การวิเคราะห์ความสนใจของนักท่องเที่ยวด้วยเทคนิคการเรียนรู้ของเครื่อง : กรณีศึกษา ถนนเยาวราช ประเทศไทย

ANALYSIS OF TOURIST ATTRACTIONS USING MACHINE LEARNING TECHNIQUES: A CASE STUDY OF YAOWARAT ROAD, THAILAND

นิธิกร เลิศชาญวุฒิ

NITHIKORN LERTCHANVUTH

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาสถิติและการวิเคราะห์ธุรกิจ

ภาควิชาสถิติ คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2564

KMITL- 2021-SC-M-050-004

ANALYSIS OF TOURIST ATTRACTIONS USING MACHINE LEARNING TECHNIQUES: A CASE STUDY OF YAOWARAT ROAD, THAILAND

NITHIKORN LERTCHANVUTH

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE PROGRAM IN STATISTICS AND BUSINESS ANALYTICS

SCHOOL OF SCIENCE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2021

KMITL-2021-SC-M-050-004

หัวข้อวิทยานิพนธ์	การวิเคราะห์ความสนใจของนักท่องเที่ยวด้วยเทคนิคการเรียนรู้ของเครื่อง : กรณีศึกษา ถนนเยาวราช ประเทศไทย
ชื่อนักศึกษา	นายนิธิกร เลิศชาญวุฒิ
รหัสประจำตัว	61605116
ปริญญา	วิทยาศาสตรมหาบัณฑิต (สถิติและการวิเคราะห์ธุรกิจ)
ภาควิชา	สถิติ
พ.ศ.	2564
อาจารย์ที่ปรึกษาวิทยานิพนธ์	ผู้ช่วยศาสตราจารย์ ดร.กนกกรรณ์ ลิ้โรจนาประภา

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อวิเคราะห์กลุ่มความสนใจ ความรู้สึก และหาข้อมูลเชิงลึกของนักท่องเที่ยวด้วยวิธีการทำเหมืองข้อความ โดยข้อมูลที่ใช้คือบทวิจารณ์ออนไลน์จากนักท่องเที่ยวที่กล่าวถึงเยาวราช ประเทศไทย โดยข้อมูลที่ใช้จะเป็นบทวิจารณ์ออนไลน์จากเว็ป Tripadvisor ทั้งหมด 3,992 บทวิจารณ์ โดยผู้วิจัยได้ใช้แบ่งการวิเคราะห์ออกเป็น 4 ส่วนใหญ่ ส่วนที่หนึ่งคือ การวิเคราะห์กลุ่มความสนใจของนักท่องเที่ยว ด้วยการจัดกลุ่มแบบเคมีน ร่วมกับการจัดสรรหัวข้อแฝง ซึ่งพบว่านักท่องเที่ยวที่เดินทางมายังเยาวราชสามารถแบ่งความสนใจออกได้เป็น 4 กลุ่มได้แก่ แหล่งช้อปปิ้ง ตลาดอาหารริมทางยามค่ำ อาหาร และการเที่ยวชมเมือง ส่วนที่สองคือ การวิเคราะห์ความรู้สึกของนักท่องเที่ยวด้วยการจำแนกความรู้สึกเป็นเชิงบวกและเชิงลบ ซึ่งจะใช้โมเดลทั้งหมด 3 โมเดลได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีน โดยจะนำโมเดลทั้ง 3 มาร่วมกันตัดสินใจซึ่งให้มีค่าความถูกต้อง 88.62% ส่วนที่สามคือการหาข้อมูลเชิงลึกโดยใช้เทคนิคต่างๆ ได้แก่ การวิเคราะห์ความนิยม การวิเคราะห์ความเด่นและความแพร่หลาย การวิเคราะห์แนวโน้ม และส่วนสุดท้ายจะเป็นการเสนอแนวทางการประยุกต์ใช้โมเดลด้วยการสร้างเว็ปแอปพลิเคชั่นเพื่อใช้ในการวิเคราะห์บทวิจารณ์แบบอัตโนมัติ

คำสำคัญ : เหมืองข้อความ ความสนใจของนักท่องเที่ยว แบบจำลองหัวข้อ การวิเคราะห์ความรู้สึก

Thesis Title	Analysis of Tourist Attractions using Machine Learning Techniques: A Case Study of Yaowarat Road, Thailand
Student Name	Nithikorn Lertchanvuth
Student ID	61605116
Degree	Master of Science (Statistics and Business Analytics)
Department	Statistics
Year	2021
Thesis Advisor	Assist. Prof. Dr. Kanogkan Leerojanaprapa

Abstract

The objectives of this research are to analyze tourist attraction, sentiment and insight of tourist using text mining. Online reviews from the tourist who reviewed about Yaowarat road through Tripadvisor which are dataset, 3,992 reviews in total. This research is separated into 4 parts. The first part is the analysis of tourism attraction segment using K-means clustering and Latent dirichlet allocation. The result shows that tourism attraction is clustered into 4 interest segments which are shopping place, night street food market, food, and sightseeing. The second part is binary sentiment analysis (positive and negative) using 3 models included Naïve bayes, Logistic regression, and Support vector machine. Then majority vote ensembling be developed with 88.62% of accuracy. The third part is insight analysis using several techniques included popularity analysis, salience-valence analysis, and trend analysis. The final part is a way to apply model with development of automated review analyzer as a web application.

Keywords: text mining, tourism attraction, latent dirichlet allocation, sentiment analysis

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สำเร็จลุล่วงไปได้ด้วยดีด้วยความอนุเคราะห์จาก ผศ.ดร.กนกกรรณ์ ลิ้โรจนาประภา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ผศ.ดร.สรวิชญ์ เยาวสุวรรณไชย และผศ.ดร.พรพิมล ชัยวุฒิศักดิ์ กรรมการสอบวิทยานิพนธ์ที่ได้กรุณาให้ความช่วยเหลือแนะนำช่วยตรวจทานแก้ไข้อผิดพลาดต่างๆ อีกทั้งขอขอบคุณ ดร.โกเมษ จันทวิมล ที่ช่วยให้คำแนะนำต่างๆจนวิทยานิพนธ์เล่มนี้สำเร็จลุล่วงสมบูรณ์ สุดท้ายนี้คุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์เล่มนี้ ข้าพเจ้าขอมอบให้แก่ผู้มีพระคุณทุกท่าน

