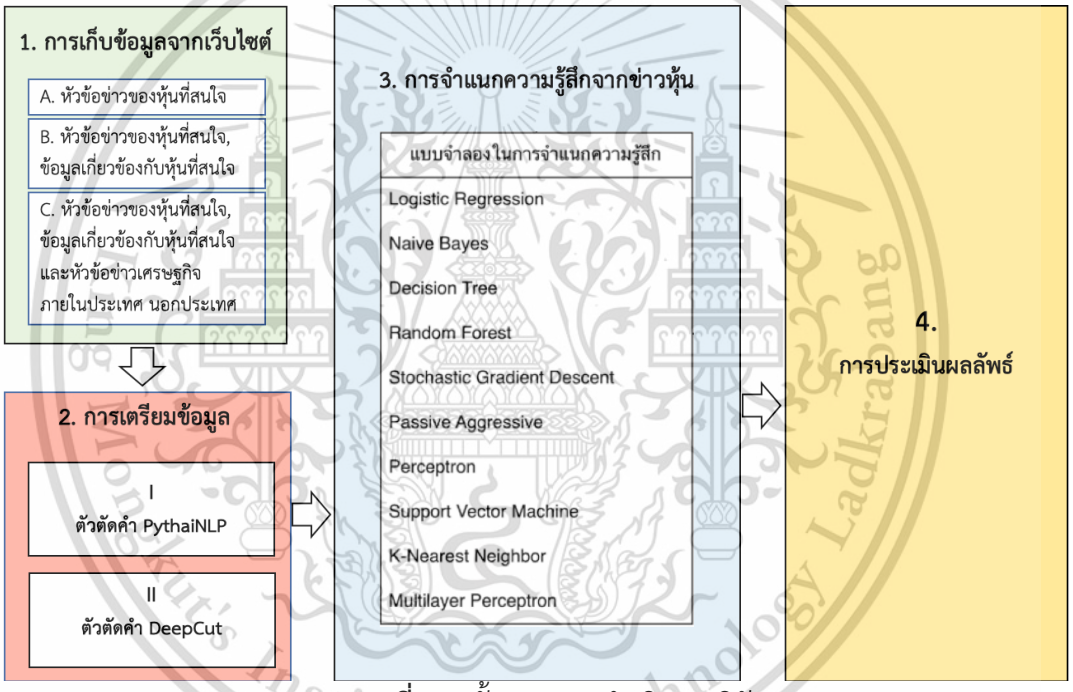
e-News headlines และ Tweets มีการเสนอแบบจำลองหน่วยความจำระยะสั้นหลายขั้นตอน Output หลายตัวแปร MMLSTM เพื่อให้การคาดการณ์หนึ่งสัปดาห์สำหรับมูลค่าปิดของหุ้นสำหรับบริษัทเทคโนโลยี “Apple Inc.” โดยใช้ชื่อหุ้นว่า “AAPL” โดยแบบจำลองของเขามีค่า Mean Square Error (MSE) สูงถึงร้อยละ 65 เมื่อเทียบกับแบบจำลอง ARIMA และ Random Forest นอกจากนี้ Multivariate Multistep Long Short-Term Memory ที่เสนอยังมีประสิทธิภาพดีกว่าแบบจำลอง Long Short-Term Memory

นิธิกร (2565) ได้ทำการวิเคราะห์กลุ่มความสนใจ ความรู้สึก และหาข้อมูลเชิงลึกของนักท่องเที่ยวด้วยวิธีการทำเหมืองข้อความ โดยข้อมูลที่ใช้คือบทวิจารณ์ออนไลน์จากนักท่องเที่ยวที่กล่าวถึงเยาวราช ประเทศไทย โดยข้อมูลที่ใช้จะเป็นบทวิจารณ์ออนไลน์จากเว็ป Tripadvisor ทั้งหมด 3,992 บทวิจารณ์ โดยผู้วิจัยได้ใช้แบ่งการวิเคราะห์ออกเป็น 4 ส่วนใหญ่ ส่วนที่หนึ่งคือ การวิเคราะห์กลุ่มความสนใจของนักท่องเที่ยว ด้วยการจัดกลุ่มแบบเคมีน ร่วมกับการจัดสรรหัวข้อแฝง ซึ่งพบว่านักท่องเที่ยวที่เดินทางมายังเยาวราช สามารถแบ่งความสนใจออกได้เป็น 4 กลุ่มได้แก่ แหล่งช้อปปิ้ง ตลาดอาหารริมทางยามค่ำ อาหาร และการเที่ยวชมเมือง ส่วนที่สองคือ การวิเคราะห์ความรู้สึกของนักท่องเที่ยวด้วยการจำแนกความรู้สึกเป็นเชิงบวกและเชิงลบ ซึ่งจะใช้โมเดลทั้งหมด 3 โมเดลได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีน โดยจะนำโมเดลทั้ง 3 มา ร่วมกันตัดสินใจซึ่งให้มีค่าความถูกต้องร้อยละ 88.62 ส่วนที่สามคือการหาข้อมูลเชิงลึกโดยใช้เทคนิคต่างๆ ได้แก่ การวิเคราะห์ความนิยม การวิเคราะห์ความเด่นและความแพร่หลาย การวิเคราะห์แนวโน้ม และส่วนสุดท้ายจะเป็นการเสนอแนวทางการประยุกต์ใช้โมเดลด้วยการสร้างเว็ปแอปพลิเคชั่นเพื่อใช้ในการวิเคราะห์บทวิจารณ์แบบอัตโนมัติ

ปกป้อง (2565) ได้ทำการพัฒนาแบบจำลองการจำแนกประเภทความคิดเห็นที่เกี่ยวข้องกับความมั่นคงโดยใช้เทคนิคเหมืองข้อมูล และเปรียบเทียบหาแบบจำลองที่มีประสิทธิภาพดีที่สุด ซึ่งผู้วิจัยได้เลือกเก็บข้อมูลจากทวิตเตอร์ตั้งแต่เดือนธันวาคม พ.ศ. 2564 จนถึงเดือนมกราคม พ.ศ. 2565 โดยใช้คำค้นที่เกี่ยวข้องกับองค์ประกอบของความมั่นคงแห่งชาติและจากคำแนะนำของผู้เชี่ยวชาญได้ข้อมูลทั้งหมด 1,638 ข้อความ ทำการฝึกฝนข้อมูลด้วยอัลกอริทึมการจำแนกประเภททั้งหมด 10 ชนิด คือ Logistic Regression, Naive Bayes, Decision Trees, Random Forest, Stochastic Gradient Descent, Passive Aggressive, Perceptron, Support Vector Machine (SVM), K- Nearest Neighbors (KNN) และ Neural Network ผลพบว่าแบบจำลอง Random Forest มีประสิทธิภาพการจำแนกความคิดเห็นที่เกี่ยวข้องกับความมั่นคงสูงที่สุด โดยมีค่าความถูกต้องร้อยละ 89.84 รองลงมาคือ Support Vector Machine มีค่าความถูกต้องร้อยละ 86.99 และ Logistic Regression ที่แม้จะมีค่าความถูกต้องร้อยละ 86.99 เท่ากันกับ Support Vector Machine แต่มีค่าความแม่นยำ (Precision) และค่าความถ่วงดุล (F-Measure) ต่ำกว่า Support Vector Machine

บทที่ 3
วิธีดำเนินการวิจัย

ในบทนี้ได้นำเสนอวิธีการดำเนินงานวิจัยซึ่งประกอบด้วยขั้นตอนการวิจัย 4 ส่วนดังแสดงในรูปที่ 3.1 แบ่งออกเป็น 4 ส่วนหลักได้แก่ สีเขียวเป็นการเก็บรวบรวมข้อมูลในการวิจัยนี้มีการเก็บข้อมูลทั้งหมด 3 รูปแบบต่อหุ้น 1 ตัว สีแดงเป็นส่วนของการเตรียมข้อมูลมีการใช้ตัดคำ 2 แบบคือ PythaiNLP และ DeepCut ในงานวิจัยนี้ ส่วนต่อมาคือสีฟ้าเป็นส่วนของการจำแนกความรู้สึกข่าวหุ้นรายวันในส่วนนี้จะมีทั้งหมด 10 แบบจำลองที่นำมาช่วยในการหาค่าที่ดีที่สุดในการทำแบบจำลอง และสีเหลืองเป็นส่วนของการประเมินผล



รูปที่ 3.1 ขั้นตอนการดำเนินงานวิจัย

3.1 การเก็บข้อมูลจากเว็บไซต์

งานวิจัยนี้ได้ทำการรวบรวมข้อมูลจากแหล่งที่มา 2 ที่คือ หัวข้อข่าวจากเว็บไซต์ข่าวหุ้นตั้งแต่เดือนมกราคม พ.ศ. 2561 จนถึงเดือนธันวาคม พ.ศ. 2565 ด้วยการใช้ภาษา Python บน Google Colab ซึ่งเป็นเครื่องมือพัฒนาโปรแกรมที่ทำงานบนคลาวด์ (Cloud) ของ Google

รูปที่ 3.2 Google Colaboratory หรือ Google Colab

ในวิจัยนี้ทำการดึงข้อมูลมาจาก 2 แหล่งข้อมูลคือ

1. ข้อความหัวข้อข่าวเพื่อใช้ในการจำแนกความรู้สึก
2. ราคาปิดรายวันของหุ้นทั้ง 2 ตัว

โดยมีวิธีในการเก็บข้อมูลดังนี้

3.1.1 วิธีการเก็บข้อความหัวข้อข่าวเพื่อใช้ในการจำแนกความรู้สึก

วิธีการเก็บข้อความหัวข้อข่าวเพื่อใช้ในการจำแนกความรู้สึก จะทำการเก็บข้อมูลด้วยเครื่องมือ Beautiful Soup ซึ่งเป็นเครื่องมือที่ใช้ในสำหรับดึงข้อมูลหน้าเว็บ หรือ HTML

```
page = 1
title_list=[]
date_list=[]
while page <= 1326:
  data = requests.get('https://www.kaohoon.com/page/'+str(page)+'?s=อสังหา')
  soup = bs4.BeautifulSoup(data.text)
  for c in soup.find_all('div',{'class':'post-details'}):
    title_list.append(c.find('h2',{'class':'post-title'}).find('a').text)
    date_list.append(c.find('span',{'class':'date meta-item tie-icon'}).text)
  print("complete page:" , page)
  page += 1

table = pd.DataFrame([title_list,date_list]).transpose()
table.columns = ['title','date']
table.set_index('title')
```

รูปที่ 3.3 คำสั่งในการดึงข้อมูลจาก เว็บข่าวหุ้น ด้วย Beautiful Soup

จากรูปที่ 3.3 เป็นการสั่งงานให้ทำการดึง ข้อมูลจากเว็บ www.kaohoon.com/page การดึงข้อมูลจะใช้คีย์เวิร์ดของหุ้นแต่ละตัว เป็นคำค้นในการดึงข้อมูลหัวข้อข่าวมาเก็บไว้ใช้จำแนกความรู้สึกจากหัวข้อข่าวต่อไป ในงานวิจัยนี้จะมีการเก็บข้อมูลอยู่ทั้งหมด 3 รูปแบบต่อหุ้น 1 ตัว

รูปแบบการเก็บข้อมูล

แบบ A คือ การเก็บข้อมูลหัวข้อข่าว โดยใช้ ชื่อย่อหุ้น เท่านั้น

แบบ B คือ การเก็บข้อมูลหัวข้อข่าว โดยใช้ ชื่อย่อหุ้น และ คำที่เกี่ยวข้องกับธุรกิจของหุ้น

แบบ C คือ การเก็บข้อมูลหัวข้อข่าว โดยใช้ ชื่อย่อหุ้น , คำที่เกี่ยวข้องกับธุรกิจของหุ้น และหัวข้อข่าวเศรษฐกิจภายใน-นอกประเทศ

เมื่อเก็บข้อมูลตามรูปแบบดังกล่าวจะได้ข้อมูลจำนวนหัวข้อข่าวตามตารางที่ 3.1 และ 3.2 ดังนี้

ตารางที่ 3.1 รูปแบบที่ใช้ดึงข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH)

รูปแบบ	คีย์เวิร์ดที่ใช้ดึงข้อมูลหัวข้อข่าว	จำนวน
A	“RATCH”	317 หัวข้อข่าว
B	“RATCH”, “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล”, “โรงไฟฟ้า”	1,501 หัวข้อข่าว
C	“RATCH”, “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล”, “โรงไฟฟ้า”, “เศรษฐกิจภายในประเทศ”, “เศรษฐกิจนอกประเทศ”	6,904 หัวข้อข่าว

ตารางที่ 3.2 รูปแบบที่ใช้ดึงข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH)

รูปแบบ	คีย์เวิร์ดที่ใช้ดึงข้อมูลหัวข้อข่าว	จำนวน
A	“QH”	65 หัวข้อข่าว
B	“QH”, “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์”	837 หัวข้อข่าว
C	“QH”, “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์”, “เศรษฐกิจภายในประเทศ”, “เศรษฐกิจนอกประเทศ”	6,240 หัวข้อข่าว

ตารางที่ 3.3 ตัวอย่างข้อความที่รวบรวมจากเว็บข่าวหุ้น

ข้อความหัวข้อข่าว	วันที่
ยกเลิกบริษัทร่วมทุนกัมพูชา ยันไม่กระทบการดำเนินงาน	25/01/2018
อัดฉีดงบฯปีนี้หมื่นลบ. ลุยเปิดดีล M&A โรงไฟฟ้าใน-ตปท. หวังเพิ่มกำลังผลิตอีก 750MW	08/02/2018
กำไรปี 60 ขยับลงเล็กน้อยเหลือ 6.11 พันลบ. เหตุรายได้ลด-ต้นทุนขายพุ่ง	14/02/2018
ควักเงิน 1.3 พันลบ.ซื้อหุ้น “RAC” เพิ่ม หวังต่อยอดธุรกิจโรงไฟฟ้าในออสเตรเลีย	23/04/2018

3.1.2 วิธีการเก็บข้อมูลราคาปิดรายวันของหุ้นทั้ง 2 ตัว

การเก็บข้อมูลอีกส่วนคือข้อมูลพื้นฐานในการซื้อขายหุ้นรายวัน โดยเก็บข้อมูลจากเว็บไซต์ยาฮูไฟแนนซ์ตั้งแต่เดือนมกราคม พ.ศ. 2561 จนถึงเดือนธันวาคม พ.ศ. 2565 ด้วยการใช้ภาษา Python บน Google Colab ซึ่งเป็นเครื่องมือพัฒนาโปรแกรมที่ทำงานบนคลาวด์ (Cloud) ของ Google ทำ

การเก็บข้อมูลด้วยเครื่องมือ Yahoo Finance ซึ่งเป็นชุดคำสั่งสำเร็จรูปที่ช่วยให้เราสามารถดึงข้อมูลสาธารณะจาก Yahoo Finance มาได้

```
[ ] !pip install yfinance

import yfinance as yf
data = yf.download("RATCH.BK", start="2018-01-01", end="2022-12-31")

[ ] data.to_csv("RATCH.csv")
```

รูปที่ 3.4 คำสั่งในการดึงข้อมูลจาก Yahoo Finance

จากรูปที่ 3.4 เป็นการสั่งงานให้ทำการดึง ข้อมูลจากเว็บ https://finance.yahoo.com/ จะได้ข้อมูลดังรูปที่ 3.5

Date	Open	High	Low	Close	Adj Close	Volume
2018-01-03	54.25	55.25	54.25	54.5	43.3539924621582	3552700
2018-01-04	54.75	54.75	54.25	54.5	43.3539924621582	1702800
2018-01-05	54.75	55.75	54.25	54.5	43.3539924621582	4915200
2018-01-08	54.75	54.75	54.25	54.5	43.3539924621582	948400
2018-01-09	54.5	54.5	54.5	54.5	43.3539924621582	171900
2018-01-10	55.0	55.5	54.75	55.0	43.751739501953125	2671800
2018-01-11	55.25	55.75	54.75	55.25	43.950599670410156	2613500
2018-01-12	55.5	55.5	55.25	55.5	44.149478912353516	1795200
2018-01-15	55.5	56.25	55.25	55.5	44.149478912353516	4021900
2018-01-16	55.5	56.0	55.25	55.25	43.950599670410156	1653800

รูปที่ 3.5 ตัวอย่างข้อมูลจาก Yahoo Finance

เมื่อได้ข้อมูลทั้งสองส่วนแล้วก็เอาวันที่ของข้อมูลทั้งสองมาเชื่อมต่อกันจะได้ข้อมูลดังรูปที่ 3.6

Date	open	high	low	close	adj close	volume	Taxt
25/01/2018	55.5	55.5	55.25	55.25	44.8770943	806400	RATCH ยกเลิกบริษัทร่วมทุนกัมพูชา ยันไม่กระทบการดำเนินงาน
08/02/2018	54.25	54.5	54	54.25	44.0648422	708000	RATCH อัดฉีดงบ4ปีนี้หมื่นลบ. ลุยเปิดดีล M&A โรงไฟฟ้าใน-ตปท. หวังเพิ่มกำลังผลิตอีก 750MW
14/02/2018	54	54.25	53.75	54	43.8617744	1142100	RATCH กำไรปี 60 ขยับลงเล็กน้อยเหลือ 6.11 พันลบ. เหตุรายได้ลด-ต้นทุนขายพุ่ง
23/04/2018	52	52.25	52	52	43.149929	893300	RATCH ครึกเงิน 1.3 พันลบ.ซื้อหุ้น "RAC" เพิ่ม หวังต่อยอดธุรกิจโรงไฟฟ้าในออสเตรเลีย
15/05/2018	52	52.25	51.75	51.75	42.9424706	5130300	RATCH รายได้ลด-ขาดทุนอัตราแลกเปลี่ยนกระทบกำไร Q1/61 วูบ 40%

รูปที่ 3.6 ตัวอย่างข้อมูลที่เชื่อมต่อกันแล้วผ่านวันที่

เมื่อได้ข้อมูลทั้ง 2 ส่วนมาแล้วก็จะเข้าสู่กระบวนการสร้างคำตอบเพื่อใช้ในการฝึกสอนแบบจำลองประเภทการจำแนกในหัวข้อถัดไป

3.1.3 การสร้างคำตอบเพื่อใช้ในการฝึกสอนแบบจำลองประเภทการจำแนก

ก่อนที่จะนำข้อมูลไปใช้ในการจำแนกได้ต้องมีการสร้างคำตอบให้ชุดข้อมูลก่อนโดยกำหนดให้คำตอบของแต่ละหัวข้อข่าวมาจากการหาค่าความแตกต่างของราคาหุ้น โดยจากรูปที่ 3.6 ข้อมูลที่ได้มีการเก็บมาจะมีวันที่และราคาปิด (วิกานดา และ อัญชนา, 2563) เพื่อใช้กำหนดสถานะของราคาปิดตลาดในแต่ละวันจากสมการ

$$Diff = Closeprice_i - Closeprice_{i-1}$$

กำหนดให้ $Diff$ คือค่าความแตกต่างของราคาหุ้น

$Closeprice_i$ คือราคาปิดของหุ้นวันที่ i

$Closeprice_{i-1}$ คือราคาปิดของหุ้นวันที่ i -1

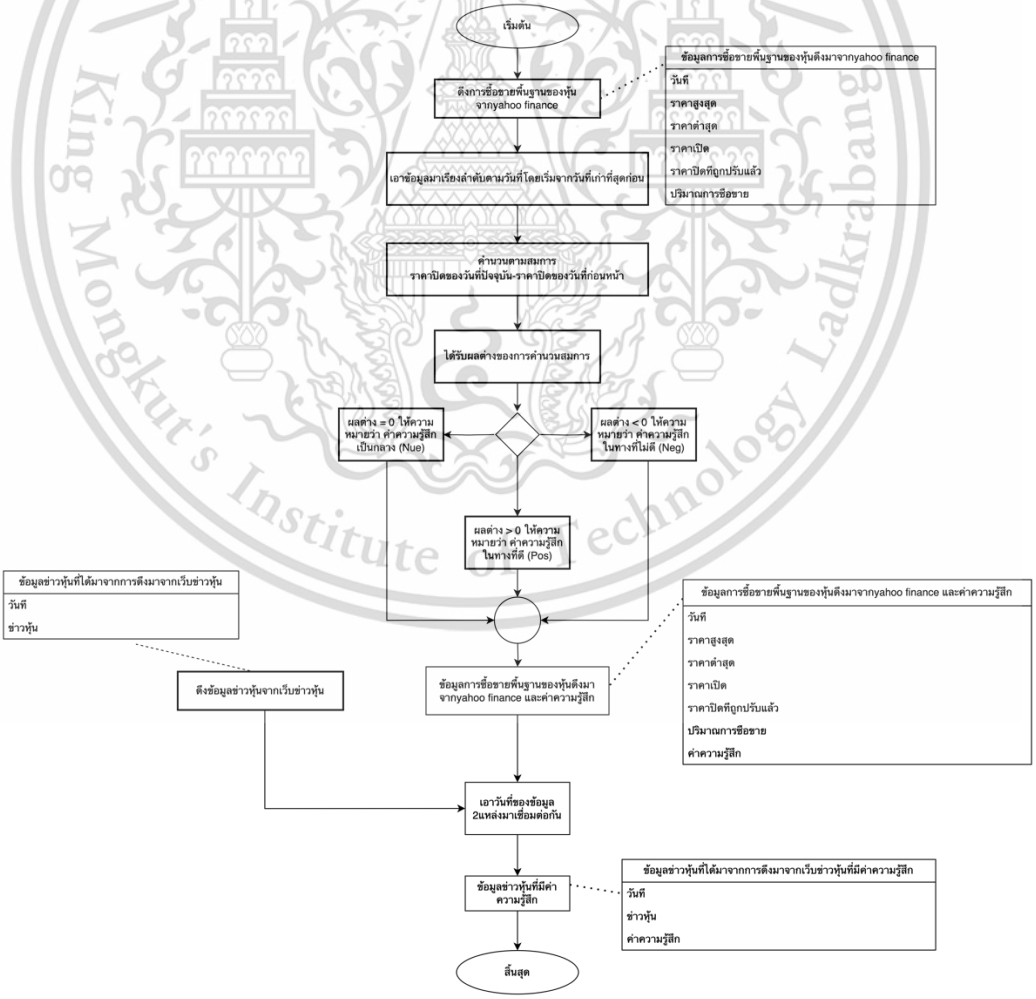
จากนั้นนำค่า $Diff$ มาหาสถานะโดยกำหนดให้