นายนิธิกร เลิศชาญวุฒิ

สารบัญ

บทคัดย่อภาษาไทย ก

บทคัดย่อภาษาอังกฤษ ข

กิตติกรรมประกาศ ค

สารบัญ ง

สารบัญตาราง ช

สารบัญรูปภาพ ช

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของงานวิจัย 4

1.3 ขอบเขตของงานวิจัย 4

1.4 ประโยชน์ที่คาดว่าจะได้รับ 4

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 5

2.1 การเตรียมข้อมูลบทวิจารณ์ 5

2.1.1 การตัดคำ (Tokenization) 5

2.1.2 การลบเครื่องหมายวรรคตอน (Removing Punctuation) 5

2.1.3 การนอร์มอลไลซ์ตัวอักษร (Character Normalization) 5

2.1.4 การลบตัวเลขและกลุ่มคำที่ไม่มีความหมาย (Stop Word) 5

2.1.5 การตัดส่วนท้ายของคำ (Word Stemming) 6

2.1.6 ถุงคำศัพท์ (Bag of Word: BOW) 6

2.1.7 การบ่งบอกประเภทของความรู้สึกในบทวิจารณ์ (Class Labeling) 6

2.2 การเรียนรู้แบบไม่มีผู้สอนเพื่อจัดกลุ่มความสนใจของนักท่องเที่ยว 6

2.2.1 การแบ่งกลุ่มแบบเคมีน (K-means Clustering) 6

2.2.2 การกำหนดจำนวนกลุ่มที่เหมาะสม (Optimal Cluster Number Selection) 9

2.2.3 การแบ่งกลุ่มข้อความ (Text Clustering) 9

2.2.3.1 การแจกแจงดีรีเคล (Dirichlet Distribution) 9

2.2.3.2 การจัดสรรหัวข้อแฝง (Latent Dirichlet Allocation: LDA) 10

2.3 การเรียนรู้แบบมีผู้สอนเพื่อทำนายความรู้สึกของนักท่องเที่ยว 14

2.3.1 นาอีฟเบย์ (Naïve Bayes) 14

สารบัญ (ต่อ)

2.3.2 การถดถอยเชิงโลจิสติค (Logistic Regression) 15

2.3.3 ซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น (Linear Support Vector Machine) 16

2.3.4 ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้น (Non-Linear Support Vector Machine) 18

2.3.5 การเรกูลาไรซ์ (Regularization) 21

2.3.6 การเคลื่อนลงตามความชัน (Gradient Descent) 22

2.3.7 การแก้ไขปัญหาประเภทคำตอบไม่เท่า (Class Balancing) 23

2.4 การเปรียบเทียบประสิทธิภาพการทำนาย 23

2.4.1 เมทริกซ์ความสับสน (Confusion Matrix) 23

2.4.2 กันการเลือกโมเดลที่ดีที่สุด (Model Selection) 24

2.5 การวิเคราะห์ความเด่นและความแพร่หลาย (Salience Valence Analysis) 25

2.5.1 การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม (Dimensional Salience Valence Analysis) 25

2.5.2 การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์ (Lexical Salience Valence Analysis 26

2.6 เว็ปแอพพลิเคชั่น 27

2.7 งานวิจัยที่เกี่ยวข้อง 28

บทที่ 3 วิธีการดำเนินงานวิจัย 33

3.1 การเก็บข้อมูลจากเว็ปไซต์ 33

3.2 การเตรียมข้อมูล 34

3.3 การแบ่งกลุ่มความสนใจของนักท่องเที่ยว 36

3.4 การวิเคราะห์ความรู้สึกของนักท่องเที่ยว 37

3.4.1 การแก้ไขปัญหาประเภทคำตอบไม่เท่า 37

3.4.2 นาอีฟเบย์ 38

3.4.3 การถดถอยเชิงโลจิสติค 38

3.4.4 ซัพพอร์ตเวกเตอร์แมชชีน 38

3.4.5 การทำนายกลุ่มจากการตัดสินใจร่วมกัน 38

3.5 การวิเคราะห์ความนิยม 39

3.6 การวิเคราะห์ความเด่นและความแพร่หลาย 39

สารบัญ (ต่อ)

3.6.1 การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม 39

3.6.2 การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์ 40

3.7 การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก 43

3.8 เว็ปแอพพลิเคชั่นสำหรับการวิเคราะห์กลุ่มความสนใจและความรู้สึก 45

3.9 การวิเคราะห์แนวโน้มของคำที่นักท่องเที่ยวใช้ในบทวิจารณ์ 45

บทที่ 4 ผลการวิจัยและการอภิปรายผล 46

4.1 ลักษณะของข้อมูลที่ใช้ในการทดลอง 46

4.2 ผลลัพธ์การแบ่งกลุ่มความสนใจของนักท่องเที่ยว 47

4.2.1 ผลลัพธ์การวิเคราะห์ข้อมูลแบบเคมีน 47

4.2.2 ผลลัพธ์โมเดลการจัดสรรหัวข้อแฝง 48

4.3 ผลลัพธ์การวิเคราะห์ความรู้สึกของนักท่องเที่ยว 51

4.3.1 ผลลัพธ์นาอีฟเบย์ 51

4.3.2 ผลลัพธ์การถดถอยเชิงโลจิสติค 53

4.3.3 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น 55

4.3.4 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้น 56

4.3.5 การเพิ่มประสิทธิภาพโมเดล 60

4.3.5.1 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติค 61

4.3.5.2 การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีน 64

4.3.6 ผลลัพธ์การเลือกโมเดลมาช่วยในการตัดสินใจ 66

4.4 ผลลัพธ์การวิเคราะห์ความนิยม 66

4.5 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลาย 67

4.5.1 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม 67

4.5.2 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์ 68

4.6 ผลลัพธ์การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก 73

4.7 ผลลัพธ์กการวิเคราะห์แนวโน้มของคำที่นักท่องเที่ยวใช้ในบทวิจารณ์ 74

4.8 การนำโมเดลไปประยุกต์ใช้งาน 76

บทที่ 5 สรุปผลการวิจัยและข้อเสนอแนะ 79

5.1 สรุปผลการวิจัย 79

สารบัญ (ต่อ)

5.2 ข้อเสนอแนะ 82
5.3 ข้อจำกัดของงานวิจัย 82
เอกสารอ้างอิง 83
ภาคผนวก ก 87
ภาคผนวก ข 104
ภาคผนวก ค 107
ประวัติผู้เขียน 168

สารบัญตาราง

ตารางที่

2.1 ตัวอย่างประโยคและการแปลงถุงคำศัพท์ 8
2.2 เมทริกซ์ความสับสน 23
2.3 ตัวอย่างการหาผลบวกจริง (TP) ผลลบจริง (TN) ผลบวกลวง (FP) และผลลบลวง (FN) 24
2.4 ความหมายของความเด่นและความแพร่หลาย 26
2.5 งานวิจัยที่เกี่ยวข้อง 30
3.1 ตัวอย่างขั้นตอนการเตรียมข้อมูล 36
3.2 ตัวอย่างข้อมูลสำหรับการคำนวณความเด่นและความแพร่หลายเชิงกลุ่ม 39
3.3 บทวิจารณ์ตัวอย่างสำหรับการคำนวณความเด่นและความแพร่หลายเชิงคำศัพท์ 41
3.4 ตัวอย่างค่าความเด่นและความแพร่หลายเชิงคำศัพท์ 42
3.5 ผลลัพธ์การวิเคราะห์ความรู้สึกของประโยค 44
4.1 จำนวนบทวิจารณ์ที่มีต่อถนนเยาวราชจำแนกตามปีและคะแนน 46
4.2 คำศัพท์ที่อยู่ในกลุ่มความสนใจของนักท่องเที่ยวจากโมเดลการจัดสรรหัวข้อแฝง 49
4.3 คำศัพท์ที่ถูกเลือกกลุ่มความสนใจของนักท่องเที่ยวจากโมเดลการจัดสรรหัวข้อแฝง 50
4.4 ตารางการตั้งชื่อกลุ่มความสนใจ 50
4.5 คำศัพท์ที่มีความน่าจะเป็นแบบมีเงื่อนไขสูงสุด 5 อันดับแรกของความรู้สึกเชิงลบ 52
4.6 คำศัพท์ที่มีความน่าจะเป็นแบบมีเงื่อนไขสูงสุด 5 อันดับแรกของความรู้สึกเชิงบวก 52
4.7 ผลการทดสอบของนาอีฟเบย์ด้วยชุดข้อมูลฝึกสอน 52
4.8 ผลการทดสอบของนาอีฟเบย์ด้วยชุดข้อมูลทดสอบ 53
4.9 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของการถดถอยเชิงโลจิสติค 53
4.10 ผลการทดสอบการถดถอยเชิงโลจิสติคด้วยชุดข้อมูลฝึกสอน 54
4.11 ผลการทดสอบการถดถอยเชิงโลจิสติคด้วยชุดข้อมูลทดสอบ 54
4.12 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น 55
4.13 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลฝึกสอน 55
4.14 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลทดสอบ 56
4.15 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นด้วยข้อมูลฝึกสอน 57
4.16 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นด้วยข้อมูลฝึกสอน
4.17 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นด้วยข้อมูลทดสอบ 58

สารบัญตาราง (ต่อ)

ตารางที่

4.18 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลด้วยข้อมูลฝึกสอน 58
4.19 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลด้วยข้อมูลทดสอบ 59
4.20 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ด้วยข้อมูลฝึกสอน 59
4.21 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ด้วยข้อมูลทดสอบ 60
4.22 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลฝึกสอน 62
4.23 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลทดสอบ 63
4.24 การเพิ่มประสิทธิภาพซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF โดยใช้ข้อมูลฝึกสอน 64
4.25 การเพิ่มประสิทธิภาพซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF โดยใช้ข้อมูลทดสอบ 65
4.26 ค่าความถูกต้องของโมเดลทั้ง 3 เมื่อเทียบกับโมเดลร่วมกันตัดสินใจแบบเสียงข้างมากโดยใช้ข้อมูลชุดฝึกสอน 66
4.27 ค่าความถูกต้องของโมเดลทั้ง 3 เมื่อเทียบกับโมเดลร่วมกันตัดสินใจแบบเสียงข้างมากโดยใช้ข้อมูลทดสอบ 66
4.28 คะแนนเฉลี่ยของแต่ละกลุ่มความสนใจ 67
4.29 ค่าความเด่นและค่าความแพร่หลายจำนวนกลุ่มความสนใจ 68
4.30 ค่าความเด่นและความแพร่หลายของกลุ่มความสนใจทั้ง 4 กลุ่ม 69
4.31 ค่าความเด่นและความแพร่หลาย 70
4.32 ผลลัพธ์การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก 73
4.33 จำนวนบทวิจารณ์มีคำศัพท์ในแต่ละปี 75

สารบัญรูป

รูปที่

2.1 ผลลัพธ์การทำเคมีนของประโยคที่แปลงด้วยถุงคำศัพท์ 8
2.2 การเปลี่ยนแปลงรูปร่างของการแจกแจงดีรีเคล 10
2.3 ขั้นตอนการทำงานทั้งหมดของโมเดลการจัดสรรหัวข้อแฝง 11
2.4 การกระจายแบบดีรีเคลของกลุ่มความสนใจ 12
2.5 กลุ่มความสนใจที่ถูกสุ่มขึ้นมา 12
2.6 การกระจายแบบดีรีเคลของคำศัพท์ 13
2.7 คำศัพท์ที่สุ่มขึ้นมาได้ในแต่ละกลุ่มความสนใจ 14
2.8 การกระจายของข้อมูลที่แบ่งแบบเชิงเส้นได้ 18
2.9 การกระจายของข้อมูลที่ไม่สามารถแบ่งแบบเชิงเส้นได้ 18
2.10 การแปลงข้อมูลให้อยู่ในปริภูมิที่สูงขึ้น 19
2.11 การปรับเปลี่ยนค่าน้ำหนักด้วยวิธีการเคลื่อนลงตามความชัน 22
3.1 โครงสร้างของงานวิจัย 33
3.2 ตัวอย่างบทวิจารณ์วิจารณ์ 34
3.3 ไฟล์เอกเซลที่เก็บข้อมูล 34
3.4 ขั้นตอนการเตรียมข้อมูล 35
3.5 ขั้นตอนการแบ่งกลุ่มนักท่องเที่ยว 37
3.7 ตัวอย่างแผนภูมิฟองสบู่ 43
4.1 ผลลัพธ์การแบ่งกลุ่มแบบเคมีน 48
4.2 แผนภาพฟองสบู่ของการวิเคราะห์ตวามเด่นและความแพร่หลาย 70
4.3 กราฟแนวโน้มของคำศัพท์ที่มีความชันเป็นบวก 76
4.4 กราฟแนวโน้มของคำศัพท์ที่มีความชันเป็นลบ 76
4.5 ลักษณะของหน้าเว็ปแอพพลิเคชัน 77
4.6 ผลลัพธ์ที่แสดงหลังจากทำการวิเคราะห์บทวิจารณ์ 77

บทที่ 1
บทนำ

งานวิจัยนี้มีที่มาที่แสดงถึง ความสำคัญของการกำหนดปัญหาที่ใช้ในการศึกษา และการกำหนดวัตถุประสงค์ในการวิจัยดังนี้

1.1 ที่มาและความสำคัญของปัญหา

การท่องเที่ยวคือปรากฏการณ์ทางสังคมและทางเศรษฐศาสตร์ที่ส่งผลกระทบต่อสังคมร่วมสมัย โดยปัจจุบันอุตสาหกรรมการท่องเที่ยวถือว่าเป็นพฤติกรรมทางธุรกิจประเภทหนึ่งเนื่องจากการท่องเที่ยวส่งผลกระทบต่อเขตเศรษฐกิจในพื้นที่ ซึ่งก่อให้เกิดการแข่งขันในหลายพื้นที่ ทั้งการแข่งขันระดับจังหวัดในประเทศเดียวกันเองจนถึงการแข่งขันในระดับข้ามประเทศโดยมีเป้าหมายเพื่อที่จะดึงดูดนักท่องเที่ยวให้นำเงินเข้ามาใช้จ่ายในประเทศให้ได้มากที่สุด (Chang Liu and Ning 2014)