$Diff > 0$ หมายถึง ราคาหุ้นมีแนวโน้มเป็นบวก (Positive)

$Diff < 0$ หมายถึงราคาหุ้นมีแนวโน้มเป็นลบ (Negative)

$Diff = 0$ หมายถึงราคาหุ้นมีแนวโน้มเป็นกลาง (Neutral)

โดยในกระบวนการนี้จะทำการจำแนกว่าแต่ละหัวข้อความข่าวนั้นเป็นลักษณะข้อความที่เป็น ในทางมีแนวโน้มเป็นบวกแนวโน้มเป็นลบหรือแนวโน้มเป็นกลาง โดยผ่านขั้นตอนต่างๆ ตามรูปที่ 3.7 และแสดงตัวอย่างการกำหนดสถานะการจำแนกจากราคาปิดตามตารางที่ 3.4



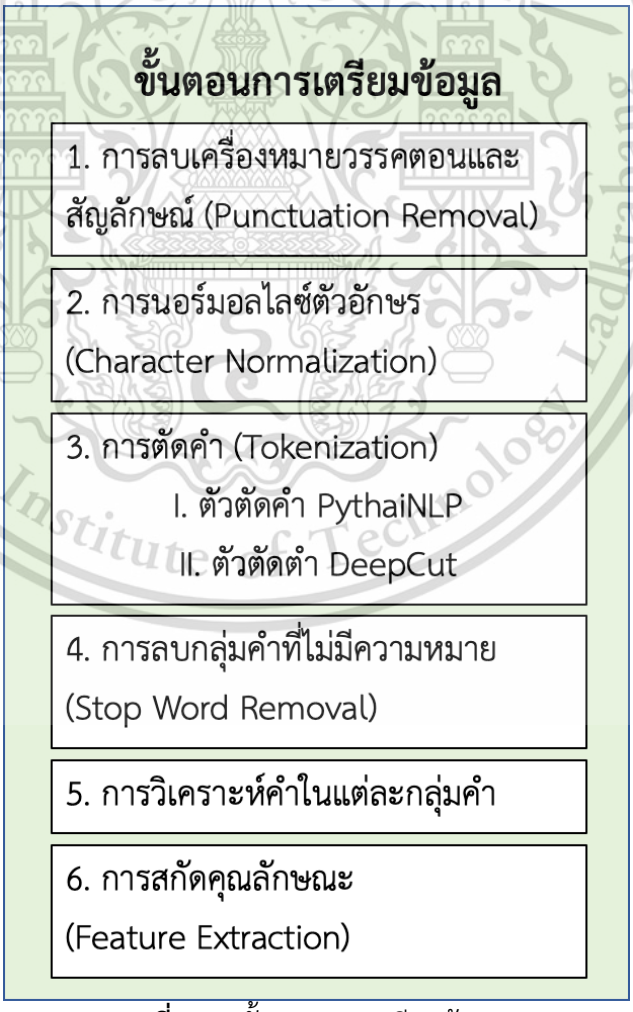
รูปที่ 3.7 ขั้นตอนการจำแนกความรู้สึกจากข่าว

ตารางที่ 3.4 ตัวอย่างข้อมูลสถานะการจำแนกจากราคาปิด

Date	open	high	low	close	adj close	volume	Sentiment	หัวข้อข่าว
23/07/2018	52.25	52.25	51.75	52	43.149929	1413600	Nue	-
24/07/2018	51.75	52.25	51	51.25	42.5275688	4753800	Neg	เขื่อนผลิตไฟฟ้า"เซเปียน-เซน้ำน้อย" สปป.ลาวแตก จับตาเปิดบ่ายหุ้น RATCH รูดหนัก!
25/07/2018	50.75	51.5	50.25	51.5	42.7350235	5827000	Pos	โบรกฯชี้ "เขื่อนแตก" ฉุด RATCH กำลังผลิตหด-กระทบราคาหุ้น 1 บ. จ่อชดใช้ค่าเสียหายอื้อ

3.2 การเตรียมข้อมูล

หลังจากทำการเก็บข้อมูลจากเว็ปไซต์เรียบร้อยแล้วจากหัวข้อที่ 3.1 ข้อมูลรูปแบบของข้อความนั้นไม่สามารถนำไปใช้ในการวิเคราะห์ได้ทันทีจะต้องมีกระบวนการในการเตรียมข้อมูล โดยการเตรียมข้อมูลส่วนหัวข้อข่าวคือการนำข้อความหัวข้อข่าวมาเข้าสู่ขั้นตอนการแปลงข้อมูลต่างๆ ให้มีรูปแบบเป็นเวกเตอร์โดยมีกระบวนการแบ่งเป็น 5 ขั้นตอนแสดงดังรูปที่ 3.7



รูปที่ 3.8 ขั้นตอนการเตรียมข้อมูล

ตัวอย่างการเตรียมข้อมูลในการเตรียมข้อมูลนี้ ประโยคที่ใช้คือประโยค “ไตรมาส1โชว์กำไร585ล้านบาทโต36%ยอดขายบ้านเพิ่ม-โรงแรมฟื้น” ซึ่งจะต้องผ่านขั้นตอนการเตรียมข้อมูลทั้ง 5 ขั้นตอน ได้แก่ การลบเครื่องหมายวรรคตอน สัญลักษณ์ และ ตัวเลข การนอร์มอลไลซ์ตัวอักษร การตัดคำ โดยในส่วนการตัดคำนั้นจะมีการใช้ตัวตัดคำ 2 แบบ เพื่อนำผลที่ได้มาใช้ในการเปรียบเทียบต่อไปโดยการตัดคำจะแบ่งได้ดังนี้

แบบ I ใช้ PythaiNLP ในการตัดคำข้อความหัวข้อข่าว

แบบ II ใช้ DeepCut ในการตัดคำข้อความหัวข้อข่าว

ต่อมาก็จะเป็นการลบกลุ่มคำที่ไม่มีความหมาย และ การสกัดคุณลักษณะ ซึ่งจากประโยคตัวอย่างดังกล่าวเมื่อ นำมาผ่านขั้นตอนการเตรียมข้อทั้ง 5 ขั้นตอนดังนี้

3.2.1 การลบเครื่องหมายวรรคตอนและสัญลักษณ์ออก (Punctuation Removal)

จากตัวอย่างข้อมูลหัวข้อข่าวประโยค “ไตรมาส1โชว์กำไร585ล้านบาทโต36%ยอดขายบ้านเพิ่ม-โรงแรมฟื้น” จะเห็นได้ว่าภายในข้อความที่นั้นมีสิ่งไม่พึงประสงค์รวมอยู่ด้วย เช่นอักขระพิเศษหรือเครื่องหมายต่าง ๆ มีถึงมีลิงก์ (Link) และตัวเลขต่าง ๆ ด้วยเหตุนี้ผู้วิจัยจึงทำการลบสิ่งไม่พึงประสงค์เหล่านั้นออกไป เพื่อให้ได้ข้อความเพียงอย่างเดียว แสดงตัวอย่างผลลัพธ์ที่ได้จากการทำความสะอาดข้อมูลดังตารางที่ 3.5

ตารางที่ 3.5 ตัวอย่างขั้นตอนการการลบเครื่องหมายวรรคตอนสัญลักษณ์และตัวเลขออก

ขั้นตอน	คำอธิบาย	ผล
การลบเครื่องหมายวรรคตอนสัญลักษณ์และตัวเลขออก	ลบเลข “1”, “585”, “36” และเครื่องหมาย “%” และ “-” ออก	ไตรมาสโชว์กำไรล้านบาทโตยอดขายบ้านเพิ่มโรงแรมฟื้น

3.2.2 การนอล์มอลไลซ์ตัวอักษร (Character Normalization)

การนอล์มอลไลซ์ตัวอักษรคือการแปลงคำต่างๆ ให้กลายเป็นตัวอักษรในแบบเดียวกันเพื่อความสะดวกในการวิเคราะห์ เช่นการแปลงตัวอักษรตัวเล็กในภาษาอังกฤษ เนื่องจากคอมพิวเตอร์จะมองตัวอักษร์เล็กใหญ่เป็นคนละตัวกันจากประโยค “ไตรมาสโชว์กำไรล้านบาทโตยอดขายบ้านเพิ่มโรงแรมฟื้น” ที่ได้มาจากการขั้นตอนการการลบเครื่องหมายวรรคตอนสัญลักษณ์และตัวเลขออกจะเห็นได้ว่าภายในข้อความที่นั้นไม่มีตัวอักษรภาษาอังกฤษอยู่เมื่อแปลงมาจึงได้ประโยคเหมือนเดิม แสดงตัวอย่างผลลัพธ์ที่ได้จากการนอล์มอลไลซ์ตัวอักษรดังตารางที่ 3.6

ตารางที่ 3.6 ตัวอย่างการนอล์มอลไลซ์ตัวอักษร

ขั้นตอน	คำอธิบาย	ผล
การนอร์มอลไลซ์ตัวอักษร	ทำให้อยู่ในรูปแบบเดียวกัน	ไตรมาสโชว์กำไรล้านบาทโตยอดขายบ้านเพิ่มโรงแรมฟื้น

3.2.3 การตัดคำ (Tokenization)

การนำประโยคมาแบ่งออกเป็นคำต่างๆ ในงานวิจัยนี้จะใช้ตัวตัดคำ 2 แบบคือ แบบ I ใช้ PythaiNLP ในการตัดคำข้อความหัวข้อข่าว แบบ II ใช้ DeepCut ในการตัดคำข้อความหัวข้อข่าว จากประโยค “ไตรมาสโชว์กำไรล้านบาทโตยอดขายบ้านเพิ่มโรงแรมฟื้น” ที่ได้มาจากการขั้นตอนการนอล์มอลไลซ์ตัวอักษรเมื่อนำมาตัดคำแล้ว แสดงตัวอย่างผลลัพธ์ที่ได้จากการตัดคำดังตารางที่ 3.7

ตารางที่ 3.7 ตัวอย่างการเปรียบเทียบผลของตัวตัดคำ

ตัวตัดคำ	ผล
การตัดคำด้วย PythaiNLP	ไตรมาส โชว์ กำไร ล้าน บาท โต ยอดขาย บ้าน เพิ่ม โรงแรม ฟื้น
การตัดคำด้วย Deepcut	ไตรมาส โชว์ กำไร ล้าน บาท โตยอด ขาย บ้าน เพิ่ม โรง แรม ฟื้น

3.2.4 การกำจัดคำหยุด (Stop Word Removal)

การกำจัดคำหยุด (Stop Word Removal) เป็นการลบคำคุณลักษณะหรือคำที่ไม่มีนัยสำคัญในข้อความออกไป โดยที่ความหมายของข้อความนั้นจะไม่เปลี่ยนแปลงไปจากเดิมเพื่อลดความซับซ้อนของข้อมูลลง แสดงตัวอย่างผลลัพธ์ที่ได้จากการลบคำหยุดดังตารางที่ 3.8

ตารางที่ 3.8 ตัวอย่างการกำจัดคำหยุด

ขั้นตอน	คำอธิบาย	ผล
การลบกลุ่มคำที่ไม่มีความหมาย	ลบคำว่า “โต” เพราะจะมีหรือไม่ความหมายของประโยคยังคงเดิม	ไตรมาส โชว์ กำไร ล้าน บาท ยอดขาย บ้าน โรงแรม ฟื้น

3.2.5 การวิเคราะห์คำในแต่ละกลุ่ม (Analysis Word in Group)

การวิเคราะห์คำในแต่ละกลุ่มโดยดูจากความถี่ของคำด้วย Word Cloud หมายถึงการสร้างภาพที่แสดงถึงความถี่ของคำที่ปรากฏในกลุ่มข้อมูล โดยคำที่มีความถี่มากจะปรากฏในภาพด้วยขนาดที่ใหญ่กว่า และคำที่มีความถี่น้อยจะปรากฏด้วยขนาดที่เล็กกว่า วิธีนี้ช่วยให้สามารถมองเห็นคำที่สำคัญและบ่อยในข้อมูลได้ง่ายขึ้น ได้มองเห็นภาพรวมของคำที่มีความสำคัญและพบมากในกลุ่มข้อมูล (Heimerl et al., 2014)

3.2.6 การสกัดคุณลักษณะ (Feature Extraction)

การสกัดคุณลักษณะ (Feature Extraction) เป็นกระบวนการแปลงข้อมูลให้อยู่ในรูปแบบที่คอมพิวเตอร์สามารถประมวลผลได้เรียกว่าการทำดัชนี (Indexing) เพื่อสร้างตัวแทนเอกสาร (Document Terms Matrix) ที่อยู่ในรูปของเวกเตอร์ของน้ำหนักคำ (Term Weighting) ซึ่งการทำดัชนี (Indexing) แสดงตัวอย่างตัวแทนเอกสารที่ได้จากการสกัดคุณลักษณะ (Feature Extraction) ดังตารางที่ 3.9

ตารางที่ 3.9 ตัวอย่างตัวแทนเอกสารที่ได้จากการสกัดคุณลักษณะ (Feature Extraction)

ข้อความ	กำไร	ก๊าซชีวภาพ	ก๊าซธรรมชาติ	ขาย	ตั้งเป้า	ถ่านหิน
ขาย หุ้นกู้ พัน ล้าน บาท เกลี้ยง ลุย ขยาย โรงไฟฟ้า ตั้งเป้า ปี แตะ พัน เมกะ วัตต์	0	0	0	0.226599	0.308631	0
ไตรมาส โชว์ กำไร ล้าน บาท โตยอด ขาย บ้าน เพิ่ม โรง แรม ฟื้น	0	0	0	0	0	0
จ่อ โรงไฟฟ้า ก๊าซชีวภาพ เดือน ดัน รายได้ ปี	0	0.593998	0	0	0	0
กำไร แตะ ล้าน บาท ปีก่อน ล้าน บาท ธุรกิจ ถ่านหิน เอทานอล	0.214927	0	0	0	0	0.221682
ไลเซนส์ นำเข้า แสน ตัน ปี ป้อน โรงไฟฟ้า ในเครือ	0	0	0	0	0	0

3.3 การจำแนกความรู้สึกจากข่าวหุ้นรายวัน

หลังจากได้ที่ได้ทำการสกัดคุณลักษณะและแปลงหัวข้อความเป็นเวกเตอร์ด้วยภาษาไพธอนแล้วในขั้นตอนก่อนหน้า ข้อมูลเหล่านี้จะถูกนำมาเข้ากระบวนการฝึกฝนข้อมูลด้วยอัลกอริทึมการจำแนกประเภททั้งหมด 10 ชนิด คือ การถดถอยเชิงโลจิสติค (Logistic Regression) นาอีฟเบย์ (Naïve Bayes) การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ป่าสุ่ม (Random Forest) ต้นไม้ตัดสินใจ (Decision Tree) ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) เพอร์เซปตรอน (Perceptron) สโตแคสติกการ์เดียนดิเซนท์ (SGD) พาสซีฟ อากัสซีฟ (Passive Aggressive) โดยใช้ไลบรารี่ Sklearn ซึ่งเป็นไลบารี่สำหรับสร้างโมเดลการเรียนรู้ของเครื่องในภาษาไพธอน ซึ่งหลังจากข้อมูลผ่านกระบวนการเตรียมข้อมูลในหัวข้อที่ 3.2 แล้วข้อมูลจะมีการแบ่งชุดข้อมูลเป็น ชุดข้อมูลเรียนรู้ 70% ชุดข้อมูลทดสอบ 30% การแบ่งข้อมูลแบบนี้มีข้อดีหลายประการโดยเฉพาะกับข้อมูลขนาดเล็ก (Jason., 2020) แบ่งโดยการสุ่ม หลังจากนั้นจะถูกนำไปสร้างโมเดล

3.3.1 การถดถอยเชิงโลจิสติค (Logistic Regression)

การสร้างการถดถอยเชิงเส้นโลจิสติคจะใช้หนึ่งในไลบรารี่ย่อยของ Sklearn คือ Sklearn.linear_model.LogisticRegression ที่ใช้สำหรับโมเดลเชิงเส้นและในงานวิจัยนี้มีการปรับจูนค่า C ด้วย ซึ่งพารามิเตอร์ C ใน Logistic Regression เป็นพารามิเตอร์ที่ใช้ในการควบคุมการ

แพร่กระจาย (Regularization) ของโมเดล Regularization ช่วยในการป้องกันปัญหา Overfitting การหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับ Logistic Regression สามารถทำได้โดยใช้ GridSearchCV ซึ่งเป็นเครื่องมือในไลบรารี Sklearn ที่ช่วยให้เราค้นหาค่าพารามิเตอร์ที่ดีที่สุดโดยการทดลองค่าในช่วงที่กำหนดและประเมินผลโดยใช้วิธีการ Cross-validation ในขั้นตอนการปรับจูนค่า C ด้วย GridSearchCV เราจะกำหนดช่วงของค่าพารามิเตอร์ C ที่ต้องการทดลองที่ [0.1, 1, 10, 100] จากนั้น GridSearchCV จะทำการฝึกโมเดลสำหรับแต่ละค่าของ C และประเมินผลโดยใช้วิธีการ Cross-validation หลังจากนั้นจะเลือกค่าพารามิเตอร์ที่ให้ผลดีที่สุด

3.3.2 นาอีฟเบย์ (Naïve Bayes)

การสร้างนาอีฟเบย์จะใช้ไลบรารี Sklearn.naive_bayes ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลนาอีฟเบย์สำหรับงานประเภทการจำแนก (Classification) และในงานวิจัยนี้มีการปรับจูนค่า alpha ด้วย ซึ่งพารามิเตอร์ alpha ในนาอีฟเบย์เป็นพารามิเตอร์ที่ช่วยในการแก้ปัญหาข้อมูลที่ไม่มีในชุดฝึก (Training set) โดยการเพิ่มค่าเล็กน้อยเข้าไปในจำนวนการนับของแต่ละลักษณะ (Feature) เพื่อไม่ให้ค่าความน่าจะเป็นเป็นศูนย์ ค่า alpha ที่เหมาะสมจะช่วยเพิ่มความแม่นยำของโมเดลในการทำนายผล การหาค่าพารามิเตอร์ alpha ที่เหมาะสมที่สุดสำหรับ นาอีฟเบย์สามารถทำได้โดยใช้ GridSearchCV ซึ่งเป็นเครื่องมือในไลบรารี Sklearn ที่ช่วยให้เราค้นหาค่าพารามิเตอร์ที่ดีที่สุดโดยการทดลองค่าในช่วงที่กำหนดและประเมินผลโดยใช้วิธีการ Cross-validation ในขั้นตอนการปรับจูนค่า alpha ด้วย GridSearchCV จะมีการกำหนดช่วงของค่าพารามิเตอร์ alpha ที่ต้องการทดลองที่ [0.0001, 0.001, 0.01, 1] จากนั้น GridSearchCV จะทำการฝึกโมเดลสำหรับแต่ละค่าของ alpha และประเมินผลโดยใช้วิธีการ Cross-validation หลังจากนั้นจะเลือกค่าพารามิเตอร์ที่ให้ผลดีที่สุด

3.3.3 การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors)

การสร้าง K-Nearest Neighbors จะใช้ไลบรารี Sklearn.neighbors ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดล K-Nearest Neighbors และในงานวิจัยนี้มีการปรับจูนค่าพารามิเตอร์ที่สำคัญที่สุดของ K-Nearest Neighbors คือจำนวนของเพื่อนบ้าน (n_neighbors) ที่เหมาะสมที่สุดสำหรับ K-Nearest Neighbors สามารถทำได้โดยใช้ GridSearchCV ซึ่งเป็นเครื่องมือในไลบรารี Sklearn ที่ช่วยให้เราค้นหาค่าพารามิเตอร์ที่ดีที่สุดโดยการทดลองค่าในช่วงที่กำหนดและประเมินผลโดยใช้วิธีการ Cross-validation การใช้ GridSearchCV กับ K-Nearest Neighbors จะทำการปรับจูนค่าของ n_neighbors โดยพิจารณาค่าที่แตกต่างกันที่ [3, 5, 7] ซึ่งเป็นค่าที่นิยมนำมาใช้ในการทดสอบ

3.3.4 ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines)

การสร้างซัพพอร์ตเวกเตอร์แมชชีนจะใช้ไลบรารี Sklearn.svm ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลซัพพอร์ตเวกเตอร์แมชชีนสำหรับงานประเภทการจำแนก (Classification) และในงานวิจัยนี้มีการปรับจูนค่าพารามิเตอร์ของซัพพอร์ตเวกเตอร์แมชชีนโดยใช้

GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดล โดยการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด พิจารณาค่าพารามิเตอร์โดยจะมี C พารามิเตอร์ที่ควบคุม trade-off ระหว่างการมีขอบเขตที่กว้างกับการจัดกลุ่มข้อมูลอย่างถูกต้อง ค่าที่ต่ำจะทำให้ขอบเขตกว้างขึ้นแต่มีการยอมให้ข้อมูลบางจุดอยู่ผิดกลุ่ม ในขณะที่ค่าที่สูงจะพยายามจัดกลุ่มข้อมูลให้ถูกต้องมากที่สุดในที่นี้ใช้ค่า [0.1, 1, 10] kernel ฟังก์ชันที่ใช้แปลงข้อมูลให้อยู่ในมิติที่สูงขึ้น มี kernel หลายประเภทคือ linear rbf (Radial Basis Function) poly และค่า gamma พารามิเตอร์ที่ใช้กับ kernel rbf เพื่อกำหนดรูปทรงของพื้นที่ในมิติสูงที่ใช้แบ่งข้อมูลช่วงค่าพารามิเตอร์ที่ใส่เข้าไปคือ scale และ auto