จากการศึกษาของ องค์การการท่องเที่ยวโลก (World Tourism Organization 2011) พบว่าจำนวนนักท่องเที่ยวจากทั่วโลกจะมีอัตราการเพิ่มจำนวนร้อยละ 3.3 ต่อปีโดยเฉลี่ย หรือประมาณ 1,800 ล้านคน ในปี ค.ศ. 2030 โดยภูมิภาคที่มีการเพิ่มจำนวนนักท่องเที่ยวมากที่สุดคือภูมิภาคเอเชียและแปซิฟิกโดยจากรายงานของการท่องเที่ยวแห่งประเทศไทย (ททท.) (Chairerk and Wongmontha 2019) พบว่าการท่องเที่ยวสำหรับประเทศไทยเป็นแหล่งรายได้หลักที่สำคัญของประเทศไทย ซึ่งอุตสาหกรรมการท่องเที่ยวได้สร้างรายได้ให้กับประเทศไทยร้อยละ 9 ของมูลค่าของอุตสาหกรรมผลิตภัณฑ์รวมภายในประเทศทั้งหมด โดยรายรับจากการท่องเที่ยวของประเทศไทยจัดอยู่ในอันดับ 1 ในภูมิภาคอาเซียน และจากค่าดัชนีของสถานที่ที่นักท่องเที่ยวเดินไปมากที่สุด ซึ่งจัดทำโดยมาสเตอร์การ์ดพบว่าในปี พ.ศ. 2562 กรุงเทพมหานครเป็นสถานที่ที่นักท่องเที่ยวจากทั่วโลกได้เดินทางมาเยือนมากที่สุดในโลก ซึ่งส่งผลให้ช่วง 2 ไตรมาสแรกของปี พ.ศ.2562 รายได้จากการท่องเที่ยวมีมูลค่า 0.95 ล้านล้านบาท ซึ่งเติบโตขึ้นจากปี 2561 ถึง 2.32% (ข่าวเศรษฐกิจ 2561)

สถานที่ท่องเที่ยวในประเทศไทยมีรูปแบบที่หลากหลายและมีเอกลักษณ์ที่แตกต่างกันออกไปเพื่อตอบสนองความสนใจของนักท่องเที่ยวแต่ละกลุ่ม หนึ่งในสถานที่ท่องเที่ยวที่ได้รับความนิยมจากนักท่องเที่ยวที่มาเยือนกรุงเทพมหานครคือเยาวราช หรือไชน่าทาวน์ของประเทศไทย ตั้งอยู่ที่เยาวราช เขตสัมพันธวงศ์ กรุงเทพมหานคร เป็นย่านที่อยู่อาศัยของคนไทยเชื้อสายจีนที่มีความยาวของถนนประมาณ 1.4 กิโลเมตร ซึ่งจากข้อมูลของเว็บไซต์การประเมินการท่องเที่ยวที่น่าเชื่อถือ Tripadvisor พบว่า เยาวราช เป็นสถานที่ยอดนิยมสำหรับชาวต่างชาติอันดับที่ 17 จาก 697 สถานที่ท่องเที่ยว

กรุงเทพมหานคร โดยคิดค่าความนิยมเรียงตามรายการโปรดของนักท่องเที่ยว ทำให้เยาวราชเป็นหนึ่งในสถานทีที่นักท่องเที่ยวให้ความสำคัญในการเดินทางมาท่องเที่ยวที่กรุงเทพมหานคร อีกทั้งจากการจัดอันดับแหล่งอาหารริมทางที่ดีที่สุดในโลกจากสำนักข่าว CNN พบว่าเยาวราชเป็นแหล่งอาหารริมทางที่ดีที่สุดในกรุงเทพมหานคร ซึ่งกรุงเทพมหานครได้ถูกจัดอันดับเป็น 1 ใน 23 อันดับของเมืองที่มีอาหารริมทางดีที่สุดในโลกทำให้เยาวราชเป็นหนึ่งในสถานที่ที่นักท่องเที่ยวให้ความสำคัญในการเดินทางมาท่องเที่ยวที่กรุงเทพมหานคร (พิชชานันท์ และ เจริญชัย 2561)

การพัฒนาการท่องเที่ยวของประเทศไทยให้เป็นการท่องเที่ยวอย่างยั่งยืน (Sustainable Tourism) นั้นมีความสำคัญทำให้ประเทศไทยต้องมีการพัฒนาและวางนโยบายที่เหมาะสมให้กับสถานที่ท่องเที่ยวต่างๆ เพื่อตอบสนองความต้องการของนักท่องเที่ยวโดยการนำความคิดเห็นของนักท่องเที่ยวที่มาเยี่ยมชมมาใช้ในการปรับปรุงสถานที่ท่องเที่ยว เช่นการแบ่งกลุ่มนักท่องเที่ยวตามความสนใจ โดยในปัจจุบันการแสดงความพึงพอใจและความประทับใจของนักท่องเที่ยวต่างๆ สามารถทำได้ง่ายขึ้นและ ได้รับความสนใจอยู่บนโลกของสังคมออนไลน์ (Electronic Words of Mouth) ที่มีพื้นที่ส่วนตัวให้ผู้ใช้แสดงความคิดเห็นและบทวิจารณ์บนแพลตฟอร์มต่างๆ เช่น Facebook หรือ Twitter เป็นต้น โดยนักท่องเที่ยวสามารถวิจารณ์ความประทับใจของสถานที่ท่องเที่ยวต่างๆ ที่ตนได้ไปเยือนได้โดยปราศจากการควบคุมจากบริษัทหรือมีส่วนได้ส่วนเสียซึ่งจะทำให้ได้ข้อมูลความรู้สึกที่แท้จริงจากนักท่องเที่ยว (Bi et al. 2019)

โดยในปัจจุบันนักท่องเที่ยวนิยมวางแผนการเดินทางท่องเที่ยวโดยใช้ข้อมูลบนโลกออนไลน์ซึ่งทำให้เว็บไซต์ที่ใช้สำหรับการประเมินและ แนะนำสถานที่ท่องเที่ยวที่มาจากประสบการณ์จริงของนักท่องเที่ยว ได้รับความสนใจมากขึ้นซึ่งข้อมูลเหล่านี้ส่งผลต่อการวางแผนของนักท่องเที่ยว เนื่องจากเว็ปไซต์ที่แนะนำสถานที่ท่องเที่ยวนั้นประกอบไปด้วย ประสบการณ์ มุมมอง ความประทับใจทั้งด้านบวกและด้านลบ พร้อมทั้งมีระบบการให้คะแนนความพึงพอใจสำหรับสถานที่นั้นๆ ให้นักท่องเที่ยวที่สนใจสามารถวางแผนการเดินทางและตัดสินใจเลือกสถานที่ท่องเที่ยวที่เหมาะกับไลฟ์สไตล์ของตัวเองได้