3.3.5 ป่าสุ่ม (Random Forest)

การสร้างป่าสุ่มจะใช้ไลบรารี Sklearn.ensemble.RandomForestClassifier ซึ่งเป็นหนึ่งในไลบรารีย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลสุ่มป่าไม้สำหรับงานประเภทการจำแนก (Classification) โดยในงานวิจัยนี้จะมีการปรับจูนค่าพารามิเตอร์ของป่าสุ่มโดยใช้ GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดล โดยการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด สำหรับการปรับจูนป่าสุ่มจะพิจารณาค่าพารามิเตอร์ดังนี้ n_estimators: [100, 200, 300] และmax_depth: [None, 10, 20]

3.3.6 ต้นไม้ตัดสินใจ (Decision Tree)

การสร้างต้นไม้ตัดสินใจจะใช้ไลบรารี Sklearn.tree.DecisionTreeClassifier ซึ่งเป็นหนึ่งในไลบรารีย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลต้นไม้ตัดสินใจสำหรับงานประเภทการจำแนก (Classification) โดยในงานวิจัยนี้จะมีการปรับจูนค่าพารามิเตอร์ของต้นไม้ตัดสินใจไม้โดยใช้ GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดล โดยการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด สำหรับการปรับจูนต้นไม้ตัดสินใจเราจะพิจารณาค่าพารามิเตอร์ดังนี้ max_depth: [None, 10, 20] โดย max_depth คือความลึกสูงสุดของต้นไม้ตัดสินใจ การกำหนดค่าให้ None จะทำให้ต้นไม้โตจนกว่าทุกใบจะบริสุทธิ์ หรือจนกว่าจะมีตัวอย่างน้อยกว่าค่าขั้นต่ำ การจำกัดความลึกของต้นไม้สามารถช่วยลดความซับซ้อนและป้องกันการ Overfitting

3.3.7 ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP)

การสร้างตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (Multi-Layer Perceptron Classifier) เป็นอัลกอริทึมการเรียนรู้ด้วยเครื่องแบบเครือข่ายประสาทเทียม (Neural Network) ที่ใช้งานง่ายและมีประสิทธิภาพสูงในการจำแนก (Classification) MLP เป็นเครือข่ายประสาทที่ประกอบด้วยชั้นข้อมูลเข้า (Input Layer) ชั้นข้อมูลซ่อน (Hidden Layers) และชั้นข้อมูลออก (Output Layer) การใช้งาน MLPClassifier จากไลบรารี Sklearn ของ Python และวิธีการปรับจูนพารามิเตอร์ด้วย GridSearchCV โดยพารามิเตอร์ที่มีการปรับจูนคือ hidden_layer_sizes หรือขนาดของชั้นข้อมูลซ่อนในเครือข่ายประสาท พารามิเตอร์นี้กำหนดจำนวนและขนาดของชั้นข้อมูล

ซ่อนคือ (100,) หมายถึงมีชั้นข้อมูลซ่อน 1 ชั้นที่มี 100 หน่วยประสาท (Neurons) (50, 50) หมายถึงมีชั้นข้อมูลซ่อน 2 ชั้นที่แต่ละชั้นมี 50 หน่วยประสาทและ (50, 100) หมายถึงมีชั้นข้อมูลซ่อน 2 ชั้นที่แต่ละชั้นมี 50 กับ 100 หน่วยประสาท

3.3.8 เพอร์เซปตรอน (Perceptron)

การสร้างเพอร์เซปตรอนจะใช้ไลบรารี Sklearn.linear_model.Perceptron ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลเพอร์เซปตรอนสำหรับงานประเภทการจำแนก (Classification) โดยในงานวิจัยนี้จะมีการปรับจูนค่าพารามิเตอร์ของเพอร์เซปตรอนโดยใช้ GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดล โดยการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด สำหรับการปรับจูนเพอร์เซปตรอนจะพิจารณาค่าพารามิเตอร์ดังนี้ alpha: [0.0001, 0.001, 0.01] โดยค่า alpha คือพารามิเตอร์ที่ใช้ในการควบคุมอัตราการเรียนรู้ (Learning Rate) หรือค่าปรับแก้ของน้ำหนักในกระบวนการฝึกสอน ค่าที่สูงกว่าจะทำให้การเปลี่ยนแปลงของน้ำหนักใหญ่ขึ้น แต่ก็อาจทำให้การฝึกสอนไม่เสถียร ขณะที่ค่าที่ต่ำกว่าจะทำให้การเปลี่ยนแปลงของน้ำหนักเล็กลงและกระบวนการฝึกสอนเสถียรมากขึ้น

3.3.9 สโตแคสติกการ์เดียนดิเซนท์ (SGD)

การสร้างการลดความชันแบบสุ่มจะใช้ไลบรารี Sklearn.linear_model.SGDClassifier ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลการลดความชันแบบสุ่มสำหรับงานประเภทการจำแนก (Classification) โดยในงานวิจัยนี้จะมีการปรับจูนค่าพารามิเตอร์ของการลดความชันแบบสุ่มโดยใช้ GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดลจากการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด สำหรับการปรับจูนการลดความชันแบบสุ่มจะพิจารณาค่าพารามิเตอร์ดังนี้ alpha: [0.0001, 0.001, 0.01] โดยค่า alpha คือพารามิเตอร์การเรียนรู้ที่ใช้ในการป้องกันการ Overfitting การปรับน้ำหนัก ค่า alpha ที่ต่ำหมายถึงการลดการปรับน้ำหนักอย่างมาก ในขณะที่ค่า alpha ที่สูงหมายถึงการลดการปรับน้ำหนักน้อยลง

3.3.10 พาสซีฟ อากัสซีฟ (Passive Aggressive)

การสร้างพาสซีฟ อากัสซีฟ จะใช้ไลบรารี่ย่อยของ Sklearn คือ Sklearn.linear_model. PassiveAggressiveClassifier ที่ใช้สำหรับการสร้างโมเดลพาสซีฟ อากัสซีฟ โดยในงานวิจัยนี้จะมีการปรับจูนค่าพารามิเตอร์ของพาสซีฟ อากัสซีฟ โดยใช้ GridSearchCV เป็นเครื่องมือที่ช่วยในการค้นหาค่าพารามิเตอร์ที่เหมาะสมที่สุดสำหรับโมเดลจากการทดลองค่าต่าง ๆ และทำการ Cross-validation หลายครั้งเพื่อหาชุดค่าที่ให้ผลดีที่สุด สำหรับการปรับจูนการลดความชันแบบสุ่มจะพิจารณาค่าพารามิเตอร์ดังนี้ C: [0.1, 1, 10] โดยค่า C คือพารามิเตอร์ที่ใช้ในการควบคุมอัตราการอัปเดตของโมเดล ค่าที่ต่ำหมายถึงการอัปเดตที่น้อยและระมัดระวัง ในขณะที่ค่าที่สูงหมายถึงการอัปเดตที่มากและรวดเร็ว การเลือกค่า C ที่เหมาะสมเป็นสิ่งสำคัญเนื่องจากมีผลต่อประสิทธิภาพของโมเดล

3.3.11 สรุปผลการปรับค่าพารามิเตอร์

การปรับค่าพารามิเตอร์เป็นขั้นตอนสำคัญในการพัฒนาแบบจำลอง เพื่อให้ได้ประสิทธิภาพที่ดีที่สุดในงานวิจัยนี้มีการปรับค่าพารามิเตอร์โดยการใช้ GridSearchCV ที่เป็นเครื่องมือค้นหาค่าพารามิเตอร์ที่ดีที่สุดโดยการสำรวจค่าที่เป็นไปได้ทั้งหมดในช่วงที่กำหนดโดยมีการกำหนดช่วงค่าของแต่ละแบบจำลอง การแสดงจะแยกตามชุดข้อมูลส่วนแรกจะเป็นชุดข้อมูลของบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) การกำหนดช่วงค่าของแต่ละแบบจำลองตามตารางที่ 3.10 และแสดงค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลองในตารางที่ 3.11 ตามมาด้วย บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) แสดงการกำหนดช่วงค่าของแต่ละแบบจำลองตามตารางที่ 3.12 และค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลองในตารางที่ 3.13

ตารางที่ 3.10 พารามิเตอร์ที่นำมาใช้ในการตั้งค่าสำหรับเพิ่มประสิทธิภาพของแบบจำลอง (RATCH)

แบบจำลอง	พารามิเตอร์	คำอธิบาย	ช่วงค่าที่พิจารณา
Logistic Regression	C	Regularization parameter	0.1, 1, 10, 100
Naive Bayes	alpha		0.0001, 0.001, 0.01, 1
KNN	n_neighbors	Regularization parameter	3, 5, 7
SVM	kerner	Kernal	‘linear’, ‘poly’, ‘rbf’
	gamma	Kernal Coefficient	‘scale’, ‘auto’
	C	Regularization parameter	0.1, 1, 10
Random Forest	N	Number of Estimators	100, 200, 300
	MaxDepth		None, 10, 20
Decision Tree	MaxDepth		None, 10, 20
Neural network	hidden_layer_sizes		(100,), (50, 50), (50, 100)
Perceptron	alpha		0.0001, 0.001, 0.01
Stochastic Gradient Descent	alpha		0.0001, 0.001, 0.01
Passive Aggressive	C	Regularization parameter	0.1, 1, 10

ตารางที่ 3.11 ค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลอง (RATCH)

แบบจำลอง	ค่าที่ดีที่สุด
Logistic Regression	C = 100
Naive Bayes	alpha = 1
KNN	N_neighbors = 7
SVM	C = 1, Gamma = scale, kernel=rbf
Random Forest	max_depth=10, n_estimators=200
Decision Tree	max_depth=None
Neural network	hidden_layer_sizes=50, 100
Perceptron	alpha=0.0001
Stochastic Gradient Descent	alpha=0.0001
Passive Aggressive	C=10

ตารางที่ 3.12 พารามิเตอร์ที่นำมาใช้ในการตั้งค่าสำหรับเพิ่มประสิทธิภาพของแบบจำลอง (QH)

แบบจำลอง	พารามิเตอร์	คำอธิบาย	ช่วงค่าที่พิจารณา
Logistic Regression	C	Regularization parameter	0.1, 1, 10, 100
Naive Bayes	alpha		0.0001, 0.001, 0.01, 1
KNN	n_neighbors	Regularization parameter	3, 5, 7
SVM	kerner	Kernal	‘linear’, ‘poly’, ‘rbf’
	gamma	Kernal Coefficient	‘scale’, ‘auto’
	C	Regularization parameter	0.1, 1, 10
Random Forest	N	Number of Estimators	100, 200, 300
	MaxDepth		None, 10, 20
Decision Tree	MaxDepth		None, 10, 20
Neural network	hidden_layer_sizes		(100,), (50, 50), (50, 100)
Perceptron	alpha		0.0001, 0.001, 0.01
Stochastic Gradient Descent	alpha		0.0001, 0.001, 0.01
Passive Aggressive	C	Regularization parameter	0.1, 1, 10

ตารางที่ 3.13 ค่าพารามิเตอร์ที่ใช้เพิ่มประสิทธิภาพแบบจำลอง (QH)

แบบจำลอง	ค่าที่ดีที่สุด
Logistic Regression	C = 100
Naive Bayes	Alpha = 1.0
KNN	N_neighbors = 3
SVM	C = 10, Gamma = scale, kernel=linear
Random Forest	max_depth=20, n_estimators=300
Decision Tree	max_depth=None
Neural network	hidden_layer_sizes=50, 100
Perceptron	alpha=0.0001
Stochastic Gradient Descent	alpha=0.001
Passive Aggressive	C=1

3.4 การวัดประสิทธิภาพ

วิธีการประเมินและเปรียบประสิทธิภาพของแต่ละแบบจำลองนั้น จะใช้ตัววัดคือตาราง Confusion matrix เพื่อเปรียบเทียบข้อมูลจริงและข้อมูลที่เป็นผลลัพธ์จากการทำนายของแบบจำลอง โดยจะนำผลการทำนายมาคำนวณหาค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) และค่าความถ่วงดุล (F-Measure) จากนั้นจึงนำผลการคำนวณมาวิเคราะห์และทำการเปรียบเทียบประสิทธิภาพของแบบจำลอง เพื่อหาแบบจำลองที่ดีที่สุด

บทที่ 4
ผลการวิจัยและการอภิปรายผล

จากบทที่ 3 ที่ได้มีการนำเสนอขั้นตอนการดำเนินงานวิจัยและได้นำเสนอดังรูปที่ 3.1 สำหรับบทที่ 4 จะเป็นผลการศึกษาในส่วนต่างๆ ซึ่งประกอบไปด้วย การเก็บรวบรวมข้อมูลที่ใช้ในการวิจัย การวิเคราะห์ความรู้สึกจากหัวข้อข่าวหุ้น การวิเคราะห์ความรู้สึกจากหัวข้อข่าวหุ้น และ ผลการเปรียบเทียบประสิทธิภาพของแบบจำลองส่วนการวิเคราะห์ความรู้สึกจากจากหัวข้อข่าวหุ้น

4.1 ลักษณะของข้อมูลที่ใช้ในการทดลอง

ข้อมูลที่ใช้ในการศึกษาเป็นข้อความหัวข้อข่าว ที่ถูกดึงมากจากเว็บไซต์ข่าวหุ้นระหว่างปี พ.ศ. 2561 - 2565 จำนวนข้อมูลของบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ที่นำมาทดลองจะมีทั้งหมด 3 ชุดข้อมูล โดยแยกเป็น

1) ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น

2) ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) และ ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” , “น้ำมันเตา” , “ฟอสซิล” และ “โรงไฟฟ้า”ร่วมด้วย

3) ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” , “น้ำมันเตา” , “ฟอสซิล” ,“โรงไฟฟ้า” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย

ส่วนบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) นำมาทดลองจะมีทั้งหมด 3 ชุดข้อมูล

1) ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น

2) ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย” และ “ควอลิตี้เฮ้าส์” ร่วมด้วย

3) ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย โดยมีจำนวนข่าวทั้งสิ้นดังตารางที่ 4.1

ตารางที่ 4.1 จำนวนข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH)

ลำดับ	ข้อมูล	ข้อมูลสอน	ข้อมูลทดสอบ	รวม
1	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น	222 หัวข้อข่าว	95 หัวข้อข่าว	317 หัวข้อข่าว
2	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) และปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล” และ “โรงไฟฟ้า” ร่วมด้วย	1,050 หัวข้อข่าว	451 หัวข้อข่าว	1,501 หัวข้อข่าว
3	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” “น้ำมันเตา” , “ฟอสซิล” ,“โรงไฟฟ้า” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ”และ “เศรษฐกิจนอกประเทศ” เข้ามาร่วมด้วย	4,833 หัวข้อข่าว	2,071 หัวข้อข่าว	6,904 หัวข้อข่าว

ตารางที่ 4.2 จำนวนข้อมูลข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH)

ลำดับ	ข้อมูล	ข้อมูลสอน	ข้อมูลทดสอบ	รวม
1	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น	46 หัวข้อข่าว	20 หัวข้อข่าว	65 หัวข้อข่าว
2	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย” และ “ควอลิตี้เฮ้าส์” ร่วมด้วย	586 หัวข้อข่าว	261 หัวข้อข่าว	837 หัวข้อข่าว
3	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ“เศรษฐกิจนอกประเทศ” เข้ามาร่วมด้วย	4,368 หัวข้อข่าว	1,872 หัวข้อข่าว	6,240 หัวข้อข่าว

จากนั้นนำข้อมูลมาเข้าสู่ส่วนการเตรียมข้อมูล โดยมีการพิจารณาตามข้อความข่าวถ้าเป็นข้อความข่าวภาษาอังกฤษ ก็จะไม่ทำการกำจัดเครื่องหมายวรรคตอน แต่ถ้าเป็น ข้อความข่าวภาษาไทย ก็จะมีการกำจัดเครื่องหมายวรรคตอนด้วย แล้วจึงนำข้อความข่าวไปเชื่อมโยงกับราคาหุ้นในแต่วัน โดยใช้วันที่ในการเชื่อมโยงกันและตรวจสอบว่าราคาปิดของวันที่มีข่าวสูงหรือต่ำกว่าวันก่อนหน้าเพื่อนำมาใช้เป็นความรู้สึก Positive Negative และ Neutral ของข่าวนั้น ๆ จะได้จำนวนของความรู้สึกข้อมูลหัวข้อข่าว ตามตาราง 4.3 และ 4.4

ตารางที่ 4.3 จำนวนข้อมูลของ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH)

ลำดับ	ข้อมูล	Pos	Neg	Neu	รวม
1	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น	139	111	67	317
2	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) และปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล”และ“โรงไฟฟ้า” ร่วมด้วย	577	563	361	1,501
3	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล”, “โรงไฟฟ้า” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ”และ“เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	2,519	2,594	1,791	6,904

จากนั้นได้มีการวิเคราะห์คำในแต่ละกลุ่มโดยดูจากความถี่ของคำด้วย Word Cloud เพื่อเป็นการแสดงให้เห็นความถี่ของคำที่ปรากฏในกลุ่มข้อมูล โดยคำที่มีความถี่มากจะปรากฏในภาพด้วยขนาดที่ใหญ่กว่า และคำที่มีความถี่น้อยจะปรากฏด้วยขนาดที่เล็กกว่า ในชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น

รูปที่ 4.1 Word Cloud ของข้อความที่เป็น Positive (Pos) ของหุ้น RATCH

จากรูปที่ 4.1 ที่ได้จากข้อความที่เป็น Positive (Pos) แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเชิงบวก โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความเชิงบวก โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “โรงไฟฟ้า” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความเชิงบวก คำนี้อาจสื่อถึงข่าวดีเกี่ยวกับการเปิดตัวโรงไฟฟ้าใหม่หรือการผลิตพลังงานที่มีประสิทธิภาพ “ล้าน” เป็นคำที่ใช้บ่อยในการระบุจำนวนเงินที่เกี่ยวข้องกับธุรกรรมหรือกำไรที่สูงในข่าวหุ้น “MW” หน่วยวัดพลังงานไฟฟ้าเมกะวัตต์ แสดงถึงการผลิตพลังงานในปริมาณมาก มักใช้ในบริบทของข่าวที่เกี่ยวข้องกับการเพิ่มกำลังการผลิตหรือโครงการพลังงานใหม่ “บาท” หน่วยเงินตราของไทย ใช้บ่งบอกมูลค่าหรือผลกำไรที่ได้รับจากการลงทุน “ซื้อ” แสดงถึงการทำธุรกรรมการซื้อขายหุ้น ซึ่งในบริบทของข้อความเชิงบวก อาจสื่อถึงการซื้อที่เป็นประโยชน์หรือการลงทุนที่ดี

คำที่มีความสำคัญรองลงมา “ปี” บ่งบอกถึงระยะเวลาหรือการเปลี่ยนแปลงในช่วงเวลาหลายปี เช่น การเติบโตของธุรกิจหรือการบรรลุเป้าหมายระยะยาว “กำไร” คำนี้เน้นถึงผลกำไรที่ได้รับจากการดำเนินธุรกิจหรือการลงทุน ซึ่งเป็นปัจจัยสำคัญในการสร้างความรู้สึกเชิงบวก ”ผลิต” แสดงถึงการผลิตที่เพิ่มขึ้นหรือประสิทธิภาพในการผลิต ซึ่งเป็นข่าวดีสำหรับนักลงทุนและผู้ถือหุ้น คำที่เกี่ยวข้องกับพลังงานและการผลิต เช่น โรงไฟฟ้า, MW, และ ผลิตมักปรากฏร่วมกัน แสดงถึงการรายงานข่าวเกี่ยวกับโครงการพลังงานหรือการขยายกำลังการผลิต คำที่เกี่ยวกับการเงิน เช่น ล้าน, บาท, กำไร, และ ซื้อ มักเชื่อมโยงกับข่าวการทำธุรกรรมที่สำเร็จและการได้รับผลตอบแทนที่ดี

รูปที่ 4.2 Word Cloud ของข้อความที่เป็น Negative (Neg) ของหุ้น RATCH

จากรูปที่ 4.2 ที่ได้จากข้อความที่เป็น Negative แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเชิงลบ โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความเชิงลบ โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “บาท” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความเชิงลบ คำนี้อาจสื่อถึงความผันผวนหรือการลดลงของค่าเงินบาทในการซื้อขายหุ้น “โรงไฟฟ้า” คำที่เกี่ยวข้องกับโรงไฟฟ้ามักมีข่าวสารที่ไม่ดี เช่น ปัญหาการดำเนินงานหรือการปิดตัว “ล้าน” การใช้คำว่า "ล้าน" ในบริบทเชิงลบอาจบ่งบอกถึงการสูญเสียเงินจำนวนมากหรือการรายงานผลขาดทุน “MW” หน่วยวัดพลังงานไฟฟ้าเมกะวัตต์ที่อาจเกี่ยวข้องกับการลดลงของการผลิตหรือการหยุดชะงักในโครงการพลังงาน “ซื้อ” การซื้อที่ไม่ประสบความสำเร็จหรือการลงทุนที่มีความเสี่ยงสูง

คำที่มีความสำคัญรองลงมา “กำไร” แม้คำนี้จะมีความหมายเชิงบวก แต่ในบริบทเชิงลบอาจหมายถึงการคาดการณ์กำไรที่ลดลงหรือการประกาศขาดทุน “ขาย” อาจเกี่ยวข้องกับการขายหุ้นในราคาที่ต่ำกว่าที่คาดหวังหรือการลดสัดส่วนการถือหุ้น “ลงทุน” การลงทุนที่ไม่ประสบความสำเร็จหรือมีความเสี่ยงสูง คำที่เกี่ยวข้องกับการเงินและการลงทุน เช่น บาท, ล้าน, ซื้อ, ขาย, และ กำไร มักปรากฏร่วมกัน แสดงถึงแนวโน้มที่ไม่ดีในการดำเนินธุรกิจหรือการลงทุนคำที่เกี่ยวข้องกับพลังงาน เช่น โรงไฟฟ้า และ MW อาจบ่งบอกถึงปัญหาในการผลิตหรือการดำเนินงานที่ไม่ประสบความสำเร็จ