จากข้อมูลจำนวนมากที่เป็นบทวิจารณ์สถานที่ท่องเที่ยวต่างๆ บนโลกออนไลน์ ปัจจุบันได้มีการใช้การเรียนรู้ของเครื่อง (Machine Learning) มาช่วยในการวิเคราะห์ข้อมูลกันอย่างแพร่หลาย เช่น การนำมาช่วยในการวิเคราะห์การตัดสินใจ (Decision Analysis) การเรียนรู้เชิงลึก (Deep Leaning) เป็นต้น ซึ่งโมเดลสำหรับการเรียนรู้ของเครื่องข้อมูลสามารถแบ่งออกได้เป็น 2 ส่วนหลักๆ คือ การเรียนรู้แบบมีผู้สอน (Supervised Learning) และการเรียนรู้แบบไม่มีผู้สอน (Unsupervised

Learning) การเรียนรู้ของเครื่องที่ใช้สำหรับตัวข้อมูลที่เป็นตัวอักษร (Textual Data) จะมีการนำองค์ความรู้ด้านทางการประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) มาประยุกต์ใช้ทำให้โมเดลการเรียนรู้ของเครื่องสามารถวิเคราะห์ข้อมูลต่างๆ ที่เป็นข้อความได้โดยจะถูกเรียกว่าเหมืองข้อความ (Text Mining) ซึ่งตัวอย่างการประยุกต์ใช้งานสำหรับการทำเหมืองข้อความได้แก่ การวิเคราะห์ความรู้สึก (Sentiment Analysis) และระบบแนะนำ (Recommendation System) ที่เกี่ยวกับสถานที่ท่องเที่ยวเป็นต้น

จากเหตุผลที่กล่าวไว้ข้างต้นในงานวิจัยนี้จึงขอนำเสนอวิธีการรับรู้เสียงของลูกค้า (Voice of Customer: VOC) ที่อยู่ในรูปของข้อมูลที่เป็นข้อความซึ่ง VOC ในงานวิจัยนี้คือ ความพึงพอใจหรืออารมณ์ของนักท่องเที่ยวที่มีต่อเยาวราช โดยในงานวิจัยนี้จะใช้ข้อมูลจากบทวิจารณ์และการเสนอความคิดเห็นของชาวต่างชาติที่เป็นภาษาอังกฤษจากเว็บไซต์ Tripadvisor ในส่วนของการวิเคราะห์ข้อมูล โมเดลการเรียนรู้ของเครื่องทั้งแบบมีผู้สอนและไม่มีผู้สอนจะถูกนำมาใช้ในงานวิจัย สำหรับโมเดลแบบมีผู้สอนจะใช้เทคนิคการจำแนก (Classification) ทั้ง 3 เทคนิคต่อไปนี้ การถดถอยโลจิสติค (Logistic Regression) ซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machine: SVM) และนาอีฟเบย์ (Naive Bayes) เพื่อมาใช้ในการจำแนกแสดงความคิดเห็นและบทวิจารณ์ว่าเป็นเชิงบวก (Positive: POS) หรือเชิงลบ (Negative: NEG) ในส่วนของโมเดลการเรียนรู้แบบไม่มีผู้สอนจะใช้วิธีการแบ่งกลุ่มแบบเคมีนและการจัดสรรหัวข้อแฝงหรือแอลดีเอ (Latent Dirichlet Allocation: LDA) มาใช้แบ่งกลุ่มความสนใจของนักท่องเที่ยวเพื่อนำผลลัพธ์มาวิเคราะห์เพื่อหาข้อมูลเชิงลึก (Insight) และสร้างเว็ปแอปพลิเคชันสำหรับวิเคราะห์ความสนใจและความรู้สึกของนักท่องเที่ยว

1.2 วัตถุประสงค์ของงานวิจัย

1) แบ่งกลุ่มความสนใจของนักท่องเที่ยวที่เดินทางมายังเยาวราชโดยใช้บทวิจารณ์ออนไลน์
2) จำแนกประเภทความรู้สึกนักท่องเที่ยวเดินทางมายังเยาวราชโดยใช้บทวิจารณ์ออนไลน์
3) เสนอการวิเคราะห์แนวทางเพื่อให้เข้าใจความรู้สึกของนักท่องเที่ยวเยาวราชในเชิงลึกจากบทวิจารณ์ออนไลน์
4) นำเสนอตัวอย่างการพัฒนาเว็ปแอปพลิเคชั่นสำหรับการวิเคราะห์กลุ่มความสนใจและความรู้สึกของนักท่องเที่ยวที่เดินทางมายังเยาวราช

1.3 ขอบเขตของงานวิจัย

1) ข้อมูลที่เป็นบทวิจารณ์ที่ใช้ในการวิเคราะห์เป็นภาษาอังกฤษจากเว็ปไซต์ Tripadvisor
2) ข้อมูลที่นำมาวิเคราะห์เริ่มตั้งแต่เดือนมกราคมปี พ.ศ.2555 จนถึงเดือน เมษายน พ.ศ. 2563
3) บทวิจารณ์ที่ใช้ในการวิเคราะห์เป็นบทวิจารณ์ของนักท่องเที่ยวที่เคยไปเที่ยวเยาวราช

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ทำให้ทราบกลุ่มความสนใจของนักท่องเที่ยวที่เข้ามาเยี่ยมชมเยาวราชเพื่อนำมาใช้สร้างนโยบายและแผนการตลาดที่เหมาะสม
2) ทราบจุดอ่อนและจุดแข็งด้านต่างๆ ของเยาวราชเพื่อนำมาเป็นแนวทางในการปรับปรุงหรือส่งเสริมการท่องเที่ยวได้อย่างเหมาะสม
3) เป็นแนวทางในการประยุกต์ใช้การนำบทวิจารณ์มาใช้ในการส่งเสริมการท่องเที่ยวสำหรับสถานที่ท่องเที่ยวอื่นๆ

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการศึกษาครั้งนี้ผู้วิจัยได้นำเสนอเนื้อหาที่เน้นถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องโดยมีรายละเอียดของเนื้อหาประกอบ ดังต่อไปนี้

2.1 การเตรียมข้อมูลบทวิจารณ์

2.2 การเรียนรู้แบบไม่มีผู้สอนเพื่อจัดกลุ่มความสนใจของนักท่องเที่ยว

2.3 การเรียนรู้แบบมีผู้สอนเพื่อทำนายความรู้สึกของนักท่องเที่ยว

2.4 การเปรียบเทียบผลการทำนาย

2.5 งานวิจัยที่เกี่ยวข้อง

2.1 การเตรียมข้อมูลบทวิจารณ์

การเตรียมข้อมูลในงานวิจัยฉบับนี้จะเป็นการเตรียมข้อมูลสำหรับข้อความโดยอ้างอิงจาก (Sarkar D. 2019) ซึ่งขั้นตอนที่ใช้ในการเตรียมข้อมูลมีดังนี้

2.1.1 การตัดคำ (Tokenization)

การนำประโยคบทวิจารณ์มาแบ่งออกเป็นคำต่าง (Token) ตามพจนานุกรม (Lexicon) เช่น ประโยค “I love yaowarat !” จะถูกแปลงเป็นคำศัพท์ “I”, “love”, “yaowarat”, “!”

2.1.2 การลบเครื่องหมายวรรคตอน (Punctuation Removal)

การลบเครื่องหมายต่างๆ ที่ไม่จำเป็นในการวิเคราะห์เช่น เครื่องหมายลูกน้ำ (,) เครื่องหมายอัศเจรีย์ (!) เช่น ประโยค “I love yaowarat !” จะถูกแปลงเป็น “I”, “love”, “yaowarat”

2.1.3 การนอร์มอลไลซ์ตัวอักษร (Character Normalization)

การนอร์มอลไลซ์ตัวอักษรคือ การแปลงคำต่างๆ ให้หลายเป็นตัวอักษรในแบบเดียวกันเพื่อความสะดวกในการวิเคราะห์ เช่น การแปลงเป็นตัวอักษรตัวเล็กในภาษาอังกฤษ เนื่องจากคอมพิวเตอร์จะมองตัวอักษรใหญ่และเล็กเป็นคนละตัวกัน เช่น ประโยค “I love yaowarat” จะถูกแปลงเป็น “i”, “love”, “yaowarat”

2.1.4 การลบตัวเลขและกลุ่มคำที่ไม่มีความหมาย (Stop Word Removal)

การลบตัวเลขต่างๆ ในประโยคบทวิจารณ์เพื่อลดความซ้ำซ้อนก่อนในไปใช้วิเคราะห์ เช่น “We found 2 food stores” จะถูกเปลี่ยนเป็น “We”, “found”, “food”, “stores”

2.1.5 การตัดส่วนท้ายของคำ (Word Stemming)

กระบวนการการตัดส่วนท้ายของคำเพื่อให้สามารถลดความซับซ้อนที่เกิดจากคำพูดโดยที่ไม่รู้บริบทและไม่ตรงกับไวยกรณ์ เช่น การแปลงคำว่า “better” เป็น “good”

2.1.6 ถุงคำศัพท์ (Bag of Word: BOW)

ถุงคำศัพท์เป็นการนำข้อความที่ผ่านการเตรียมแล้วมานำเสนอในรูปของเวกเตอร์โดยตัวเลขที่ใส่มีค่าเป็นความถี่ของคำ (ธนภัทร์ 2559) เช่น ข้อความ “I love yaowarat yaowarat” ซึ่งจากพจนานุกรมคำศัพท์ [“I”, “love”, “yaowarat”] จะได้เป็นเวกเตอร์ของข้อความดังนี้ [1, 1, 2] เนื่องจากในประโยคมีคำว่า “yaowarat” เกิดขึ้น 2 ครั้งซึ่งคำว่า “I” และ “love” เกิดขึ้นเพียงครั้งเดียว

2.1.7 การบ่งบอกประเภทของความรู้สึกในบทวิจารณ์ (Class Labeling)

การบ่งบอกความรู้สึกในบทวิจารณ์จะถูกกำหนดโดยระดับคะแนนที่ผู้เขียนบทวิจารณ์ให้ไว้ในข้อมูลที่เก็บมาจากเว็ปไซต์ Tripadvisor การให้คะแนนจะเป็นระบบคะแนนแบบห้าดาว (5 Star Scoring System) โดยระบบการให้คะแนนแบบนี้จะเรียงลำดับความพึงพอใจของลูกค้าได้ โดยเมื่อคะแนนเท่ากับ 1 คือนักท่องเที่ยวมีความพึงพอใจต่ำสุด และคะแนนเท่ากับ 5 คือมีความพึงพอใจสูงที่สุด ซึ่งในงานวิจัยของ Liu and Zhang (2012) ได้อธิบายไว้ว่าการจำแนกความรู้สึกสามารถแบ่งได้เป็นความรู้สึกเชิงบวก (4-5 คะแนน) และความรู้สึกเชิงลบลบ (1-3 คะแนน) ซึ่งเราสามารถนำประเภทความรู้สึกนี้เป็นชุดข้อมูลฝึก

2.2 การเรียนรู้แบบไม่มีผู้สอนเพื่อจัดกลุ่มความสนใจของนักท่องเที่ยว

ในการจัดกลุ่มความสนใจของนักท่องเที่ยวโดยใช้บทวิจารณ์ในงานวิจัยฉบับนี้ได้ใช้ 2 เทคนิคในการวิเคราะห์ได้แก่ การแบ่งกลุ่มแบบเคมีน และการจัดสรรหัวข้อแฝง ซึ่งจะใช้การบ่งกลุ่มแบบเคมีนในการหาจำนวนกลุ่มที่เหมาะสมที่สุดก่อนนำใช้ในโมเดลการจัดสรรหัวข้อแฝงสำหรับหากลุ่มความสนใจ

2.2.1 การแบ่งกลุ่มแบบเคมีน (K-means Clustering)

การแบ่งกลุ่มข้อมูลแบบเคมีน เป็นโมเดลการเรียนรู้ของเครื่องแบบไม่มีผู้สอน ที่ใช้ในการแบ่งกลุ่มของข้อมูลออกเป็น K กลุ่มที่ไม่ซ้อนทับกัน (Hard Clustering) โดยในงานวิจัยฉบับนี้ได้ใช้การแบ่งกลุ่มแบบเคมีนเนื่องจากเป็นวิธีการแบ่งกลุ่มข้อมูลที่ง่ายต่อการตีความ และสามารถใช้ในการหา

จำนวนกลุ่มที่เหมาะสำหรับโมเดลการจัดสรรหัวข้อแฝงได้ (Taecharungroj and Mathayomchan 2019)

การแบ่งกลุ่มแบบเคมีนที่ดีจะต้องมีค่าผลรวมค่าความคลาดเคลื่อนกำลังสอง (Sum of Square Error) น้อยที่สุดเท่าที่เป็นไปได้ (James et al. 2013) ซึ่งเป็นการพิจารณาจากผลรวมค่าระยะทางแบบยูคลิดที่น้อยที่สุดดังสมการที่ 2.1

$$e^* = \min_{e}(\sum_{k=1}^{\mathrm{K}} D_k) \tag{2.1}$$

ซึ่งจะเห็นได้ว่าค่า e^* จะมีค่าน้อยที่สุดมาจากค่าระยะทางยูคลิด D_k ที่มีค่าน้อยๆ และค่า D_k จะมีค่าน้อยได้ต้องเกิดจากการกำหนดจุดกึ่งกลางกลุ่ม (Centroid) ที่เป็นตัวแทนของแต่ละกลุ่มที่เหมาะสม

$$D_k = \sum_{i=1}^{N}(x_i - c_k)^2 \tag{2.2}$$

$$c_k = \frac{1}{N}\sum_{i=1}^{N}(x_{i,k}) \tag{2.3}$$

เมื่อ e^* คือ ค่าความผิดพลาดที่น้อยที่สุด

D_k คือ ระยะทางแบบยุคลิด (Euclidean Distance)

x_i คือ คำศัพท์คำที่ i โดยที่ $i = 1,2,3,...,N$

$x_{i,k}$ คือ คำศัพท์คำที่ i ในกลุ่มที่ k

k คือ ลำดับของกลุ่มโดยที่ $k = 1,2,3,...,K$

K คือ จำนวนกลุ่มที่เหมาะสม

c_k คือ ค่าจุดกึ่งกลางกลุ่ม (Centroid)