รูปที่ 4.3 Word Cloud ของข้อความที่เป็น Neutral (Neu) ของหุ้น RATCH

จากรูปที่ 4.3 ที่ได้จากข้อความที่เป็น Neutral แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเป็นกลาง โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความรู้สึกเป็นกลาง โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “บาท” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความที่มีความรู้สึกเป็นกลาง คำนี้มักใช้ในการรายงานมูลค่าหรือราคาในข่าวหุ้น “โรงไฟฟ้า” เป็นคำที่มีความถี่สูงเช่นกัน แสดงถึงข่าวเกี่ยวกับการผลิตพลังงานหรือการดำเนินงานของโรงไฟฟ้า “ล้าน” การใช้คำว่า "ล้าน" ในบริบทเป็นกลางมักจะหมายถึงการระบุจำนวนเงินหรือมูลค่าโดยไม่เน้นถึงความรู้สึกเชิงบวกหรือลบ “MW” หน่วยวัดพลังงานไฟฟ้าเมกะวัตต์ที่อาจใช้ในการรายงานการผลิตพลังงานหรือกำลังการผลิต “ซื้อ” แสดงถึงการทำธุรกรรมการซื้อขายหุ้นที่รายงานโดยไม่เน้นถึงผลกระทบเชิงบวกหรือลบ

คำที่มีความสำคัญรองลงมา “COD” การเปิดดำเนินการเชิงพาณิชย์ (Commercial Operation Date) เป็นคำที่ใช้บ่อยในการรายงานวันที่เริ่มดำเนินการผลิตพลังงาน “พัน” คำนี้ใช้ในการรายงานจำนวนเงินที่มีขนาดเล็กกว่าล้าน “กำไร” ในบริบทเป็นกลางอาจหมายถึงการรายงานกำไรที่ได้รับโดยไม่เน้นถึงผลกระทบเชิงบวกหรือลบ คำที่เกี่ยวข้องกับการเงินและการลงทุน เช่น บาท, ล้าน, ซื้อ, ขาย, และ กำไร มักปรากฏร่วมกัน แสดงถึงการรายงานข่าวสารทางการเงินและการลงทุนที่เป็นกลาง คำที่เกี่ยวข้องกับพลังงาน เช่น โรงไฟฟ้า, MW, และ COD แสดงถึงการรายงานข่าวสารเกี่ยวกับการผลิตพลังงานและการดำเนินงานในอุตสาหกรรมพลังงาน

ตารางที่ 4.4 จำนวนข้อมูลของ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH)

ลำดับ	ข้อมูล	Pos	Neg	Neu	รวม
1	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น	27	24	14	65
2	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ ปัจจัยที่เกี่ยวข้องกับบริษัท โดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย” และ “ควอลิตี้เฮ้าส์” ร่วมด้วย	279	284	274	837
3	ข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	1,840	2,122	2,278	6,240

จากนั้นได้มีการวิเคราะห์คำในแต่ละกลุ่มโดยดูจากความถี่ของคำด้วย Word Cloud เพื่อเป็นการแสดงให้เห็นความถี่ของคำที่ปรากฏในกลุ่มข้อมูล โดยคำที่มีความถี่มากจะปรากฏในภาพด้วยขนาดที่ใหญ่กว่า และคำที่มีความถี่น้อยจะปรากฏด้วยขนาดที่เล็กกว่า ในชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น

รูปที่ 4.4 Word Cloud ของข้อความที่เป็น Positive (Pos) ของหุ้น QH

จากรูปที่ 4.4 ที่ได้จากข้อความที่เป็น Positive (Pos) แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเชิงบวก โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความเชิงบวก โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “กำไร” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความเชิงบวก คำนี้มักจะใช้ในการรายงานผลกำไรของบริษัทซึ่งสร้างความรู้สึกเชิงบวกในข่าว “บาท” คำที่ใช้บ่อยในการระบุจำนวนเงินหรือมูลค่าในการทำธุรกรรมทางการเงิน โดยมีความรู้สึกเชิงบวกเกี่ยวกับการเติบโตหรือกำไร “ปันผล” แสดงถึงการจ่ายเงินปันผลให้กับผู้ถือหุ้น ซึ่งเป็นข่าวดีและสร้างความรู้สึกเชิงบวก “ปี” การระบุช่วงเวลาหรือผลลัพธ์ในปีที่กำหนด เช่น การเติบโตในช่วงปีที่ผ่านมา

คำที่มีความสำคัญรองลงมา “โครงการ” คำที่แสดงถึงการดำเนินโครงการใหม่หรือการขยายโครงการที่ประสบความสำเร็จ “ไตรมาส” การรายงานผลประกอบการในแต่ละไตรมาสที่มีผลเชิงบวก “เด่น” คำที่ใช้ในการบรรยายผลลัพธ์หรือคุณสมบัติที่โดดเด่นของบริษัทหรือหุ้น คำที่เกี่ยวข้องกับการเงินและการลงทุน เช่น กำไร, บาท, และ ปันผล มักปรากฏร่วมกัน แสดงถึงแนวโน้มที่ดีในการดำเนินธุรกิจและการลงทุนคำที่เกี่ยวข้องกับการดำเนินโครงการ เช่น โครงการ และ ไตรมาส แสดงถึงความสำเร็จในการดำเนินโครงการและการรายงานผลประกอบการที่ดี

รูปที่ 4.5 Word Cloud ของข้อความที่เป็น Negative (Neg) ของหุ้น QH

จากรูปที่ 4.5 ที่ได้จากข้อความที่เป็น Negative แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเชิงลบ โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความเชิงลบ โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “กำไร” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความเชิงลบ ในบริบทเชิงลบ คำนี้อาจหมายถึงการคาดการณ์กำไรที่ลดลงหรือการประกาศขาดทุน “ปันผล” การปันผลที่ไม่เป็นไปตามคาดหวังหรือการลดจำนวนเงินปันผลอาจสร้างความรู้สึกเชิงลบในข่าว “ล้าน” การใช้คำว่า "ล้าน" ในบริบทเชิงลบอาจบ่งบอกถึง

การสูญเสียเงินจำนวนมากหรือการรายงานผลขาดทุน “บาท” คำนี้มักใช้ในการระบุจำนวนเงินที่เกี่ยวข้องกับการขาดทุนหรือการสูญเสียทางการเงิน “รายได้” การรายงานรายได้ที่ลดลงหรือไม่เป็นไปตามเป้าหมายอาจเป็นปัจจัยที่สร้างความรู้สึกเชิงลบ

คำที่มีความสำคัญรองลงมา “ปี” การรายงานผลประกอบการในแต่ละปีที่มีผลลัพธ์เชิงลบ “โครงการ” ปัญหาในโครงการที่ไม่สำเร็จหรือเกิดความล่าช้า “ไตรมาส” การรายงานผลประกอบการในแต่ละไตรมาสที่ไม่เป็นไปตามคาดหวัง คำที่เกี่ยวข้องกับการเงินและการลงทุน เช่น กำไร, ปันผล, ล้าน, บาท, และ รายได้ มักปรากฏร่วมกัน แสดงถึงแนวโน้มที่ไม่ดีในการดำเนินธุรกิจหรือการลงทุน คำที่เกี่ยวข้องกับการดำเนินโครงการ เช่น โครงการ และ ไตรมาส อาจบ่งบอกถึงปัญหาในการดำเนินโครงการหรือการรายงานผลประกอบการที่ไม่เป็นไปตามคาดหวัง

รูปที่ 4.6 Word Cloud ของข้อความที่เป็น Neutral (Neu) ของหุ้น QH

จากรูปที่ 4.6 ที่ได้จากข้อความที่เป็น Neutral แสดงถึงคำที่ปรากฏบ่อยในข่าวหุ้นที่มีความรู้สึกเป็นกลาง โดยแต่ละคำมีขนาดแตกต่างกันขึ้นอยู่กับความถี่ของการปรากฏในข้อความรู้สึกเป็นกลาง โดยสามารถวิเคราะห์คำที่ปรากฏบ่อยที่สุด คือ “กำไร” คำนี้มีขนาดใหญ่ที่สุดใน Word Cloud แสดงถึงความสำคัญและความถี่ที่พบในข้อความที่มีความรู้สึกเป็นกลาง การรายงานกำไรในบริบทที่ไม่เน้นถึงความรู้สึกเชิงบวกหรือลบ “ปันผล” การรายงานการจ่ายปันผลให้กับผู้ถือหุ้น โดยไม่มีการเน้นถึงผลกระทบเชิงบวกหรือลบ “ล้าน” การใช้คำว่า "ล้าน" ในการระบุจำนวนเงินหรือมูลค่าที่เกี่ยวข้องกับการทำธุรกรรมทางการเงิน “บาท” คำนี้มักใช้ในการระบุจำนวนเงินในบริบทของการรายงานทางการเงิน “รายได้” การรายงานรายได้ที่ได้รับโดยไม่เน้นถึงผลกระทบเชิงบวกหรือลบ

คำที่มีความสำคัญรองลงมา “ไตรมาส” การรายงานผลประกอบการในแต่ละไตรมาสที่มีข้อมูลเป็นกลาง “ปี” การระบุช่วงเวลาหรือผลลัพธ์ในแต่ละปีที่เป็นกลาง “ซื้อ” การทำธุรกรรมการซื้อขายหุ้นที่รายงานโดยไม่เน้นถึงผลกระทบเชิงบวกหรือลบ คำที่เกี่ยวข้องกับการเงินและการลงทุน

เช่น กำไร, ปันผล, ล้าน, บาท, และ รายได้ มักปรากฏร่วมกัน แสดงถึงการรายงานข่าวสารทางการเงินและการลงทุนที่เป็นกลาง คำที่เกี่ยวข้องกับการดำเนินโครงการ เช่น ไตรมาส และ ปี แสดงถึงการรายงานผลประกอบการและการดำเนินงานที่เป็นกลาง

4.2 การเปรียบเทียบผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันตามตัวตัดคำ

การจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจะเริ่มจากการเปรียบเทียบตัวตัดคำภาษาไทยตามชุดข้อมูลของหุ้นแต่ละตัวที่ได้ทำการเก็บมาในหัวข้อ 4.1 ที่ได้มาการนำมาใช้ในงานวิจัยโดยผลส่วนนี้จะเป็นคำตอบที่สอดคล้องกับวัตถุประสงค์ที่ 1 ของงานวิจัยนี้ โดยจะสรุปผลตามหุ้น

4.2.1.ผลจากตัวตัดคำของหัวข้อข่าวหุ้นรายวันบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH)

ผลจากตัวตัดคำจะออกมาตามแบบจำลองที่ได้นำมาใช้ในงานวิจัย 10 แบบจำลอง แสดงค่าการวัดประสิทธิภาพตามค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) และค่าความถ่วงดุล (F-Measure) โดยแยกเป็น ตัวตัดคำ PythaiNLP ได้ผลตามตาราง 4.5 ตัวตัดคำ DeepCut ได้ผลตามตาราง 4.6

ตารางที่ 4.5 ผลจากตัวตัดคำ PythaiNLP บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น	Logistic Regression	52.08	54.03	52.08	49.69
2		Naive Bayes	56.25	54.98	56.25	51.79
3		KNN	38.54	40.85	38.54	39.25
4		SVM	62.50	67.29	62.50	60.31
5		Random Forest	51.04	57.56	51.04	51.38
6		Decision Tree	52.08	60.91	52.08	53.37
7		Neural network	50.00	51.85	50.00	50.21
8		Perceptron	50.00	50.35	50.00	50.02
9		Stochastic Gradient Descent	52.08	54.95	52.08	52.34
10		Passive Aggressive	48.96	51.52	48.96	49.21

ตารางที่ 4.5 ผลจากตัวตัดคำ PythaiNLP บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ (ต่อ)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
11	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) และปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” , “น้ำมันเตา”, “ฟอสซิล” และ “โรงไฟฟ้า”ร่วมด้วย	Logistic Regression	39.69	37.57	39.69	37.76
12		Naive Bayes	42.79	46.70	42.79	37.78
13		KNN	36.59	37.07	36.59	35.93
14		SVM	41.46	47.48	41.46	38.29
15		Random Forest	41.24	41.97	41.24	39.34
16		Decision Tree	41.02	40.77	41.02	40.76
17		Neural network	38.80	38.70	38.80	38.74
18		Perceptron	40.58	39.93	40.58	40.09
19		Stochastic Gradient Descent	40.13	39.38	40.13	39.47
20		Passive Aggressive	39.47	39.12	39.47	39.05
21	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” , “น้ำมันเตา” , “ฟอสซิล” ,“โรงไฟฟ้า” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	Logistic Regression	38.71	36.73	38.71	36.75
22		Naive Bayes	39.19	37.02	39.19	35.06
23		KNN	37.11	36.27	37.11	35.80
24		SVM	40.69	39.63	40.69	37.19
25		Random Forest	39.67	38.04	39.67	37.67
26		Decision Tree	37.74	37.37	37.74	37.51
27		Neural network	38.51	38.03	38.51	38.15
28		Perceptron	36.63	36.91	36.63	36.76
29		Stochastic Gradient Descent	38.47	37.38	38.47	37.38
30		Passive Aggressive	37.26	37.90	37.26	37.50
ค่าเฉลี่ย			**43.31**	**44.28**	**43.31**	**42.15**

ตารางที่ 4.6 ผลจากตัวตัดคำ DeepCut บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น	Logistic Regression	56.25	54.77	56.25	53.63
2		Naive Bayes	58.33	46.92	58.33	50.81
3		KNN	38.54	41.24	38.54	38.77
4		SVM	62.50	63.99	62.50	58.73
5		Random Forest	53.13	59.05	53.13	53.36
6		Decision Tree	47.92	54.21	47.92	49.01
7		Neural network	53.13	53.84	53.13	52.91
8		Perceptron	52.08	57.48	52.08	52.57
9		Stochastic Gradient Descent	59.38	58.37	59.38	58.14
10		Passive Aggressive	52.08	55.10	52.08	52.44
11	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) และปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน”, “น้ำมันเตา”, “ฟอสซิล” และ “โรงไฟฟ้า”ร่วมด้วย	Logistic Regression	41.02	40.57	41.02	38.85
12		Naive Bayes	39.02	48.97	39.02	34.97
13		KNN	34.15	34.07	34.15	33.40
14		SVM	42.35	52.74	42.35	38.76
15		Random Forest	42.57	41.08	42.57	39.97
16		Decision Tree	39.69	39.33	39.69	39.18
17		Neural network	40.35	39.88	40.35	39.97
18		Perceptron	40.80	39.92	40.80	40.11
19		Stochastic Gradient Descent	38.80	38.34	38.80	38.43
20		Passive Aggressive	41.02	40.44	41.02	40.53

ตารางที่ 4.6 ผลจากตัวตัดคำ DeepCut บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ชุดข้อมูลทดสอบ (ต่อ)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
21	ข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “ก๊าซธรรมชาติ”, “ถ่านหิน” , “น้ำมันเตา” , “ฟอสซิล” ,“โรงไฟฟ้า” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	Logistic Regression	39.48	37.48	39.48	36.62
22		Naive Bayes	40.25	41.89	40.25	34.20
23		KNN	37.16	36.83	37.16	36.43
24		SVM	40.35	39.11	40.35	36.26
25		Random Forest	39.62	37.85	39.62	37.52
26		Decision Tree	35.76	35.35	35.76	35.51
27		Neural network	38.27	37.94	38.27	38.07
28		Perceptron	39.43	37.94	39.43	37.72
29		Stochastic Gradient Descent	39.86	38.18	39.86	37.58
30		Passive Aggressive	38.13	37.87	38.13	37.53
ค่าเฉลี่ย			**44.05**	**44.69**	**44.05**	**42.40**

สรุปผลการวัดประสิทธิภาพจากตัวตัดคำได้ว่า ตัวตัดคำ PythaiNLP กับข้อมูลบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) มีค่าเฉลี่ยจากแบบจำลองทั้ง 10 แบบ 3 ชุดข้อมูลต่อหุ้น (A, B, C) ส่วนชุดข้อมูลทดสอบได้ค่าความถูกต้องเฉลี่ยร้อยละ 43.31 ค่าความแม่นยำเฉลี่ยร้อยละ 44.28 ค่าความระลึกเฉลี่ยร้อยละ 43.31 และค่าความถ่วงดุลเฉลี่ยร้อยละ 42.15 ซึ่งต่ำกว่าค่าเฉลี่ยของตัวตัดคำ DeepCut ที่ได้ค่าความถูกต้องเฉลี่ยร้อยละ 44.05 ค่าความแม่นยำเฉลี่ยร้อยละ 44.69 ค่าความระลึกเฉลี่ยร้อยละ 44.05 และค่าความถ่วงดุลเฉลี่ยร้อยละ 42.40

ผลการวัดประสิทธิภาพรายชุดข้อมูลของแต่ละแบบจำลองทั้ง 10 แบบ ตัวตัดคำ PythaiNLP ชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นส่วนชุดข้อมูลทดสอบของแบบจำลอง ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) มีค่าประสิทธิภาพสูงที่สุด อยู่ที่ค่าความถูกต้องร้อยละ 62.50 ค่าความแม่นยำร้อยละ 67.29 ค่าความระลึกร้อยละ 62.50 และค่าความถ่วงดุลร้อยละ 60.31จึงทำให้สรุปได้ว่า ตัวตัดคำ PythaiNLP ใช้ได้ดีกับชุดข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น จึงได้นำตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นมาใช้ในการพัฒนาแบบจำลองซึ่งจะแสดงผลในหัวข้อต่อไป

4.2.2.ผลจากตัวตัดคำของหัวข้อข่าวหุ้นรายวันบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH)

ผลจากตัวตัดคำจะออกมาตามแบบจำลองที่ได้นำมาใช้ในงานวิจัย 10 แบบจำลอง แสดงค่าการวัดประสิทธิภาพตามค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) และค่าความถ่วงดุล (F-Measure) โดยแยกเป็นตัวตัดคำ PythaiNLP ได้ผลตามตาราง 4.7 ตัวตัดคำ DeepCut ได้ผลตามตาราง 4.8

ตารางที่ 4.7 ผลจากตัวตัดคำ PythaiNLP บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ชุดข้อมูลทดสอบ

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น	Logistic Regression	40.00	24.67	40.00	29.80
2		Naive Bayes	45.00	31.82	45.00	37.26
3		KNN	40.00	35.24	40.00	34.86
4		SVM	40.00	25.00	40.00	29.33
5		Random Forest	60.00	72.50	60.00	58.45
6		Decision Tree	60.00	70.00	60.00	59.00
7		Neural network	55.00	63.47	55.00	56.39
8		Perceptron	60.00	65.00	60.00	60.00
9		Stochastic Gradient Descent	60.00	66.86	60.00	60.51
10		Passive Aggressive	65.00	64.67	65.00	64.60
11	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย” และ “ควอลิตี้เฮ้าส์” ร่วมด้วย	Logistic Regression	37.70	37.58	37.70	37.54
12		Naive Bayes	39.29	39.70	39.29	38.51
13		KNN	35.32	35.08	35.32	34.60
14		SVM	36.11	36.00	36.11	35.99
15		Random Forest	39.29	39.08	39.29	38.95
16		Decision Tree	35.71	36.13	35.71	35.67
17		Neural network	40.48	40.47	40.48	40.46
18		Perceptron	36.11	36.22	36.11	36.11
19		Stochastic Gradient Descent	36.90	36.91	36.90	36.85
20		Passive Aggressive	39.29	39.31	39.29	39.22

ตารางที่ 4.7 ผลจากตัวตัดคำPythaiNLPบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH)ชุดข้อมูลทดสอบ (ต่อ)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
21	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	Logistic Regression	38.57	38.57	38.57	37.98
22		Naive Bayes	37.66	37.92	37.66	34.85
23		KNN	37.02	37.17	37.02	35.85
24		SVM	38.09	38.05	38.09	35.95
25		Random Forest	38.62	38.26	38.62	37.59
26		Decision Tree	36.11	35.80	36.11	35.81
27		Neural network	37.13	37.22	37.13	37.11
28		Perceptron	36.32	36.16	36.32	35.93
29		Stochastic Gradient Descent	38.14	38.06	38.14	37.97
30		Passive Aggressive	37.82	37.80	37.82	37.81
ค่าฉลี่ย			**42.56**	**42.36**	**42.56**	**41.03**

ตารางที่ 4.8 ผลจากตัวตัดคำ DeepCut บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ชุดข้อมูลทดสอบ

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น	Logistic Regression	40.00	17.78	40.00	24.62
2		Naive Bayes	45.00	30.00	45.00	35.45
3		KNN	30.00	21.38	30.00	24.50
4		SVM	40.00	17.78	40.00	24.62
5		Random Forest	55.00	70.00	55.00	48.57
6		Decision Tree	60.00	71.54	60.00	55.24
7		Neural network	40.00	46.00	40.00	35.87
8		Perceptron	45.00	37.50	45.00	36.67
9		Stochastic Gradient Descent	35.00	36.67	35.00	33.33
10		Passive Aggressive	40.00	36.00	40.00	34.20

ตารางที่ 4.8 ผลจากตัวตัดคำ DeepCut บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ชุดข้อมูลทดสอบ (ต่อ)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
11	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และ ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย” และ “ควอลิตี้เฮ้าส์” ร่วมด้วย	Logistic Regression	38.49	38.99	38.49	38.38
12		Naive Bayes	37.70	38.73	37.70	36.83
13		KNN	38.10	39.17	38.10	37.81
14		SVM	40.08	40.01	40.08	39.10
15		Random Forest	31.35	31.75	31.35	31.47
16		Decision Tree	32.14	32.43	32.14	32.20
17		Neural network	36.11	36.18	36.11	36.13
18		Perceptron	34.52	34.65	34.52	34.44
19		Stochastic Gradient Descent	35.71	35.82	35.71	35.62
20		Passive Aggressive	38.49	38.61	38.49	38.41
21	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) ปัจจัยที่เกี่ยวข้องกับบริษัทโดยมีข้อมูลที่เกี่ยวข้องกับ “อสังหา”, “คอนโด”, “บ้าน”, “ปล่อยกู้”, “ดอกเบี้ย”, “หนี้เสีย”, “ควอลิตี้เฮ้าส์” และยังมีการนำหัวข้อข่าว “เศรษฐกิจภายในประเทศ” และ “เศรษฐกิจนอกประเทศ”เข้ามาร่วมด้วย	Logistic Regression	36.16	35.61	36.16	34.97
22		Naive Bayes	37.55	36.43	37.55	32.48
23		KNN	33.92	33.64	33.92	32.44
24		SVM	36.75	36.31	36.75	34.35
25		Random Forest	38.51	38.10	38.51	37.42
26		Decision Tree	36.49	36.41	36.49	36.30
27		Neural network	34.94	34.91	34.94	34.89
28		Perceptron	37.61	36.55	37.61	34.01
29		Stochastic Gradient Descent	37.02	37.07	37.02	36.27
30		Passive Aggressive	35.47	35.78	35.47	35.30
ค่าเฉลี่ย			**38.57**	**37.06**	**38.57**	**35.40**

สรุปผลการวัดประสิทธิภาพจากตัวตัดคำได้ว่า ตัวตัดคำ PythaiNLP กับข้อมูลบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) มีค่าเฉลี่ยจากแบบจำลองทั้ง 10 แบบ 3 ชุดข้อมูลต่อหุ้น (A, B, C) ส่วนชุดข้อมูลทดสอบได้ค่าความถูกต้องเฉลี่ย 42.56 ค่าความแม่นยำเฉลี่ยร้อยละ 42.36 ค่าความระลึกเฉลี่ยร้อยละ 42.56 และค่าความถ่วงดุลเฉลี่ยร้อยละ 41.03 ซึ่งสูงกว่า ค่าเฉลี่ยของตัวตัดคำ DeepCut ที่ได้ค่าความถูกต้องเฉลี่ยร้อยละ 38.57 ค่าความแม่นยำเฉลี่ยร้อยละ 37.06 ค่าความระลึกเฉลี่ยร้อยละ 38.57 และค่าความถ่วงดุลเฉลี่ยร้อยละ 35.40

ผลการวัดประสิทธิภาพรายชุดข้อมูลของแต่ละแบบจำลองทั้ง 10 แบบแล้ว ตัวตัดคำ PythaiNLP ชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นส่วนชุดข้อมูลทดสอบ ของแบบจำลอง พาสซีฟ อากัสซีฟ (Passive Aggressive) มีค่าประสิทธิภาพสูงที่สุด อยู่ที่ค่าความถูกต้องร้อยละ 65 ค่าความแม่นยำร้อยละ 64.67 ค่าความระลึกร้อยละ 65 และค่าความถ่วงดุลร้อยละ 64.60 จึงทำให้สรุปได้ว่า ตัวตัดคำ PythaiNLP ใช้ได้ดีกับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น จึงได้นำตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นมาใช้ในการพัฒนาแบบจำลองซึ่งจะแสดงผลในหัวข้อต่อไป

4.3 ผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจากแบบจำลอง

จากผลของการการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจากตัวตัดคำหัวข้อที่ 4.2 ที่ว่าตัวตัดทำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นมีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงกว่าตัวตัดคำกับชุดข้อมูลอื่น และตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นมีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงกว่าตัวตัดคำกับชุดข้อมูลอื่นในหัวข้อนี้จึงนำตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นและตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น นำมาปรับจูนแบบจำลองทั้ง 10 แบบจำลองของทั้ง 2 หุ้นให้มีประสิทธิภาพสูงที่สุด โดยค่าพารามิเตอร์ในการใช้เพิ่มประสิทธิภาพแบบจำลองจะอยู่ในตารางที่ 3.9 และ 3.11

ผลของส่วนนี้จะเป็นคำตอบที่สอดคล้องกับวัตถุประสงค์ที่ 2 ของงานวิจัยนี้ โดยจะมีแบบจำลองที่ใช้ในการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันได้แก่ แบบจำลองการถดถอยเชิงโลจิสติค (Logistic Regression) นาอีฟเบย์ (Naïve Bayes) การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ป่าสุ่ม (Random Forest) ต้นไม้ตัดสินใจ (Decision Tree) ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) เพอร์เซปตรอน (Perceptron) สโตแคสติกการ์เดียนดิเซนท์ (SGD) พาสซีฟ อากัสซีฟ (Passive Aggressive) ซึ่งหลังจากข้อมูลผ่านกระบวนการเตรียมข้อมูลในหัวข้อที่ 3.2 แล้วข้อมูลจะมีการแบ่งชุดข้อมูลเป็น ชุดข้อมูลเรียนรู้ 70% ชุดข้อมูลทดสอบ 30% แบ่งโดยการสุ่มหลังจากนั้นจะถูกนำไปสร้างแบบจำลอง

ข้อมูลสำหรับการฝึกสอนนั้นจะถูกนำไปใช้สอนแบบจำลองต่างๆ เพื่อให้แบบจำลองเกิดการเรียนรู้ เพื่อให้สามารจำแนกความรู้สึก Positive Negative และ Neutral ซึ่งแบบจำลองที่ผ่านการเรียนรู้นั้นจะถูกนำมาทดสอบด้วยข้อมูลทดสอบ 30% โดยการวัดผลสำหรับแบบจำลองจะให้ชุดข้อมูลทดสอบซึ่งใช้ตัววัด 3 แบบได้แก่ ค่าความถูกต้อง ค่าความระลึก และค่าความแม่นยำ

4.3.1.ผลการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันจากแบบจำลองชุดข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นและใช้ตัวตัดคำ PythaiNLP

4.3.1.1 ผลจากแบบจำลองการถดถอยเชิงโลจิสติค (Logistic Regression)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการถดถอยเชิงโลจิสติคคือ C = 100 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของการถดถอยเชิงโลจิสติค ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.9 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 16 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 11 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 29 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 22 ครั้ง

ตารางที่ 4.9 ผลการทดสอบแบบจำลองการถดถอยเชิงโลจิสติคด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	16	5	6	27
Negative	5	5	8	18
Neutral	18	4	29	51
รวม	39	14	43	96

จากตาราง 4.9 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง (Accuracy)

$$\text{ค่าความถูกต้อง (Accuracy)} = \frac{16+5+29}{96} = 0.5208 \times 100 = 52.08$$

2. ค่าความแม่นยำ (Precision)

$$\text{ค่าความแม่นยำ (Precision) สำหรับ Pos} = \frac{16}{16+23} = 0.4103$$

$$\text{ค่าความแม่นยำ (Precision) สำหรับ Neg} = \frac{5}{5+9} = 0.3571$$

ค่าความแม่นยำ (Precision) สำหรับ Nue = $\frac{29}{29+14}$ = 0.6744

จำนวนตัวอย่างในแต่ละคลาส

คลาส Positive = 27

คลาส Negative = 18

คลาส Neutral = 51

Precision เฉลี่ย = $\frac{(0.4103\times27)+(0.3571\times18)+(0.6744\times51)}{96}$ x 100

= 54.06

3. ค่าความระลึก (Recall)

ค่าความระลึก (Recall) สำหรับ Pos = $\frac{16}{16+11}$ = 0.5925

ค่าความระลึก (Recall) สำหรับ Neg = $\frac{5}{5+13}$ = 0.2778

ค่าความระลึก (Recall) สำหรับ Nue = $\frac{29}{29+22}$ = 0.5686

จำนวนตัวอย่างในแต่ละคลาส

คลาส Positive = 27

คลาส Negative = 18

คลาส Neutral = 51

Precision เฉลี่ย = $\frac{(0.5925\times27)+(0.2778\times18)+(0.5686\times51)}{96}$ x 100

= 52.08

จากตาราง 4.9 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 52.08%
2. ค่าความแม่นยำ 54.06%
3. ค่าความระลึก 52.08%

4.3.1.2 ผลจากแบบจำลองนาอีฟเบย์ (Naïve Bayes)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองนาอีฟเบย์ คือ alpha = 1 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของนาอีฟเบย์ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.10 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 15 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 12 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 1 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 18 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 38 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง

ตารางที่ 4.10 ผลการทดสอบแบบจำลองนาอีฟเบย์ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	15	1	11	**31**
Negative	2	1	15	**18**
Neutral	13	0	38	**51**
รวม	**30**	**2**	**64**	**96**

จากตาราง 4.10 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 56.25%
2. ค่าความแม่นยำ 54.98%
3. ค่าความระลึก 56.25%

4.3.1.3 ผลจากแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด คือ n_neighbors = 7 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของการหาค่าด้วยเพื่อนบ้านที่ใกล้

ที่สุด k จุด ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.11 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 14 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 2 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 16 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 25 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 26 ครั้ง

ตารางที่ 4.11 ผลการทดสอบแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	14	6	7	**27**
Negative	4	2	12	**18**
Neutral	14	12	25	**51**
รวม	**32**	**20**	**44**	**96**

จากตาราง 4.11 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 42.70%
2. ค่าความแม่นยำ 44.36%
3. ค่าความระลึก 42.70%

4.3.1.4 ผลจากแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน (SVM)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน คือ C = 1, gamma = scale และ kernel = rbf โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.12 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 13 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 14 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 42 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 9 ครั้ง

ตารางที่ 4.12 ผลการทดสอบแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	13	0	14	**27**
Negative	2	5	11	**18**
Neutral	9	0	42	**51**
รวม	**24**	**5**	**67**	**96**

จากตาราง 4.12 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 62.5%
2. ค่าความแม่นยำ 67.29%
3. ค่าความระลึก 62.5%

4.3.1.5 ผลจากแบบจำลองป่าสุ่ม (Random Forest)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองป่าสุ่ม คือ max_depth = 10 และ n_estimators = 200 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของป่าสุ่ม ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.13 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 20 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 7 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 26 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 25 ครั้ง

ตารางที่ 4.13 ผลการทดสอบแบบจำลองป่าสุ่ม ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	20	1	6	**27**
Negative	8	5	5	**18**
Neutral	24	1	26	**51**
รวม	**52**	**7**	**37**	**96**

จากตาราง 4.13 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 53.13%
2. ค่าความแม่นยำ 61.54%
3. ค่าความระลึก 53.13%

4.3.1.6 ผลจากแบบจำลองต้นไม้ตัดสินใจ (Decision Tree)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองต้นไม้ตัดสินใจ คือ max_depth = None โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.14 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 20 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 7 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 7 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 11 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 24 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 27 ครั้ง

ตารางที่ 4.14 ผลการทดสอบแบบจำลองต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	20	4	3	**27**
Negative	8	7	3	**18**
Neutral	22	5	24	**51**
รวม	**50**	**16**	**30**	**96**

จากตาราง 4.14 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 53.16%
2. ค่าความแม่นยำ 61.95%
3. ค่าความระลึก 53.13%

4.3.1.7 ผลจากแบบจำลองตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น(MLP)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นคือ hidden_layer_sizes = (50, 100) โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของแบบโครงข่ายประสาท

เทียมหลายชั้นด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.15 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 15 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 12 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 30 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 21 ครั้ง

ตารางที่ 4.15 ผลการทดสอบแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	15	4	8	**27**
Negative	5	5	8	**18**
Neutral	14	7	30	**51**
รวม	**34**	**16**	**46**	**96**

จากตาราง 4.15 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 52.08%
2. ค่าความแม่นยำ 52.91%
3. ค่าความระลึก 52.08%

4.3.1.8 ผลจากแบบจำลองเพอร์เซปตรอน (Perceptron)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองเพอร์เซปตรอน คือ alpha = 0.0001 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของแบบเพอร์เซปตรอน ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.16 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 15 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 12 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 28 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 23 ครั้ง

ตารางที่ 4.16 ผลการทดสอบแบบจำลองเพอร์เซปตรอน ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	15	2	10	**27**
Negative	4	5	9	**18**
Neutral	13	10	28	**51**
รวม	**32**	**17**	**47**	**96**

จากตาราง 4.16 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 50%
2. ค่าความแม่นยำ 50.38%
3. ค่าความระลึก 50%

4.3.1.9 ผลจากแบบจำลองสโตแคสติกการ์เดียนดิเซนท์ (SGD)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการลดความชันแบบสุ่ม คือ alpha = 0.0001 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบ สามารถแสดงได้ดังตาราง 4.17 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 17 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 10 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 6 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 12 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 27 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 24 ครั้ง

ตารางที่ 4.17 ผลการทดสอบแบบจำลองการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	17	7	3	**27**
Negative	5	6	7	**18**
Neutral	18	6	27	**51**
รวม	**40**	**19**	**37**	**96**

จากตาราง 4.17 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 52.08%
2. ค่าความแม่นยำ 56.64%
3. ค่าความระลึก 52.08%

4.3.1.10 ผลจากแบบจำลองพาสซีฟ อากัสซีฟ (Passive Aggressive)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองพาสซีฟ อากัสซีฟ คือ C = 10 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.18 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 96 หัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 17 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 10 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 5 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 13 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 26 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 25 ครั้ง

ตารางที่ 4.18 ผลการทดสอบแบบจำลองพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	17	4	6	**27**
Negative	5	5	8	**18**
Neutral	17	8	26	**51**
รวม	**39**	**17**	**40**	**96**

จากตาราง 4.18 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 50%
2. ค่าความแม่นยำ 52.31%
3. ค่าความระลึก 50%

4.3.2 ผลจากแบบจำลองชุดข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น ด้วยตัวตัดคำ PythaiNLP

4.3.2.1 ผลจากแบบจำลองการถดถอยเชิงโลจิสติค (Logistic Regression)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการถดถอยเชิงโลจิสติค คือ C = 100 โดยหลังจากปรับพารามิเตอร์แล้ว

ได้ผลผลการทดสอบประสิทธิภาพของการถดถอยเชิงโลจิสติค ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.19 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 7 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 1 ครั้ง

ตารางที่ 4.19 ผลการทดสอบแบบจำลองการถดถอยเชิงโลจิสติคด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	6
Negative	2	3	1	6
Neutral	1	0	7	8
รวม	6	3	11	20

จากตาราง 4.19 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 65%
2. ค่าความแม่นยำ 70.45%
3. ค่าความระลึก 65%

4.3.2.2 ผลจากแบบจำลองนาอีฟเบย์ (Naïve Bayes)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองนาอีฟเบย์คือ alpha = 1.0 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของนาอีฟเบย์ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.20 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 0 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง

ตารางที่ 4.20 ผลการทดสอบแบบจำลองนาอีฟเบย์ ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	**6**
Negative	4	0	2	**6**
Neutral	2	0	6	**8**
รวม	**9**	**0**	**11**	**20**

จากตาราง 4.20 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 45%
2. ค่าความแม่นยำ 31.82%
3. ค่าความระลึก 45%

4.3.2.3 ผลจากแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด คือ n_neighbors = 3 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.21 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 4 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 0 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 5 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง

ตารางที่ 4.21 ผลการทดสอบแบบจำลองการหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	**Positive**	**Negative**	**Neutral**	**รวม**
Positive	4	0	2	**6**
Negative	5	0	1	**6**
Neutral	3	0	5	**8**
รวม	**12**	**0**	**8**	**20**

จากตาราง 4.21 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 45%
2. ค่าความแม่นยำ 35%
3. ค่าความระลึก 45%

4.3.2.4 ผลจากแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน (SVM)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน คือ C = 10, gamma = scale และ kernel = linear โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.22 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 7 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 1 ครั้ง

ตารางที่ 4.22 ผลการทดสอบแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	6
Negative	2	3	1	6
Neutral	1	0	7	8
รวม	6	3	11	20

จากตาราง 4.22 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 65%
2. ค่าความแม่นยำ 70.45%
3. ค่าความระลึก 65%

4.3.2.5 ผลจากแบบจำลองป่าสุ่ม (Random Forest)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองป่าสุ่ม คือ max_depth = 20 และ n_estimators = 300โดยหลังจาก

ปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของป่าสุ่ม ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.23 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 2 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 4 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 8 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 0 ครั้ง

ตารางที่ 4.23 ผลการทดสอบแบบจำลองป่าสุ่ม ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	**6**
Negative	0	2	4	**6**
Neutral	0	0	8	**8**
รวม	**3**	**2**	**15**	**20**

จากตาราง 4.23 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 65%
2. ค่าความแม่นยำ 81.33%
3. ค่าความระลึก 65%

4.3.2.6 ผลจากแบบจำลองต้นไม้ตัดสินใจ (Decision Tree)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองต้นไม้ตัดสินใจ คือ max_depth = None โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.24 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 2 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 4 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 2 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 4 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 7 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 1 ครั้ง

ตารางที่ 4.24 ผลการทดสอบแบบจำลองต้นไม้ตัดสินใจ ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	2	2	2	**6**
Negative	0	2	4	**6**
Neutral	1	0	7	**8**
รวม	**3**	**4**	**13**	**20**

จากตาราง 4.24 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 55%
2. ค่าความแม่นยำ 56.84%
3. ค่าความระลึก 51.39%

4.3.2.7 ผลจากแบบจำลองตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นคือ hidden_layer_sizes = (50, 100) โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของแบบโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.25 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง

ตารางที่ 4.25 ผลการทดสอบแบบจำลองแบบโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	**6**
Negative	1	3	2	**6**
Neutral	2	0	6	**8**
รวม	**6**	**3**	**11**	**20**

จากตาราง 4.25 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 60%
2. ค่าความแม่นยำ 66.82%
3. ค่าความระลึก 60%

4.3.2.8 ผลจากแบบจำลองเพอร์เซปตรอน (Perceptron)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองเพอร์เซปตรอน คือ alpha = 0.0001 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของแบบเพอร์เซปตรอน ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.26 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง

ตารางที่ 4.26 ผลการทดสอบแบบจำลองเพอร์เซปตรอน ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	6
Negative	0	3	3	6
Neutral	1	1	6	8
รวม	4	4	12	20

จากตาราง 4.26 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 60%
2. ค่าความแม่นยำ 65%
3. ค่าความระลึก 60%

4.3.2.9 ผลจากแบบจำลองสโตแคสติกการ์เดียนดิเซนท์ (SGD)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองการลดความชันแบบสุ่ม คือ alpha = 0.001 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.27 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 4 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง

ตารางที่ 4.27 ผลการทดสอบแบบจำลองการลดความชันแบบสุ่ม ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	6
Negative	1	4	1	6
Neutral	2	0	6	8
รวม	**3**	**4**	**10**	**20**

จากตาราง 4.27 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 65%
2. ค่าความแม่นยำ 69%
3. ค่าความระลึก 65%

4.3.2.10 ผลจากแบบจำลองพาสซีฟ อากัสซีฟ (Passive Aggressive)

จากผลในการหาพารามิเตอร์ที่ดีที่สุดของไลบรารี่ GridSearchCV ค่าพารามิเตอร์ที่ใช้ในการเพิ่มประสิทธิภาพแบบจำลองพาสซีฟ อากัสซีฟ คือ C = 1 โดยหลังจากปรับพารามิเตอร์แล้วได้ผลผลการทดสอบประสิทธิภาพของพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบสามารถแสดงได้ดังตาราง 4.28 พบว่าจากหัวข้อข่าวสำหรับทดสอบ 20 หัวข้อหัวข้อข่าวผลดังนี้

จำนวนข้อความที่เป็น Positive จริงและทำนายว่าเป็น Positive = 3 ครั้ง
จำนวนข้อความที่เป็น Positive จริง แต่ทำนายว่าเป็นอย่างอื่น = 3 ครั้ง
จำนวนข้อความที่เป็น Negative จริงและทำนายว่าเป็น Negative = 4 ครั้ง
จำนวนข้อความที่เป็น Negative จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง
จำนวนข้อความที่เป็น Neutral จริงและทำนายว่าเป็น Neutral = 6 ครั้ง
จำนวนข้อความที่เป็น Neutral จริง แต่ทำนายว่าเป็นอย่างอื่น = 2 ครั้ง

ตารางที่ 4.28 ผลการทดสอบแบบจำลองพาสซีฟ อากัสซีฟ ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	Positive	Negative	Neutral	รวม
Positive	3	0	3	**6**
Negative	1	4	1	**6**
Neutral	1	1	6	**8**
รวม	**5**	**5**	**10**	**20**

จากตาราง 4.28 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้อง 65%
2. ค่าความแม่นยำ 66%
3. ค่าความระลึก 65%

4.4 ผลการเปรียบเทียบประสิทธิภาพของแบบจำลอง

จากผลการคำนวณตัววัดประสิทธิภาพของแบบจำลองในหัวข้อ 4.3 นำผลที่ได้มาเปรียบเทียบกันโดยดูจากตัววัดประสิทธิภาพทั้ง 4 ตัว คือ ความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) ได้ความว่าตัวตัดคำ PythaiNLP ชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นส่วนชุดข้อมูลทดสอบของแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) มีค่าประสิทธิภาพสูงที่สุด อยู่ที่ค่าความถูกต้องร้อยละ 62.50 ค่าความแม่นยำร้อยละ 67.29 ค่าความระลึกร้อยละ 62.50 และค่าความถ่วงดุลร้อยละ 60.31 ตามตารางที่ 4.29

ตัวตัดคำ PythaiNLP ชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น ส่วนชุดข้อมูลทดสอบของแบบจำลองป่าสุ่ม (Random Forest) มีค่าประสิทธิภาพสูงที่สุด อยู่ที่ค่าความถูกต้องร้อยละ 65 ค่าความแม่นยำร้อยละ 81.33 ค่าความระลึกร้อยละ 65 และค่าความถ่วงดุลร้อยละ 62.83 ตามตารางที่ 4.30