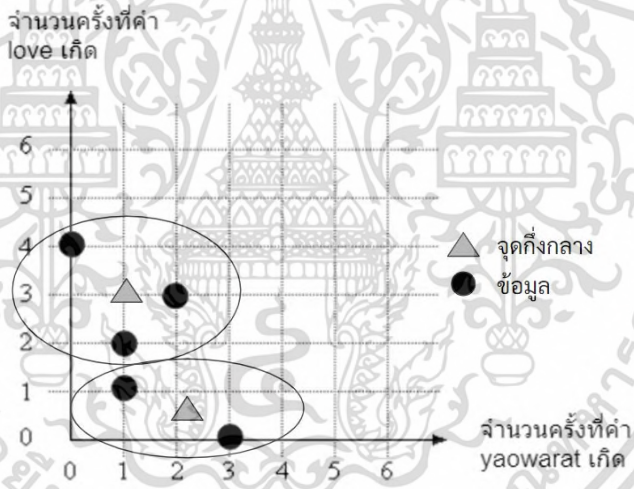
N คือ จำนวนข้อมูลทั้งหมด

โดยทั่วไปการแบ่งกลุ่มจะใช้กับข้อมูลเชิงปริมาณ แต่การทำเหมืองข้อความนั้นต้องมีการแปลงข้อมูลข้อความให้เป็นข้อมูลเชิงปริมาณในรูปเวกเตอร์ของข้อความด้วยวิธีการถุงคำศัพท์ก่อนดัง

แสดงได้ดังตัวอย่างทั้ง 6 ประโยคที่สามารถแปลงเป็นถุงคำศัพท์ได้ดังตารางที่ 2.1 โดยตัวอย่างมีการกำหนดตำแหน่งที่ 1 ในเวกเตอร์แสดงจำนวนคำศัพท์คำว่า “love” และในตำแหน่งที่ 2 ในเวกเตอร์แสดงจำนวนคำศัพท์คำว่า “yaowarat”

ตารางที่ 2.1 ตัวอย่างประโยคและการแปลงถุงคำศัพท์

ประโยคตัวอย่าง	เวกเตอร์หลังจากแปลงด้วยถุงคำศัพท์
1. love yaowarat	[1, 1]
2. yaowarat love	[1, 1]
3. love love love love	[4, 0]
4. love love love yaowarat yaowarat	[3, 2]
5. yaowarat yaowarat yaowarat	[0, 3]
6. love yaowarat love	[2, 1]



รูปที่ 2.1 ผลลัพธ์การทำเคมีนของประโยคที่แปลงด้วยถุงคำศัพท์

ค่าจำนวนคำของคำว่า “love” และ “yaowarat” ของแต่ละประโยคจะถูกนำมาแสดงความสัมพันธ์ดังรูปที่ 2.1 ซึ่งทำให้กำหนดจุดกึ่งกลางกลุ่มทั้ง 2 กลุ่มเพื่อแยกกลุ่มประโยคเป็น 2 กลุ่ม เพื่อกำหนดจุดกึ่งกลางกลุ่มที่เหมาะสม

จากประโยคตัวอย่างทั้ง 6 ประโยคในตารางที่ 2.1 เมื่อนำมาแปลงเป็นเวกเตอร์สมารถแสดงได้ดังรูปที่ 2.1

2.2.2 การกำหนดจำนวนกลุ่มที่เหมาะสม (Optimal Cluster Number Selection)

จากวิธีการเคมีนในหัวข้อที่ 2.2.1.1 วิธีการแบ่งกลุ่มแบบนี้จำเป็นต้องหาค่าจำนวนกลุ่ม K ที่เหมาะสมก่อนใช้ในการกำหนดจำนวนกลุ่มโดยสามารถหาได้ด้วยวิธีการข้อศอก (Elbow) ซึ่งเป็นการพิจารณารูปกราฟของความผิดพลาดกำลังสองน้อยที่สุดเทียบกับจำนวนกลุ่ม (Hastie, Tibshirani, and Friedman 2009 ; Phienthrakul 2008)

2.2.3 การแบ่งกลุ่มบทความ (Text Clustering)

การแบ่งกลุ่มบทความ เป็นการแบ่งกลุ่มข้อมูลที่มีลักษณะเป็นข้อความ หรือบทความต่างๆ ซึ่งการแบ่งกลุ่มบทความมีความแตกต่างจากการแบ่งกลุ่มข้อมูลแบบตารางทั่วไปเนื่องจากต้องมีการเตรียมข้อมูลแบบเฉพาะทางเพื่อให้ข้อความสามารถประมวลผลได้ (Liu and Zhang 2012) โดยในงานวิจัยฉบับนี้ใช้บทวิจารณ์ของนักท่องเที่ยวเป็นข้อมูลที่จะนำไปประมวลผล

2.2.3.1 การแจกแจงดีรีเคล (Dirichlet Distribution)

การแจกแจงดีรีเคลเป็นการแจกแจงที่ใช้สำหรับการหาความรู้ก่อนหน้า (Prior) โดยความรู้ก่อนหน้าที่นำไปใช้ในการวิเคราะห์จะเกิดจากการเปลี่ยนแปลงพารามิเตอร์ α ซึ่งจะทำให้รูปร่างของการแจกแจงเปลี่ยนไป (Bishop 2006) โดยสำหรับการแบ่งกลุ่มนักท่องเที่ยวด้วยวิธีการจัดสรรหัวข้อแฝงนั้นการแจกแจงดีรีเคลจะใช้ในการหาความรู้ก่อนหน้าของบทวิจารณ์และคำศัพท์ที่เกิดขึ้นด้วยการสุ่มค่า α ต่างๆ สมการของการแจกแจงดีรีเคลแสดงดังสมการที่ (2.4)

$$f(X_i) = \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} X_i^{\alpha_i - 1} \qquad (2.4)$$

เมื่อ X_i คือ บทวิจารณ์ลำดับที่ i โดยที่ $i = 1,2,3,...,K$

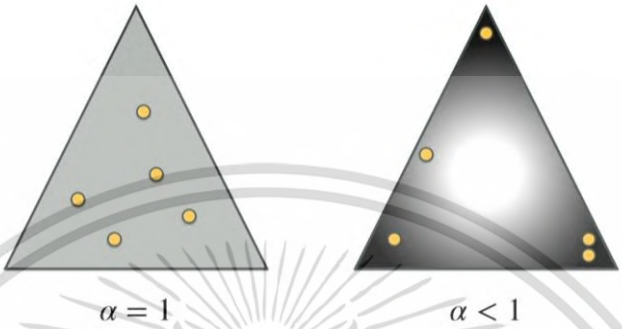
α_i คือ พารามิเตอร์ของการแจกแจงดีรีเคลลำดับที่ i โดยที่ $i = 1,2,3,...,K$