ค่าประสิทธิภาพสูงที่สุดของหุ้นทั้ง 2 ตัวเป็นแบบจำลองที่ใช้เทคนิคการเรียนรู้ของเครื่อง (Machine Learning) เนื่องจากแบบจำลองเทคนิคนี้มีความเหมาะสมกับชุดข้อมูลที่มีขนาดเล็กมากกว่าเทคนิคการเรียนรู้เชิงลึก (Deep Learning)

ตารางที่ 4.29 ค่าประสิทธิภาพของแบบจำลองตัวตัดคำ PythaiNLP บริษัท ราช กรุ๊ป จำกัด (มหาชน)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้น	Logistic Regression	52.08	54.06	52.08	52.27
2		Naive Bayes	56.25	54.98	56.25	51.79
3		KNN	42.71	44.36	42.71	43.28
4		SVM	62.50	67.29	62.50	60.31
5		Random Forest	53.13	61.54	53.13	53.13
6		Decision Tree	53.13	61.95	53.13	53.81
7		Neural network	52.08	52.91	52.08	52.21
8		Perceptron	50.00	50.35	50.00	50.02
9		Stochastic Gradient Descent	52.08	56.64	52.08	52.95
10		Passive Aggressive	50.00	52.31	50.00	50.20

ตารางที่ 4.30 ค่าประสิทธิภาพของแบบจำลองตัวตัดคำ PythaiNLP บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน)

ลำดับ	ชุดข้อมูล	แบบจำลอง	Accuracy	Precision	Recall	F1-Score
1	ข้อมูลที่เกี่ยวกับบริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น	Logistic Regression	65.00	70.45	65.00	64.47
2		Naive Bayes	45.00	31.82	45.00	37.26
3		KNN	45.00	35.00	45.00	38.33
4		SVM	65.00	70.45	65.00	64.47
5		Random Forest	65.00	81.33	65.00	62.83
6		Decision Tree	55.00	56.54	55.00	52.00
7		Neural network	60.00	66.82	60.00	60.26
8		Perceptron	60.00	65.00	60.00	60.00
9		Stochastic Gradient Descent	65.00	69.00	65.00	65.67
10		Passive Aggressive	65.00	66.00	65.00	64.85

4.5 ผลการนำแบบจำลองไปใช้งาน

จากการนำแบบจำลองที่พัฒนาขึ้นไปทดลองใช้งานอย่างง่ายบน Google Colab สามารถแสดงตัวอย่างการใช้งานได้ดังนี้

4.5.1 การตรวจสอบค่าความสำคัญของคุณลักษณะ (Feature Importances)

การตรวจสอบค่าความสำคัญของคุณลักษณะใช้ในการวัดความสำคัญของแต่ละคุณลักษณะ (Feature) ในการทำนายผลลัพธ์ของโมเดล ค่าความสำคัญเหล่านี้ช่วยให้ทราบว่าคุณลักษณะใดมีผลมากที่สุดหรือน้อยที่สุดในการทำนาย และสามารถใช้ในการวิเคราะห์และตัดสินใจเกี่ยวกับการเลือกคุณลักษณะ (Feature Selection) เพื่อลดความซับซ้อนของโมเดลและปรับปรุงประสิทธิภาพ

```
mymodel = RandomForestClassifier(max_depth=20, n_estimators=300)
mymodel.fit(tf_train_bow, y_train)

# ดึงความสำคัญของคุณลักษณะ
feature_importances = mymodel.feature_importances_
features = tfidf.get_feature_names_out()

# สร้าง DataFrame เพื่อดูความสำคัญของคุณลักษณะ
feature_importance_df = pd.DataFrame({'feature': features, 'importance': feature_importances})

# เรียงลำดับตามความสำคัญมากไปน้อย
feature_importance_df = feature_importance_df.sort_values(by='importance', ascending=False)

print("Top Features:")
print(feature_importance_df.head(10))
```

```
Top Features:
      feature  importance
110        บาท    0.067019
158       ล้าน    0.046054
118         ปี    0.030728
150     รายได้    0.029979
63       กำไร    0.024637
216     ไตรมาส    0.024561
117      บันผล    0.024039
176     อสังหา    0.023866
163       สรุป    0.023270
73   ค่าสูงสุด    0.018704
```

รูปที่ 4.7 ค่าความสำคัญของคุณลักษณะจากแบบจำลอง

4.5.2 การตรวจสอบข้อความจากผู้ใช้

การตรวจสอบข้อความจากผู้ใช้ เป็นการรับค่าข้อความจากผู้ใช้แล้วนำข้อความดังกล่าวไปจำแนกความรู้สึกจากหัวข้อข่าวโดยใช้แบบจำลองที่พัฒนาขึ้น ซึ่งผู้วิจัยทดลองป้อนข้อความเพื่อให้แบบจำลองทำการจำแนกโดยป้อนคำว่า “เผยรายได้ไตรมาส3กำไรหด” แบบจำลองจำแนกว่า “Neg” ดังรูปที่ 4.8

```
# ทดสอบกับข้อความที่รับเข้ามา (1)

my_text = input("ข้อความที่ต้องการตรวจสอบ: ")
my_text = text_process(my_text)
my_text = stopword(my_text)
my_tokens = text_process(my_text)
my_bow = tfidf.transform(pd.Series([my_tokens]))
my_predictions = mymodel.predict(my_bow)
my_predictions
```

```
ข้อความที่ต้องการตรวจสอบ:
ข้อความที่ต้องการตรวจสอบ: เผยรายได้ไตรมาส3กำไรหด
ข้อความที่ต้องการตรวจสอบ: เผยรายได้ไตรมาส3กำไรหด
array(['neg'], dtype=object)
```

รูปที่ 4.8 การตรวจสอบข้อความจากผู้ใช้

4.6 อภิปรายผล

งานวิจัยนี้มีวัตถุประสงค์ในการสร้างแบบจำลองการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นไทยรายวัน โดยทำการเปรียบเทียบรูปแบบการเก็บข้อมูล 3 แบบ (A - C) และการเตรียมข้อความข่าวด้วยตัวตัดคำภาษาไทย 2 แบบ คือ PyThaiNLP และ DeepCut (I - II) ผลการศึกษาสามารถแบ่งออกเป็น 6 การทดลอง เพื่อเลือกวิธีการเตรียมข้อความที่เหมาะสมก่อนที่จะนำไปสร้างโมเดลการจำแนก ซึ่งมีการปรับจูนไฮเปอร์พารามิเตอร์โดยใช้ GridSearchCV กับแบบจำลองและเปรียบเทียบประสิทธิภาพเพื่อเลือกแบบจำลองที่เหมาะสมที่สุดสำหรับหุ้นแต่ละตัว การเลือกหุ้น 2 ตัวจากกลุ่ม SET50 คือ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) และบริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เป็นตัวอย่างในการศึกษา การเก็บข้อมูลทั้งหัวข้อข่าวและราคาหุ้นรายวันตั้งแต่เดือนมกราคม พ.ศ. 2561 ถึงเดือนธันวาคม พ.ศ. 2565 แสดงถึงการรวบรวมข้อมูลที่ครอบคลุมและเพียงพอสำหรับการวิเคราะห์ ข้อมูลที่รวบรวมมาได้ผ่านกระบวนการเตรียมข้อมูล เช่น การลบเครื่องหมายวรรคตอน สัญลักษณ์ และตัวเลข การนอร์มอลไลซ์ตัวอักษร การตัดคำ และการสกัดคุณลักษณะ ซึ่งเป็นขั้นตอนสำคัญในการปรับปรุงความแม่นยำของโมเดล

ผลการวิจัยพบว่า การเก็บข้อความเฉพาะหัวข้อข่าวของหุ้นที่สนใจ (รูปแบบ A) และการใช้ตัวตัดคำ PyThaiNLP (รูปแบบ I) ให้ผลลัพธ์การจำแนกที่ดีที่สุดในหุ้นทั้งสองตัว การแบ่งชุดข้อมูลออกเป็นชุดข้อมูลเรียนรู้ (70%) และชุดข้อมูลทดสอบ (30%) ช่วยให้สามารถปรับจูนพารามิเตอร์ของแบบจำลองได้อย่างเหมาะสม ซึ่งแสดงถึงความสำคัญของการแบ่งชุดข้อมูลอย่างมีประสิทธิภาพเพื่อปรับปรุงการทำงานของแบบจำลองสำหรับหุ้น บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) แบบจำลองซัพพอร์ตเวกเตอร์แมชชีน (SVM) มีประสิทธิภาพสูงสุด โดยมีค่าความถูกต้องร้อยละ 62.50 ค่าความแม่นยำร้อยละ 67.29 และค่าความระลึกร้อยละ 62.50 สำหรับหุ้น บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) แบบจำลองป่าสุ่ม (Random Forest) มีประสิทธิภาพสูงสุด โดยมีค่าความถูกต้องร้อยละ 65 ค่าความแม่นยำร้อยละ 81.33 และค่าความระลึกร้อยละ 65 ผลลัพธ์เหล่านี้ชี้ให้เห็นถึงความแตกต่างในประสิทธิภาพของแบบจำลองเมื่อใช้งานกับข้อมูลของหุ้นแต่ละตัวและผลการวิจัยยังแสดงให้เห็นด้วยว่าการปรับจูนไฮเปอร์พารามิเตอร์เป็นสิ่งสำคัญที่ทำให้ประสิทธิของแบบจำลองสูงขึ้นได้ ซึ่งในงานวิจัยที่เกี่ยวข้องของวิกานดาและอัญชนาถ้ามีการปรับจูนไฮเปอร์พารามิเตอร์อาจได้ค่าประสิทธิภาพของแบบจำลองที่สูงขึ้นอีกด้วย

การอภิปรายผลนี้เน้นให้เห็นถึงความสำคัญของการเลือกวิธีการเตรียมข้อมูลและการใช้เครื่องมือที่เหมาะสมในการพัฒนาโมเดลการจำแนกความรู้สึก นอกจากนี้ ยังชี้ให้เห็นว่าการเลือกใช้แบบจำลองที่เหมาะสมสามารถช่วยเพิ่มความแม่นยำและประสิทธิภาพของการทำนายได้อย่างมาก ทั้งนี้ การวิจัยในอนาคตควรพิจารณาขยายขอบเขตของชุดข้อมูล รวมถึงการทดลองกับหุ้นเพิ่มเติมและการใช้เทคนิคการเตรียมข้อมูลและการจำแนกที่หลากหลายมากขึ้นเพื่อปรับปรุงความแม่นยำและประสิทธิภาพของโมเดลต่อไป

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

ในบทนี้จะทำการสรุปผลการดำเนินงานทั้งหมด ตลอดจนนำเสนอข้อเสนอแนะ เพื่อใช้สำหรับงานวิจัยในอนาคตแก่ผู้วิจัยหรือผู้ที่ต้องการศึกษาค้นคว้า โดยการสรุปผลจะอ้างอิงผลการดำเนินงานการค้นคว้าอิสระในบทที่ 4 โดยมีรายละเอียดดังนี้

5.1 สรุปผลการวิจัย

5.2 ข้อเสนอแนะ

5.1 สรุปผลการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาและพัฒนาแบบจำลองในการจำแนกการวิเคราะห์ความรู้สึกจากหัวข้อข่าวหุ้นรายวันรวมถึงประเมินผลประสิทธิภาพของแบบจำลองที่พัฒนาขึ้น ซึ่งผู้วิจัยได้ทำการรวบรวมข้อมูลหัวข้อในงานวิจัยนี้จะมีการเก็บข้อมูลอยู่ทั้งหมด 3 รูปแบบต่อหุ้น 1 ตัว คือแบบ A คือ การเก็บข้อมูลหัวข้อข่าว โดยใช้ ชื่อย่อหลักทรัพย์ เท่านั้น แบบ B คือ การเก็บข้อมูลหัวข้อข่าวโดยใช้ชื่อย่อหลักทรัพย์และคำที่เกี่ยวข้องกับธุรกิจของหลักทรัพย์ แบบ C คือการเก็บข้อมูลหัวข้อข่าวโดยใช้ชื่อย่อหลักทรัพย์ คำที่เกี่ยวข้องกับธุรกิจของหลักทรัพย์และหัวข้อข่าวเศรษฐกิจภายในและนอกประเทศ ทำการรวบรวมข้อมูลบน Google Colab ด้วยภาษา Python ตั้งแต่เดือนเดือนมกราคม พ.ศ. 2561 จนถึงเดือนธันวาคม พ.ศ. 2565 หลังจากได้ข้อมูลมาแล้วก็นำข้อมูลที่รวบรวมได้มาทำการเตรียมข้อมูล (Data Preparation) การลบเครื่องหมายวรรคตอน สัญลักษณ์ และ ตัวเลข การนอร์มอลไลซ์ตัวอักษร การตัดคำ โดยในส่วนการตัดคำนั้นจะมีการใช้ตัวตัดคำ 2 แบบคือแบบ I ใช้ PythaiNLP ในการตัดคำข้อความหัวข้อข่าว แบบ II ใช้ DeepCut ในการตัดคำข้อความหัวข้อข่าว ต่อมาก็จะเป็นการลบกลุ่มคำที่ไม่มีความหมายและการสกัดคุณลักษณะ เพื่อทำให้ข้อมูลอยู่ในรูปแบบที่คอมพิวเตอร์สามารถประมวลผลได้ แล้วแบ่งข้อมูลออกเป็น 2 ชุดในอัตราส่วน 70 : 30 พร้อมกำหนดให้เป็นชุดข้อมูลเรียนรู้ (Training Data) เพื่อใช้ในการพัฒนาแบบจำลองจำนวน และชุดข้อมูลทดสอบ (Test Data) เพื่อใช้ในการวัดประสิทธิภาพของแบบจำลองที่พัฒนาขึ้น

ก่อนจะทำการพัฒนาแบบจำลองพยากรณ์ทิศทางของราคาด้วยชุดข้อมูลเรียนรู้ (Training Data) โดยใช้แบบจำลอง 10 แบบคือ การถดถอยเชิงโลจิสติค (Logistic Regression) นาอีฟเบย์ (Naïve Bayes) การหาค่าด้วยเพื่อนบ้านที่ใกล้ที่สุด k จุด (K-Nearest Neighbors) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) ป่าสุ่ม (Random Forest) ต้นไม้ตัดสินใจ (Decision Tree) ตัวจำแนกแบบโครงข่ายประสาทเทียมหลายชั้น (MLP) เพอร์เซปตรอน (Perceptron) สโตแคสติกการ์เดียนดิเซนท์ (SGD) พาสซีฟ อากัสซีฟ (Passive Aggressive) แล้วทำการประเมินประสิทธิภาพของแบบจำลองทั้งในด้านค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก

(Recall) และค่าความถ่วงดุล (F-Measure) ด้วยชุดข้อมูลทดสอบ (Test Data) โดยจะแยกการเปรียบเทียบออกเป็น 2 การเปรียบเทียบคือ เปรียบเทียบตัวตัดคำว่าตัวไหนมีประสิทธิภาพสูงที่สุดกับชุดข้อมูลแบบไหน จากผลค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) จากผลของการการจำแนกความรู้สึกจากหัวข้อข่าวหุ้นรายวันตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นมีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงกว่าตัวตัดคำกับชุดข้อมูลอื่น และตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นมีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงกว่าตัวตัดคำกับชุดข้อมูลอื่น

ผู้วิจัยจึงนำตัวตัดคำPythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นและตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้นมาทำการเพิ่มประสิทธิภาพของแบบจำลองด้วยการปรับค่าพารามิเตอร์แบบจำลองทั้ง 10 แบบจำลองเพื่อนำค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) มาเปรียบเทียบต่อ สรุปผลออกมาได้ว่าตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ราช กรุ๊ป จำกัด (มหาชน) (RATCH) เท่านั้นแบบจำลองซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machines) มีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงที่สุดที่ร้อยละ 62.50 ร้อยละ 67.29 ร้อยละ 62.50 ตามลำดับและตัวตัดคำ PythaiNLP กับชุดข้อมูลที่เกี่ยวกับ บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน) (QH) เท่านั้น แบบจำลองป่าสุ่ม (Random Forest) เมื่อมีการเพิ่มประสิทธิภาพแล้วมีค่าความถูกต้อง (Accuracy) ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) สูงที่สุดที่ร้อยละ 65 ร้อยละ 81.33 ร้อยละ 65 ตามลำดับ

จากหุ้นทั้งสองตัวที่นำมาทำการทดลองเป็นไปตามสมมติฐานที่ตั้งไว้ว่าผลการประเมินประสิทธิภาพการจำแนกความรู้สึกจากข่าวจะมีค่าความถูกต้องมากกว่าร้อยละ 60 และแบบจำลองที่ได้ค่าประสิทธิภาพสูงที่สุดเป็นแบบจำลองเทคนิคการเรียนรู้ของเครื่อง (Machine Learning) เนื่องจากแบบจำลองเทคนิคนี้มีความเหมาะสมกับชุดข้อมูลที่มีขนาดเล็กมากกว่าเทคนิค การเรียนรู้เชิงลึก (Deep Learning) (Data-Cowboy, 2024)

5.2 ข้อเสนอแนะ

จากผลการดำเนินการค้นคว้าอิสระครั้งนี้ สามารถนำไปเพิ่มประสิทธิภาพแบบจำลองเพื่อการจำแนกแนวโน้มของราคาหุ้นไทยรายวันโดยใช้การวิเคราะห์ความรู้สึกจากข่าวให้มีประสิทธิภาพมากขึ้น ดังนี้

1) การใช้ตัวตัดคำภาษาไทยงานวิจัยใช้ตัวตัดคำ PythaiNLP และ DeepCut แนะนำการใช้เทคโนโลยีการตัดคำที่พัฒนาขึ้นใหม่ๆ เพิ่มเติม

2) การนำผลที่ได้จากการจำแนกความรู้สึกไปใช้ร่วมกับแบบจำลองการทำนายราคา สามารถใช้แบบจำลองการจำแนกความรู้สึกในเข้าไปช่วยในการทำนายราคาหุ้นได้
3) การปรับจูนพารามิเตอร์ของแบบจำลอง นำไปเพิ่มการทดลองเพื่อหาแนวทางในการปรับจูนพารามิเตอร์ให้ดีขึ้น
4) การขยายขอบเขตของชุดข้อมูลกับหุ้นเพิ่มเติมเพื่อเพิ่มความน่าเชื่อถือของผลลัพธ์ อาจเพิ่มข่าวจากแหล่งอื่นหรือนำข้อมูลความคิดเห็นส่วนบุคคลของหุ้นแต่ละตัวเข้ามาร่วมด้วย

เอกสารอ้างอิง

กาญจนา กิจบำรุงรัตน์. 2564. ‘การใช้ Social Listening Tools ในการพยากรณ์ความคิดเห็นของคนไทยเกี่ยวกับการซื้อสินค้าออนไลน์ผ่านแอปพลิเคชัน โดยการใช้การทำเหมืองข้อความ’, มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ.

การ์ตูน. 2566. ‘ปัจจัยสำคัญ ที่นักลงทุนไม่ควรมองข้าม ก่อนเลือกซื้อหุ้น’, สืบค้นเมื่อ 20 มีนาคม 2566, จาก https://stock2morrow.com/webboard/1/b06ef319-c894-4e33-a785-efbff482e9e9

ฐิติเมธ โภคชัย. 2565. ‘9 หุ้นปลอดภัย อุ่นใจเมื่อมีในพอร์ต’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.setinvestnow.com/th/knowledge/article/186-tsi-9-defensive-stocks-that-are-nice-to-have-in-your-portfolio

ตลาดหลักทรัพย์แห่งประเทศไทย. 2565. ‘บริษัท ควอลิตี้เฮ้าส์ จำกัด (มหาชน)’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.set.or.th/th/market/product/stock/quote/QH/company-profile/information

ตลาดหลักทรัพย์แห่งประเทศไทย. 2565. ‘บริษัท ราช กรุ๊ป จำกัด (มหาชน)’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.set.or.th/th/market/product/stock/quote/RATCH/company-profile/information

ท๊อปไลน์เนอร์. 2564. ‘แต่ละสินทรัพย์การเงิน ให้ผลตอบแทนเท่าไร?’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.finnomena.com/topliner/financial-assets-return

ธนภัทร์ คุ้มสุภา. 2559. ‘การจำแนกประเภทข้อความในภาษาไทยโดยใช้ นิวรอลเน็ตเวิร์กคอนโวลูชั่นระดับตัวอักษร’, จุฬาลงกรณ์ มหาวิทยาลัย.

นิธิกร เลิศชาญวุฒิ. 2564. ‘การวิเคราะห์ความสนใจของนักท่องเที่ยวด้วยเทคนิคการเรียนรู้ของเครื่อง : กรณีศึกษา ถนนเยาวราช ประเทศไทย’, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

ปกป้อง สังวรกาญจน์. 2564 ‘การจำแนกความคิดเห็นที่เกี่ยวข้องกับความมั่นคง’, มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ.

พนิดา กาศกลางดอน, โสภณ มงคลลักษมี, ศิริสรรพ เหล่าหะเกียรติ และรัตน์ชัยนันท์ ธรรมสุจรติ. 2564. ‘การวิเคราะห์ประสบการณ์จากการใช้บริการโรงพยาบาลในประเทศไทย จากความคิดเห็นของผู้ใช้บริการ’, มหาวิทยาลัยศรีนครินทรวิโรฒ.

พิศิษฐ์ บวรเลิศสุธี และวรภัทร ไพรีเกรง. 2565. ‘ตัวแบบการวิเคราะห์ความรู้สึกทางอารมณ์ สำหรับจำแนกประเภทบทความแนะนำสินค้าออนไลน์’, มหาวิทยาลัยธุรกิจบัณฑิตย์.

ฟีโนมีน่า. 2563. ‘การมาเยือนของ “เมื่อโรคระบาดมา ตลาดหุ้นจะตอบสนองอย่างไร?’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.finnomena.com/finnomena-ic/corona-outbreak-stock

ลงทุนศาสตร์. 2564. ‘วันที่เลวร้ายที่สุดในประวัติศาสตร์ตลาดหุ้น’, สืบค้นเมื่อ 25 ธันวาคม 2565, จาก https://www.investerest.co/economy/black-monday/

วิกานดา ผาพันธ์ และอัญชนา พิมพิศาล. 2563. ‘การพยากรณ์ทิศทางของราคาหุ้นรายวันจากข้อความข่าวภาษาไทย โดยใช้วิธีการประมวลผลภาษาธรรมชาติ’, มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ.

วรเทพ อาจารยางกูร. 2560. ‘การวิเคราะห์ข้อความและการจำแนกความรู้สึก’, จุฬาลงกรณ์มหาวิทยาลัย.

ศุภณัฐ ก้อนศิลา. 2566. ‘การพยากรณ์ทิศทางการเปลี่ยนแปลงราคาหุ้นในกลุ่ม SET50 โดยใช้ปัจจัยข่าว และการวิเคราะห์เชิงเทคนิค’, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

Bipin Aasi, Syeda Aniqa Imtiaz, Hamzah Arif Qadeer, Magdalean Singarajah, Rasha Kashef. 2021. ‘Stock Price Prediction Using a Multivariate Multistep LSTM: A Sentiment and Public Engagement Analysis Model’, In 2021 IEEE International IOT Electronics and Mechatronics Conference (IEMTRONICS).

Data-Cowboys. 2024. ‘Which Machine Learning Classifiers are Best for Small Datasets?’ Accessed 5 May 2024, https://www.data-cowboys.com/blog/which-machine-learning-classifiers-are-best-for-small-datasets

De Brabanter, Karsmakers, Ojeda, Alzate, Pelckmans, De Moor,Vandewalle, Suykens. 2011. ‘LS-SVMlab Toolbox User’s Guide’, 19.

Heimerl, F., Lohmann, S., Lange, S. 2014. ‘Text Analytics Based on Word Clouds’, In Proceedings of the 47th Hawaii International Conference on System Sciences.

Javapoint. 2024. ‘Perceptron’ Accessed 10 April 2024, https://en.wikipedia.org/wiki/Support_vector_machine

Jason. 2020. ‘Train-Test Split for Evaluating Machine Learning Algorithms’ Accessed 5 May 2024, https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

Kotsiantis. 2007. ‘Supervised Machine Learning: A Review of Classification Techniques’, 31, 249-268.

Li, X., Huang, X., Deng, X., & Zhu, S. 2014. ‘Enhancing Quantitative Intra-day Stock Return Prediction by Integrating both Market News and Stock Prices’, 142, 228-238.

Naji Mordi Naji Al-Dosary, Saad Abdulrahman Al-Hamed, Abdulwahed Mohamed Aboukarima. 2019. ‘K-Nearest Neighbors method for prediction of fuel consumption in tractor–chisel plow system, 39, 729-736.

Nektarios, T.G., 2013. ‘Weka Classify Summary’, Accessed 5 May 2024, https://www.academia.edu/5167325/Weka_Classifiers_Summary

Quinlan. 1986. ‘Induction of Decision Trees Machine Learning 1’, 81-106.

Rakpong Kittinaradorn, Korakot Chaovavanich, Titipat Achakulvisut, Kittinan Srithaworn, Pattarawat Chormai, Chanwit Kaewkasi, Tulakan Ruangrong, Krichkorn Oparad. 2019. ‘DeepCut: A Thai Word Tokenizer using Deep Learning’, Accessed 30 May 2024, https://zenodo.org/records/3457707

Salini, A, U Jeyapriya and SM College. 2018. 'A Majority Vote Based Ensemble Classifier for Predicting Students Academic Performance', International Journal of Pure and Applied Mathematics, 118, 24.

Wannaphong Phatthiyaphaibun, Korakot Chaovavanich, Charin Polpanumas, Arthit Suriyawongkul, Lalita Lowphansirikul, Pattarawat Chormai, Peerat Limkonchotiwat, Thanathip Suntorntip, Can Udomcharoenchaikit. 2023. ‘PyThaiNLP: Thai Natural Language Processing in Python’, In Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023).

Zack. 2023. ‘Support vector machine’ Accessed 2 December 2023, https://en.wikipedia.org/wiki/Support_vector_machine

ภาคผนวก ก

โปรแกรมไพธอนที่ใช้ในการทดลอง

โปรแกรมจะแบ่งเป็น 4 แบบในการรันโปรแกรม โดยแบ่งจาก ประเภทชุดข้อมูล ตัวตัดคำ และตัวหุ้น

1. หัวข้อข่าวหุ้น RATCH เท่านั้น ตัวตัดคำ PythaiNLP
2. หัวข้อข่าวหุ้น RATCH เท่านั้น ตัวตัดคำ DeepCut
3. หัวข้อข่าวหุ้น QH เท่านั้น ตัวตัดคำ PythaiNLP
4. หัวข้อข่าวหุ้น QH เท่านั้น ตัวตัดคำ DeepCut

1.หัวข้อข่าวหุ้น RATCH เท่านั้น ตัวตัดคำ PythaiNLP

```
# Install Library
!pip install pythainlp
!pip install wordcloud
!pip install python-crfsuite
!pip install matplotlib-venn
!pip install stop_words

from google.colab import drive
drive.mount('/content/drive')

import pandas as pd
import pythainlp


from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
GradientBoostingClassifier,RandomForestClassifier,AdaBoostClassifier,BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier



import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.feature_extraction.text import CountVectorizer

import re
```

```
import string

from pythainlp.corpus.common import thai_stopwords
import warnings
warnings.filterwarnings(action='ignore')

import numpy as np
from PIL import Image
from wordcloud import WordCloud

from pythainlp.tokenize import word_tokenize

from pythainlp.util import find_keyword
from pythainlp.util import rank
from wordcloud import WordCloud
import matplotlib
import matplotlib.font_manager as fm
fm.fontManager.addfont('/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf')
matplotlib.rc('font',family='TH Sarabun New',size =10)

# Import Library

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import string
import re
from pythainlp.util import find_keyword
from pythainlp.util import rank
import pythainlp
from pythainlp.tokenize import word_tokenize
from pythainlp.corpus import thai_stopwords
```

```
from pythainlp.corpus import wordnet

import nltk
nltk.download('stopwords')
from stop_words import get_stop_words
from nltk.corpus import stopwords

# นำเข้าข้อมูลจาก Github (เรียกใช้งานด้วยข้อมูลเก่าที่เคยทำ)
df = pd.read_csv("/content/drive/MyDrive/Data/data project/ratch_clean.csv")
df.head()

df.dropna(inplace=True)

# ดูว่ามีข้อมูลเกี่ยวไม่เกี่ยวว่ามีกี่ข้อมูล

df['sentiment'].value_counts()

# เปรียบเทียบจำนวนของข้อมูล

df['sentiment'].value_counts().plot.bar()

# ทำการตัดคำ

from pythainlp import word_tokenize

def text_process(text):
    final = "".join(u for u in text if u not in ("’","?", ";", ":", "!",'/',"–          ",'.', '"','”','“', "ๆ", "ฯ",
"=", "_", "(", ")","%", "&", "", ",", "+", "*", "[", "]", "#", "฿", "$", "@","
๙","1","2","3",'4','5','6','7','8','9','0','-'))
    final = word_tokenize(final)
    final = " ".join(word for word in final)
    return final
df['title_tokens'] = df['title'].apply(text_process)
```

```
df

from pythainlp.corpus.common import thai_stopwords

# ทำการกำจัดคำหยุด pythai NLP

from pythainlp.corpus.common import thai_stopwords

thai_stopwords = list(thai_stopwords())

def stopword(text):
    list_word = word_tokenize(text, keep_whitespace= False) #, engine='newmm')

    stopwords = thai_stopwords
    final = [i for i in list_word if i not in stopwords]
    return final

df['del_stopword'] = df['title_tokens'].apply(stopword)

detokenized_doc = []
for i in range(len(df)):
    t = ' '.join(map(str, df['del_stopword'][i]))
    detokenized_doc.append(t)

df['del_stopword'] = detokenized_doc
df

df.to_csv("stopword.csv")

df0 = df[df['sentiment']=='pos']

df1 =df[df['sentiment']=='neg']
```

```
df2 =df[df['sentiment']=='nue']

def get_text_list(message):
  tokenized =[]
  for i in message:
    token = word_tokenize(i)
    for j in token:
      tokenized.append(j)

  return tokenized

text_list = get_text_list(df1.del_stopword)
word_count = pd.DataFrame([find_keyword(text_list, min_len=3)]).T

word_count['word'] = word_count.index
word_count.columns = ['count','word']
word_count.index = range(len(word_count))

word_count.sort_values(by='count',ascending =False,inplace=True)
word_count

word_count.iloc[:35,:]

from tqdm import tqdm

def get_text_str(message):
  tokenized =" "
  th_stw1 = thai_stopwords
  for i in tqdm(message):
    token = word_tokenize(i)
    for j in token:
      if j not in thai_stopwords:
        tokenized =tokenized + " " + j
```

```
  return tokenized


text_str = get_text_str(df0.del_stopword)

path = 'THSarabunNew.ttf'
regexp = r"[ก-๙a-zA-Z']+"
wordcloud = WordCloud(
                    font_path='/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf',
                    min_font_size=1,
                    background_color="white",
                    width=400,
                    height=200,
                    max_words=1000,
                    colormap='plasma',
                    scale=3,
                    font_step=4,
                   #   contour_width=3,
                    contour_color='steelblue',
                    collocations=False,
                    regexp=regexp,
                    margin=2
                    ).generate(text_str)

fig, ax = plt.subplots(1, 1, figsize=(16, 12))
ax.imshow(wordcloud, interpolation='bilinear')
ax.axis("off")
fig.show()

# แบ่งข้อมูลเป็น train 70 test 30 แบบสุ่ม pythaiNLP
```

```
from sklearn.model_selection import train_test_split

X = df[['del_stopword']]  #  โจทย์
y = df['sentiment']  #  เฉลย

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1234,shuffle =True)

# นับจำนวนคำเพื่อทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(analyzer=lambda x:x.split(' '))
tfidf.fit_transform(df['del_stopword'])
tfidf.vocabulary_

# ทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer PythaiNLP

tf_train_bow = tfidf.transform(X_train['del_stopword'])
pd.DataFrame(tf_train_bow.toarray(), columns=tfidf.get_feature_names_out(),
index=X_train['del_stopword'])
pdtemp = pd.DataFrame(tf_train_bow.toarray(),
columns=tfidf.get_feature_names_out(), index=X_train['del_stopword'])

from sklearn.linear_model import LogisticRegression, SGDClassifier,
PassiveAggressiveClassifier, Perceptron
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix
import numpy as np
import pandas as pd

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression'],
    [MultinomialNB(), 'Naive Bayes'],
    [KNeighborsClassifier(), 'KNN'],
    [SVC(), 'SVM'],
    [RandomForestClassifier(), 'Random Forest'],
    [DecisionTreeClassifier(), 'Decision Tree'],
    [MLPClassifier(), 'Neural network'],
    [Perceptron(), 'Perceptron'],
    [SGDClassifier(), 'Stochastic Gradient Descent'],
    [PassiveAggressiveClassifier(), 'Passive Aggressive'],
]

model_tf_score = []
for a in compare_model_tf:
    compare_tf = a[0]
    compare_tf.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['del_stopword'])
    tf_test_predictions = compare_tf.predict(tf_test_bow)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    # คำนวณ confusion matrix
    tf_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)
```

```
    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(tf_confusion_matrix)
    FP = tf_confusion_matrix.sum(axis=0) - TP
    FN = tf_confusion_matrix.sum(axis=1) - TP
    TN = tf_confusion_matrix.sum() - (TP + FP + FN)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / tf_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print("Confusion Matrix for the Best Model (Highest Accuracy):")
    print(tf_confusion_matrix)
    print('-' * 55)

# แสดงผลสุดท้ายของคะแนนโมเดล
score2 = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1'])
```

```
print(score2)

score2

from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import confusion_matrix

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression', {'C': [0.1, 1, 10, 100]}],
    [MultinomialNB(), 'Naive Bayes', {'alpha': [0.1, 0.5, 1.0]}],
    [KNeighborsClassifier(), 'KNN', {'n_neighbors': [3, 5, 7]}],
    [SVC(), 'SVM', {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf'], 'gamma': ['scale', 'auto']}],
    [RandomForestClassifier(), 'Random Forest', {'n_estimators': [100, 200, 300],
'max_depth': [None, 10, 20]}],
    [DecisionTreeClassifier(), 'Decision Tree', {'max_depth': [None, 10, 20]}],
    [MLPClassifier(), 'Neural network', {'hidden_layer_sizes': [(100,), (50, 50), (50, 100)]}],
    [Perceptron(), 'Perceptron', {'alpha': [0.0001, 0.001, 0.01]}],
    [SGDClassifier(), 'Stochastic Gradient Descent', {'alpha': [0.0001, 0.001, 0.01]}],
    [PassiveAggressiveClassifier(), 'Passive Aggressive', {'C': [0.1, 1, 10]}]
]

model_tf_score = []

for a in compare_model_tf:
    compare_tf = a[0]
    params = a[2]

    # สร้าง GridSearchCV object
    grid_search = GridSearchCV(compare_tf, params, cv=5, scoring='accuracy')

    # Fit โมเดลเพื่อค้นหา hyperparameters ที่ดีที่สุด
    grid_search.fit(tf_train_bow, y_train)
```

```
    # ใช้โมเดลที่ปรับจูน hyperparameters แล้วดีที่สุด
    best_model = grid_search.best_estimator_

    best_model.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['del_stopword'])
    tf_test_predictions = best_model.predict(tf_test_bow)
    best_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(best_confusion_matrix)
    FP = best_confusion_matrix.sum(axis=0) - TP
    FN = best_confusion_matrix.sum(axis=1) - TP
    TN = best_confusion_matrix.sum() - (FP + FN + TP)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / best_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual, grid_search.best_params_])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
```

```
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print("Confusion Matrix for the Model")
    print(best_confusion_matrix)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print('-' * 55)

# แสดงผลการเปรียบเทียบ
score = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1', 'Best
hyperparameters'])
print(score)

score

score3 = pd.concat([score2, score])
score3
```

2.หัวข้อข่าวหุ้น RATCH เท่านั้น ตัวตัดคำ DeepCut

```
# Install Library
!pip install pythainlp
!pip install wordcloud
!pip install python-crfsuite
!pip install matplotlib-venn
!pip install stop_words
from google.colab import drive
drive.mount('/content/drive')

# Import Library
import pandas as pd
import pythainlp
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
GradientBoostingClassifier,RandomForestClassifier,AdaBoostClassifier,BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.feature_extraction.text import CountVectorizer

import re
import string

from pythainlp.corpus.common import thai_stopwords
import warnings
```

```
warnings.filterwarnings(action='ignore')

import numpy as np
from PIL import Image

from pythainlp.tokenize import word_tokenize

from pythainlp.util import find_keyword
from pythainlp.util import rank
from wordcloud import WordCloud
import matplotlib
import matplotlib.font_manager as fm
fm.fontManager.addfont('/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf')
matplotlib.rc('font',family='TH Sarabun New',size =10)

# Import Library

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import string
import re
from pythainlp.util import find_keyword
from pythainlp.util import rank
import pythainlp
from pythainlp.tokenize import word_tokenize
from pythainlp.corpus import thai_stopwords
from pythainlp.corpus import wordnet

import nltk
nltk.download('stopwords')
from stop_words import get_stop_words
```

```
from nltk.corpus import stopwords

# นำเข้าข้อมูลจาก Github (เรียกใช้งานด้วยข้อมูลเก่าที่เคยทำ)
df = pd.read_csv("/content/drive/MyDrive/Data/data project/ratch_1.csv")
df.head()

# ดูว่ามีข้อมูลเกี่ยวไม่เกี่ยวว่ามีกี่ข้อมูล

df['sentiment'].value_counts()

# เปรียบเทียบจำนวนของข้อมูล

df['sentiment'].value_counts().plot.bar()

pip install deepcut

import deepcut
def text_process(text):
    final = "".join(u for u in text if u not in ("?","?", ";", ":", "!","/","–        ",'.',  "","”",'“', "ๆ", "ฯ",
"=", "_", "(", ")","%", "&", "", ",", "+", "*", "[", "]", "#", "฿", "$", "@","
๙","1","2","3",'4','5','6','7','8','9','0','-'))
    final = deepcut.tokenize(final)
    final = " ".join(word for word in final)
    return final
df['title_tokens_deepcut'] = df['title'].apply(text_process)
df

df.to_csv("stopword.csv")

# ทำการกำจัดคำหยุด
from pythainlp import word_tokenize
from pythainlp.corpus.common import thai_stopwords
```

```
thai_stopwords = list(thai_stopwords())

def stopword(text):
    list_word = word_tokenize(text, keep_whitespace= False) #, engine='newmm')

    stopwords = thai_stopwords
    final = [i for i in list_word if i not in stopwords]
    return final

df['del_stopword_deepcut'] = df['title_tokens_deepcut'].apply(stopword)

detokenized_doc = []
for i in range(len(df)):
    t = ' '.join(map(str, df['del_stopword_deepcut'][i]))
    detokenized_doc.append(t)

df['del_stopword_deepcut'] = detokenized_doc
df

df0 = df[df['sentiment']=='pos']

def get_text_list(message):
  tokenized =[]
  for i in message:
    token = word_tokenize(i)
    for j in token:
      tokenized.append(j)

  return tokenized

text_list = get_text_list(df0.del_stopword_deepcut)
word_count = pd.DataFrame([find_keyword(text_list, min_len=3)]).T
```

```
word_count['word'] = word_count.index
word_count.columns = ['count','word']
word_count.index = range(len(word_count))

word_count.sort_values(by='count',ascending =False,inplace=True)
word_count
```

```
word_count.iloc[:35,:]
```

```
from tqdm import tqdm

def get_text_str(message):
  tokenized =" "
  th_stw1 = thai_stopwords
  for i in tqdm(message):
    token = word_tokenize(i)
    for j in token:
      if j not in thai_stopwords:
        tokenized =tokenized + " " + j

  return tokenized


text_str = get_text_str(df0.del_stopword_deepcut)
```

```
path = 'THSarabunNew.ttf'
regexp = r"[ก-๙a-zA-Z']+"
wordcloud = WordCloud(
                    font_path='/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf',
                    min_font_size=1,
                    background_color="white",
                    width=400,
```

```
                height=200,
                max_words=1000,
                colormap='plasma',
                scale=3,
                font_step=4,
              #   contour_width=3,
                contour_color='steelblue',
                collocations=False,
                regexp=regexp,
                margin=2
                ).generate(text_str)

fig, ax = plt.subplots(1, 1, figsize=(16, 12))
ax.imshow(wordcloud, interpolation='bilinear')
ax.axis("off")
fig.show()

# แบ่งข้อมูลเป็น train 70 test 30 แบบสุ่ม DEEPCUT

from sklearn.model_selection import train_test_split

X = df[['title_tokens_deepcut']]  #  โจทย์
y = df['sentiment']  #  เฉลย

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1234,shuffle =True)

# นับจำนวนคำเพื่อทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(analyzer=lambda x:x.split(' '))
tfidf.fit_transform(df['title_tokens_deepcut'])
```

```
tfidf.vocabulary_

# ทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer Deepcut

tf_train_bow = tfidf.transform(X_train['title_tokens_deepcut'])
pd.DataFrame(tf_train_bow.toarray(), columns=tfidf.get_feature_names_out(),
index=X_train['title_tokens_deepcut'])

from sklearn.linear_model import LogisticRegression, SGDClassifier,
PassiveAggressiveClassifier, Perceptron
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix
import numpy as np
import pandas as pd

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression'],
    [MultinomialNB(), 'Naive Bayes'],
    [KNeighborsClassifier(), 'KNN'],
    [SVC(), 'SVM'],
    [RandomForestClassifier(), 'Random Forest'],
    [DecisionTreeClassifier(), 'Decision Tree'],
    [MLPClassifier(), 'Neural network'],
    [Perceptron(), 'Perceptron'],
    [SGDClassifier(), 'Stochastic Gradient Descent'],
    [PassiveAggressiveClassifier(), 'Passive Aggressive'],
]
```

```
model_tf_score = []
for a in compare_model_tf:
    compare_tf = a[0]
    compare_tf.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['title_tokens_deepcut'])
    tf_test_predictions = compare_tf.predict(tf_test_bow)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    # คำนวณ confusion matrix
    tf_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(tf_confusion_matrix)
    FP = tf_confusion_matrix.sum(axis=0) - TP
    FN = tf_confusion_matrix.sum(axis=1) - TP
    TN = tf_confusion_matrix.sum() - (TP + FP + FN)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / tf_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual])

    # แสดงผล
```

```
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print("Confusion Matrix for the Best Model (Highest Accuracy):")
    print(tf_confusion_matrix)
    print('-' * 55)

# แสดงผลสุดท้ายของคะแนนโมเดล
score2 = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1'])
print(score2)

score2

from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import confusion_matrix

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression', {'C': [0.1, 1, 10, 100]}],
    [MultinomialNB(), 'Naive Bayes', {'alpha': [0.1, 0.5, 1.0]}],
    [KNeighborsClassifier(), 'KNN', {'n_neighbors': [3, 5, 7]}],
    [SVC(), 'SVM', {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf'], 'gamma': ['scale', 'auto']}],
    [RandomForestClassifier(), 'Random Forest', {'n_estimators': [100, 200, 300],
'max_depth': [None, 10, 20]}],
    [DecisionTreeClassifier(), 'Decision Tree', {'max_depth': [None, 10, 20]}],
    [MLPClassifier(), 'Neural network', {'hidden_layer_sizes': [(100,), (50, 50), (50, 100)]}],
    [Perceptron(), 'Perceptron', {'alpha': [0.0001, 0.001, 0.01]}],
```

```
    [SGDClassifier(), 'Stochastic Gradient Descent', {'alpha': [0.0001, 0.001, 0.01]}],
    [PassiveAggressiveClassifier(), 'Passive Aggressive', {'C': [0.1, 1, 10]}]
]

model_tf_score = []

for a in compare_model_tf:
    compare_tf = a[0]
    params = a[2]

    # สร้าง GridSearchCV object
    grid_search = GridSearchCV(compare_tf, params, cv=5, scoring='accuracy')

    # Fit โมเดลเพื่อค้นหา hyperparameters ที่ดีที่สุด
    grid_search.fit(tf_train_bow, y_train)

    # ใช้โมเดลที่ปรับจูน hyperparameters แล้วดีที่สุด
    best_model = grid_search.best_estimator_

    best_model.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['title_tokens_deepcut'])
    tf_test_predictions = best_model.predict(tf_test_bow)
    best_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(best_confusion_matrix)
    FP = best_confusion_matrix.sum(axis=0) - TP
    FN = best_confusion_matrix.sum(axis=1) - TP
    TN = best_confusion_matrix.sum() - (FP + FN + TP)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / best_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
```

```
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual, grid_search.best_params_])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print("Confusion Matrix for the Model")
    print(best_confusion_matrix)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print('-' * 55)

# แสดงผลการเปรียบเทียบ
score = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1', 'Best
hyperparameters'])
print(score)

score
```

```
score3 = pd.concat([score2, score])
score3
```

3.หัวข้อข่าวหุ้น QH เท่านั้น ตัวตัดคำ PythaiNLP

Install Library

```
!pip install pythainlp
!pip install wordcloud
!pip install python-crfsuite
!pip install matplotlib-venn

!pip install stop_words

from google.colab import drive
drive.mount('/content/drive')

import pandas as pd
import pythainlp


from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
GradientBoostingClassifier,RandomForestClassifier,AdaBoostClassifier,BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier



import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.feature_extraction.text import CountVectorizer
```

```
import re
import string

from pythainlp.corpus.common import thai_stopwords
import warnings
warnings.filterwarnings(action='ignore')

import numpy as np
from PIL import Image
from wordcloud import WordCloud

from pythainlp.tokenize import word_tokenize

from pythainlp.util import find_keyword
from pythainlp.util import rank
from wordcloud import WordCloud
import matplotlib
import matplotlib.font_manager as fm
fm.fontManager.addfont('/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf')
matplotlib.rc('font',family='TH Sarabun New',size =10)

# Import Library

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import string
import re
from pythainlp.util import find_keyword
from pythainlp.util import rank
import pythainlp
from pythainlp.tokenize import word_tokenize
```

```
from pythainlp.corpus import thai_stopwords
from pythainlp.corpus import wordnet

import nltk
nltk.download('stopwords')
from stop_words import get_stop_words
from nltk.corpus import stopwords

# นำเข้าข้อมูลจาก Github (เรียกใช้งานด้วยข้อมูลเก่าที่เคยทำ)
df = pd.read_csv("/content/drive/MyDrive/Data/data project/QH_only.csv")
df.head()

df.dropna(inplace=True)

# ดูว่ามีข้อมูลเกี่ยวไม่เกี่ยวว่ามีกี่ข้อมูล

df['sentiment'].value_counts()

# เปรียบเทียบจำนวนของข้อมูล

df['sentiment'].value_counts().plot.bar()

# ทำการตัดคำ

from pythainlp import word_tokenize

def text_process(text):
    final = "".join(u for u in text if u not in ("’","?", ";", ":", "!",'/',"–          ",'.', '"','”','“', "ๆ", "ฯ",
"=", "_", "(", ")","%", "&", "", ",", "+", "*", "[", "]", "#", "฿", "$", "@","
๙","1","2","3",'4','5','6','7','8','9','0','-'))
    final = word_tokenize(final)
    final = " ".join(word for word in final)
    return final
```

```
df['title_tokens'] = df['title'].apply(text_process)
df

from pythainlp.corpus.common import thai_stopwords

# ทำการกำจัดคำหยุด pythai NLP

from pythainlp.corpus.common import thai_stopwords

thai_stopwords = list(thai_stopwords())

def stopword(text):
    list_word = word_tokenize(text, keep_whitespace= False) #, engine='newmm')

    stopwords = thai_stopwords
    final = [i for i in list_word if i not in stopwords]
    return final

df['del_stopword'] = df['title_tokens'].apply(stopword)

detokenized_doc = []
for i in range(len(df)):
    t = ' '.join(map(str, df['del_stopword'][i]))
    detokenized_doc.append(t)

df['del_stopword'] = detokenized_doc
df

df.to_csv("stopword.csv")

df0 = df[df['sentiment']=='pos']

df1 =df[df['sentiment']=='neg']
```

```
df2 =df[df['sentiment']=='nue']

def get_text_list(message):
  tokenized =[]
  for i in message:
    token = word_tokenize(i)
    for j in token:
      tokenized.append(j)

  return tokenized

text_list = get_text_list(df1.del_stopword)
word_count = pd.DataFrame([find_keyword(text_list, min_len=3)]).T

word_count['word'] = word_count.index
word_count.columns = ['count','word']
word_count.index = range(len(word_count))

word_count.sort_values(by='count',ascending =False,inplace=True)
word_count

word_count.iloc[:35,:]

from tqdm import tqdm

def get_text_str(message):
  tokenized =" "
  th_stw1 = thai_stopwords
  for i in tqdm(message):
    token = word_tokenize(i)
    for j in token:
      if j not in thai_stopwords:
```

```
        tokenized =tokenized + " " + j

  return tokenized


text_str = get_text_str(df0.del_stopword)

path = 'THSarabunNew.ttf'
regexp = r"[ก-๙a-zA-Z']+"
wordcloud = WordCloud(
                    font_path='/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf',
                    min_font_size=1,
                    background_color="white",
                    width=400,
                    height=200,
                    max_words=1000,
                    colormap='plasma',
                    scale=3,
                    font_step=4,
                   #   contour_width=3,
                    contour_color='steelblue',
                    collocations=False,
                    regexp=regexp,
                    margin=2
                    ).generate(text_str)

fig, ax = plt.subplots(1, 1, figsize=(16, 12))
ax.imshow(wordcloud, interpolation='bilinear')
ax.axis("off")
fig.show()

# แบ่งข้อมูลเป็น train 70 test 30 แบบสุ่ม pythaiNLP
```

```
from sklearn.model_selection import train_test_split

X = df[['del_stopword']]  #  โจทย์
y = df['sentiment']  #  เฉลย

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1234,shuffle =True)

# นับจำนวนคำเพื่อทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(analyzer=lambda x:x.split(' '))
tfidf.fit_transform(df['del_stopword'])
tfidf.vocabulary_

# ทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer PythaiNLP

tf_train_bow = tfidf.transform(X_train['del_stopword'])
pd.DataFrame(tf_train_bow.toarray(), columns=tfidf.get_feature_names_out(),
index=X_train['del_stopword'])
pdtemp = pd.DataFrame(tf_train_bow.toarray(),
columns=tfidf.get_feature_names_out(), index=X_train['del_stopword'])

from sklearn.linear_model import LogisticRegression, SGDClassifier,
PassiveAggressiveClassifier, Perceptron
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix
import numpy as np
import pandas as pd

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression'],
    [MultinomialNB(), 'Naive Bayes'],
    [KNeighborsClassifier(), 'KNN'],
    [SVC(), 'SVM'],
    [RandomForestClassifier(), 'Random Forest'],
    [DecisionTreeClassifier(), 'Decision Tree'],
    [MLPClassifier(), 'Neural network'],
    [Perceptron(), 'Perceptron'],
    [SGDClassifier(), 'Stochastic Gradient Descent'],
    [PassiveAggressiveClassifier(), 'Passive Aggressive'],
]

model_tf_score = []
for a in compare_model_tf:
    compare_tf = a[0]
    compare_tf.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['del_stopword'])
    tf_test_predictions = compare_tf.predict(tf_test_bow)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    # คำนวณ confusion matrix
    tf_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)
```

```
    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(tf_confusion_matrix)
    FP = tf_confusion_matrix.sum(axis=0) - TP
    FN = tf_confusion_matrix.sum(axis=1) - TP
    TN = tf_confusion_matrix.sum() - (TP + FP + FN)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / tf_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print("Confusion Matrix for the Best Model (Highest Accuracy):")
    print(tf_confusion_matrix)
    print('-' * 55)

# แสดงผลสุดท้ายของคะแนนโมเดล
score2 = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1'])
```

```
print(score2)

score2

from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import confusion_matrix

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression', {'C': [0.1, 1, 10, 100]}],
    [MultinomialNB(), 'Naive Bayes', {'alpha': [0.1, 0.5, 1.0]}],
    [KNeighborsClassifier(), 'KNN', {'n_neighbors': [3, 5, 7]}],
    [SVC(), 'SVM', {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf'], 'gamma': ['scale', 'auto']}],
    [RandomForestClassifier(), 'Random Forest', {'n_estimators': [100, 200, 300],
'max_depth': [None, 10, 20]}],
    [DecisionTreeClassifier(), 'Decision Tree', {'max_depth': [None, 10, 20]}],
    [MLPClassifier(), 'Neural network', {'hidden_layer_sizes': [(100,), (50, 50), (50, 100)]}],
    [Perceptron(), 'Perceptron', {'alpha': [0.0001, 0.001, 0.01]}],
    [SGDClassifier(), 'Stochastic Gradient Descent', {'alpha': [0.0001, 0.001, 0.01]}],
    [PassiveAggressiveClassifier(), 'Passive Aggressive', {'C': [0.1, 1, 10]}]
]

model_tf_score = []

for a in compare_model_tf:
    compare_tf = a[0]
    params = a[2]

    # สร้าง GridSearchCV object
    grid_search = GridSearchCV(compare_tf, params, cv=5, scoring='accuracy')

    # Fit โมเดลเพื่อค้นหา hyperparameters ที่ดีที่สุด
    grid_search.fit(tf_train_bow, y_train)
```

```
    # ใช้โมเดลที่ปรับจูน hyperparameters แล้วดีที่สุด
    best_model = grid_search.best_estimator_

    best_model.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['del_stopword'])
    tf_test_predictions = best_model.predict(tf_test_bow)
    best_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(best_confusion_matrix)
    FP = best_confusion_matrix.sum(axis=0) - TP
    FN = best_confusion_matrix.sum(axis=1) - TP
    TN = best_confusion_matrix.sum() - (FP + FN + TP)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / best_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual, grid_search.best_params_])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
```

```
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print("Confusion Matrix for the Model")
    print(best_confusion_matrix)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print('-' * 55)

# แสดงผลการเปรียบเทียบ
score = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1', 'Best
hyperparameters'])
print(score)

score

score3 = pd.concat([score2, score])
score3
```

4. หัวข้อข่าวหุ้น QH เท่านั้น ตัวตัดคำ DeepCut

```
# Install Library

!pip install pythainlp
!pip install wordcloud
!pip install python-crfsuite
!pip install matplotlib-venn
!pip install stop_words

from google.colab import drive
drive.mount('/content/drive')

# Import Library

import pandas as pd
import pythainlp


from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
GradientBoostingClassifier,RandomForestClassifier,AdaBoostClassifier,BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier



import matplotlib.pyplot as plt
import seaborn as sns
```

```
from sklearn.feature_extraction.text import CountVectorizer

import re
import string

from pythainlp.corpus.common import thai_stopwords
import warnings
warnings.filterwarnings(action='ignore')

import numpy as np
from PIL import Image

from pythainlp.tokenize import word_tokenize

from pythainlp.util import find_keyword
from pythainlp.util import rank
from wordcloud import WordCloud
import matplotlib
import matplotlib.font_manager as fm
fm.fontManager.addfont('/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf')
matplotlib.rc('font',family='TH Sarabun New',size =10)

# Import Library

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import string
import re
from pythainlp.util import find_keyword
from pythainlp.util import rank
import pythainlp
```

```
from pythainlp.tokenize import word_tokenize
from pythainlp.corpus import thai_stopwords
from pythainlp.corpus import wordnet

import nltk
nltk.download('stopwords')
from stop_words import get_stop_words
from nltk.corpus import stopwords

# นำเข้าข้อมูลจาก Github (เรียกใช้งานด้วยข้อมูลเก่าที่เคยทำ)
df = pd.read_csv("/content/drive/MyDrive/Data/data project/QH_only.csv")
df.head()

# ดูว่ามีข้อมูลเกี่ยวไม่เกี่ยวว่ามีกี่ข้อมูล

df['sentiment'].value_counts()

# เปรียบเทียบจำนวนของข้อมูล

df['sentiment'].value_counts().plot.bar()

pip install deepcut

import deepcut
def text_process(text):
    final = "".join(u for u in text if u not in ("’","?", ";", ":", "!",'/',"–          ",'.', '"','”','“', "ๆ", "ฯ",
"=", "_", "(", ")","%", "&", "", ",", "+", "*", "[", "]", "#", "฿", "$", "@","
๙","1","2","3",'4','5','6','7','8','9','0','-'))
    final = deepcut.tokenize(final)
    final = " ".join(word for word in final)
    return final
df['title_tokens_deepcut'] = df['title'].apply(text_process)
df
```

```
df.to_csv("stopword.csv")

# ทำการกำจัดคำหยุด
from pythainlp import word_tokenize
from pythainlp.corpus.common import thai_stopwords

thai_stopwords = list(thai_stopwords())

def stopword(text):
    list_word = word_tokenize(text, keep_whitespace= False) #, engine='newmm')

    stopwords = thai_stopwords
    final = [i for i in list_word if i not in stopwords]
    return final

df['del_stopword_deepcut'] = df['title_tokens_deepcut'].apply(stopword)

detokenized_doc = []
for i in range(len(df)):
    t = ' '.join(map(str, df['del_stopword_deepcut'][i]))
    detokenized_doc.append(t)

df['del_stopword_deepcut'] = detokenized_doc
df

df0 = df[df['sentiment']=='pos']

def get_text_list(message):
  tokenized =[]
  for i in message:
    token = word_tokenize(i)
    for j in token:
```

```
        tokenized.append(j)

    return tokenized

text_list = get_text_list(df0.del_stopword_deepcut)
word_count = pd.DataFrame([find_keyword(text_list, min_len=3)]).T

word_count['word'] = word_count.index
word_count.columns = ['count','word']
word_count.index = range(len(word_count))

word_count.sort_values(by='count',ascending =False,inplace=True)
word_count

word_count.iloc[:35,:]

from tqdm import tqdm

def get_text_str(message):
    tokenized =" "
    th_stw1 = thai_stopwords
    for i in tqdm(message):
        token = word_tokenize(i)
        for j in token:
            if j not in thai_stopwords:
                tokenized =tokenized + " " + j

    return tokenized


text_str = get_text_str(df0.del_stopword_deepcut)

path = 'THSarabunNew.ttf'
```

```
regexp = r"[ก-๙a-zA-Z']+"
wordcloud = WordCloud(
                font_path='/content/drive/MyDrive/Data/data
project/THsarabunNew.ttf',
                min_font_size=1,
                background_color="white",
                width=400,
                height=200,
                max_words=1000,
                colormap='plasma',
                scale=3,
                font_step=4,
               #   contour_width=3,
                contour_color='steelblue',
                collocations=False,
                regexp=regexp,
                margin=2
                ).generate(text_str)

fig, ax = plt.subplots(1, 1, figsize=(16, 12))
ax.imshow(wordcloud, interpolation='bilinear')
ax.axis("off")
fig.show()

# แบ่งข้อมูลเป็น train 70 test 30 แบบสุ่ม DEEPCUT

from sklearn.model_selection import train_test_split

X = df[['title_tokens_deepcut']]  #  โจทย์
y = df['sentiment']  #  เฉลย

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1234,shuffle =True)
```

```
# นับจำนวนคำเพื่อทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(analyzer=lambda x:x.split(' '))
tfidf.fit_transform(df['title_tokens_deepcut'])
tfidf.vocabulary_

# ทำให้ข้อมูลอยู่ในรูป Vector ด้วย TfidfVectorizer Deepcut

tf_train_bow = tfidf.transform(X_train['title_tokens_deepcut'])
pd.DataFrame(tf_train_bow.toarray(), columns=tfidf.get_feature_names_out(),
index=X_train['title_tokens_deepcut'])

from sklearn.linear_model import LogisticRegression, SGDClassifier,
PassiveAggressiveClassifier, Perceptron
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix
import numpy as np
import pandas as pd

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression'],
    [MultinomialNB(), 'Naive Bayes'],
    [KNeighborsClassifier(), 'KNN'],
    [SVC(), 'SVM'],
```

```
    [RandomForestClassifier(), 'Random Forest'],
    [DecisionTreeClassifier(), 'Decision Tree'],
    [MLPClassifier(), 'Neural network'],
    [Perceptron(), 'Perceptron'],
    [SGDClassifier(), 'Stochastic Gradient Descent'],
    [PassiveAggressiveClassifier(), 'Passive Aggressive'],
]

model_tf_score = []
for a in compare_model_tf:
    compare_tf = a[0]
    compare_tf.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['title_tokens_deepcut'])
    tf_test_predictions = compare_tf.predict(tf_test_bow)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    # คำนวณ confusion matrix
    tf_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(tf_confusion_matrix)
    FP = tf_confusion_matrix.sum(axis=0) - TP
    FN = tf_confusion_matrix.sum(axis=1) - TP
    TN = tf_confusion_matrix.sum() - (TP + FP + FN)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / tf_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
```

```
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print("Confusion Matrix for the Best Model (Highest Accuracy):")
    print(tf_confusion_matrix)
    print('-' * 55)

# แสดงผลสุดท้ายของคะแนนโมเดล
score2 = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1'])
print(score2)

score2

from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import confusion_matrix

compare_model_tf = [
    [LogisticRegression(), 'Logistic Regression', {'C': [0.1, 1, 10, 100]}],
    [MultinomialNB(), 'Naive Bayes', {'alpha': [0.1, 0.5, 1.0]}],
```

```
    [KNeighborsClassifier(), 'KNN', {'n_neighbors': [3, 5, 7]}],
    [SVC(), 'SVM', {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf'], 'gamma': ['scale', 'auto']}],
    [RandomForestClassifier(), 'Random Forest', {'n_estimators': [100, 200, 300],
'max_depth': [None, 10, 20]}],
    [DecisionTreeClassifier(), 'Decision Tree', {'max_depth': [None, 10, 20]}],
    [MLPClassifier(), 'Neural network', {'hidden_layer_sizes': [(100,), (50, 50), (50, 100)]}],
    [Perceptron(), 'Perceptron', {'alpha': [0.0001, 0.001, 0.01]}],
    [SGDClassifier(), 'Stochastic Gradient Descent', {'alpha': [0.0001, 0.001, 0.01]}],
    [PassiveAggressiveClassifier(), 'Passive Aggressive', {'C': [0.1, 1, 10]}]
]

model_tf_score = []

for a in compare_model_tf:
    compare_tf = a[0]
    params = a[2]

    # สร้าง GridSearchCV object
    grid_search = GridSearchCV(compare_tf, params, cv=5, scoring='accuracy')

    # Fit โมเดลเพื่อค้นหา hyperparameters ที่ดีที่สุด
    grid_search.fit(tf_train_bow, y_train)

    # ใช้โมเดลที่ปรับจูน hyperparameters แล้วดีที่สุด
    best_model = grid_search.best_estimator_

    best_model.fit(tf_train_bow, y_train)
    tf_test_bow = tfidf.transform(X_test['title_tokens_deepcut'])
    tf_test_predictions = best_model.predict(tf_test_bow)
    best_confusion_matrix = confusion_matrix(y_test, tf_test_predictions)

    # คำนวณค่า TP, FP, FN, TN
    TP = np.diag(best_confusion_matrix)
```

```
    FP = best_confusion_matrix.sum(axis=0) - TP
    FN = best_confusion_matrix.sum(axis=1) - TP
    TN = best_confusion_matrix.sum() - (FP + FN + TP)

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วยตนเอง
    acc_manual = (TP.sum() / best_confusion_matrix.sum()) * 100
    pre_manual = (TP / (TP + FP)).mean() * 100
    rec_manual = (TP / (TP + FN)).mean() * 100
    f1_manual = (2 * pre_manual * rec_manual / (pre_manual + rec_manual))

    # คำนวณค่า accuracy, precision, recall, f1-score ด้วย scikit-learn
    acc = accuracy_score(y_test, tf_test_predictions) * 100
    pre = precision_score(y_test, tf_test_predictions, average='weighted') * 100
    rec = recall_score(y_test, tf_test_predictions, average='weighted') * 100
    f1 = f1_score(y_test, tf_test_predictions, average='weighted') * 100

    model_tf_score.append([a[1], acc, pre, rec, f1, acc_manual, pre_manual,
rec_manual, f1_manual, grid_search.best_params_])

    # แสดงผล
    print(f'{a[1]}')
    print('Accuracy = ', acc)
    print('Precision = ', pre)
    print('Recall = ', rec)
    print('F1-Score = ', f1)
    print("Confusion Matrix for the Model")
    print(best_confusion_matrix)
    print('Manual Accuracy = ', acc_manual)
    print('Manual Precision = ', pre_manual)
    print('Manual Recall = ', rec_manual)
    print('Manual F1-Score = ', f1_manual)
    print('-' * 55)
```

```
# แสดงผลการเปรียบเทียบ
score = pd.DataFrame(model_tf_score, columns=['classifier', 'accuracy', 'precision',
'recall', 'f1', 'manual_accuracy', 'manual_precision', 'manual_recall', 'manual_f1', 'Best
hyperparameters'])
print(score)

score

score3 = pd.concat([score2, score])
score3
```

ประวัติผู้เขียน

ชื่อ	นางสาวณัฐิมา อภินาทเมธี
วัน เดือน ปีเกิด	15 มกราคม 2533
ที่อยู่ปัจจุบัน	5/434 รามคำแหง60/2 แขวงหัวหมาก เขตบางกะปิ 10240
ประวัติการศึกษา	(2555) วิทยาศาสตร์บัณฑิต สาขาเทคโนโลยีสารสนเทศ เกรดเฉลี่ย 2.72
ทุนการศึกษาที่ได้รับ	ไม่มี
ผลงานทางวิชาการ	ไม่มี