K คือ จำนวนกลุ่มที่เหมาะสม

ตัวอย่างการอธิบายผลกระทบของค่า α

เนื่องจากการปรับเปลี่ยนพารามิเตอร์ α ส่งผลให้รูปร่างของการแจกแจงเปลี่ยนไปทำให้มีผลต่อการกระจายตัวของคำศัพท์และ การกระจายตัวของบทวิจารณ์ โดยในตัวอย่างนี้จะเป็นตัวอย่างของการกระจายของคำศัพท์ที่อยู่การแจกแจงดีรีเคล โดยในตัวอย่างนี้กำหนดให้จำนวนหัวข้อที่เหมาะสม

ซึ่งได้มาจากวิธีการข้อศอก K เท่ากับ 3 ซึ่งหมายความว่าหัวแฝงจะมีทั้งหมด 3 หัวข้อทำให้รูปร่างของการแจกแจงแบบดีรีเคลนั้นมีรูปร่างเป็นสามเหลี่ยมดังรูปที่ 2.2 ซึ่งจุดสีเหลืองในรูปคือคำศัพท์ต่างๆ ที่ปรากฏอยู่ในบทวิจารณ์และ พื้นที่สีแสดงถึงโอกาศพบเจอคำศัพท์ซึ่งไล่โทนสีจากสีดำคือโอกาสพบคำศัพท์สูงไปยังสีขาวซึ่งมีโอกาสพบคำศัพท์ต่ำ



รูปที่ 2.2 การเปลี่ยนแปลงรูปร่างของการแจกแจง

ที่มา ปรับปรุงจาก Luis 2020

จากรูปที่ 2.2 เมื่อ α เท่ากับ 1 คำศัพท์ที่กระจายตัวอยู่ในการแจกแจงจะไม่เป็นระเบียบซึ่งในมุมของการจัดสรรหัวข้อแฝง หมายถึงการที่คำศัพท์ทั้งหมดที่จะถูกสุ่มขึ้นมานั้นไม่ได้มีการแบ่งแยกอย่างชัดเจนว่าคำศัพท์อยู่ในหัวข้อแฝงใด ซึ่งต่างจากเมื่อค่า α ลดลงน้อยกว่า 1 จะสังเกตได้ว่าคำศัพท์มีการเคลื่อนที่เข้าสู่มุม ซึ่งหมายความว่าในการสุ่มแต่ละครั้งเราสามารถแยกคำศัพท์ที่สุ่มได้อย่างชัดเจนมากขึ้นว่าคำศัพท์นั้นมาจากหัวข้อแฝงใด เนื่องจากคำศัพท์อยู่ใกล้มุมมากขึ้น ซึ่งยิ่งคำศัพท์อยู่ใกล้มุมยิ่งแสดงว่าคำศัพท์นั้นมีโอกาสเกิดในขึ้นในหัวข้อแฝงนั้นมากขึ้นเท่านั้น

2.2.3.2 การจัดสรรหัวข้อแฝง (Latent Dirichlet Allocation: LDA)

การจัดสรรหัวข้อแฝงเป็นโมเดลสำหรับการสร้างหัวข้อ (Topic) ซึ่งในที่นี้อาจจะเป็นกลุ่มที่ถูกซ่อนอยู่ในเอกสาร หรือบทวิจารณ์ต่างๆ โดยในแต่ละหัวข้อแฝงจะประกอบไปด้วยคำศัพท์หลากหลายคำอยู่รวมกันโดยแต่ละคำนั้นจะมีค่าความสำคัญ (Weight) แตกต่างกันตามกลุ่ม ซึ่งทำให้คำศัพท์เหล่านี้สามารถนำไปบ่งบอกลักษณะเฉพาะของกลุ่มได้ โดยการจัดสรรหัวข้อแฝงสามารถเขียนได้ดังสมการ (2.5)

$$P(W,Z,\theta,\varphi \mid \alpha,\beta) = \prod_{m=1}^{M} P(\theta_m \mid \alpha) \prod_{k=1}^{K} P(\varphi_k \mid \beta) \prod_{t=1}^{N} P(Z_{m,t} \mid \theta_m) P(W_{m,t} \mid \varphi_{Z_{m,t}}) \quad (2.5)$$

เมื่อ $Z_{m,t}$ คือ หัวข้อแฝงที่อยู่ในกลุ่มของบทวิจารณ์ที่ m ของการสุ่มครั้งที่ t

θ_m คือ ความน่าจะเป็นของแต่ละบทวิจารณ์ลำดับที่ m

α คือ พารามิเตอร์ควบคุมการกระจายตัวของหัวข้อ

φ_k คือ ความน่าจะเป็นของบทวิจารณ์ในแต่ละหัวข้อที่ k

β คือ พารามิเตอร์ควบคุมการกระจายตัวของคำ

m คือ ลำดับของบทวิจารณ์ $m = 1,2,3,...,M$

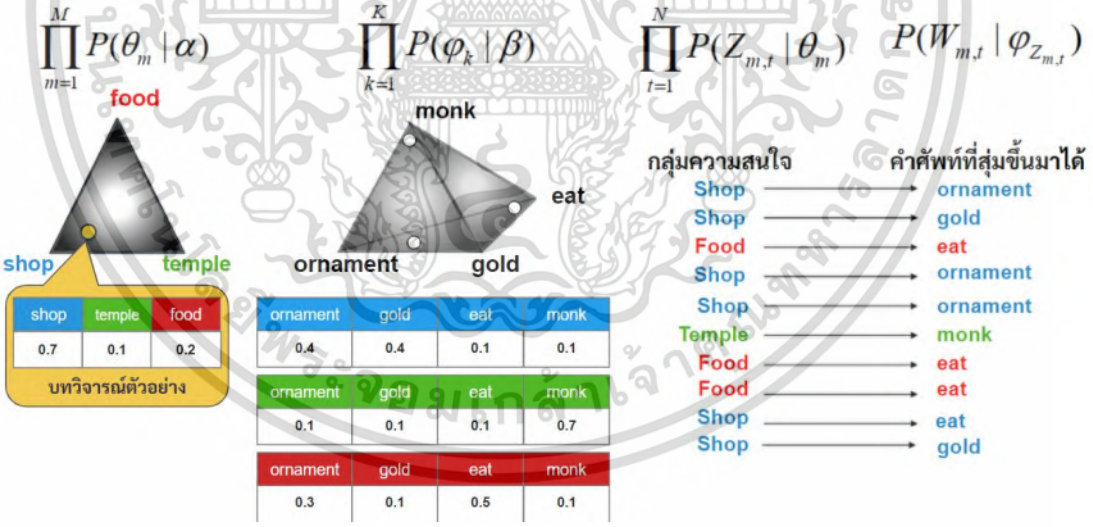
N คือ จำนวนคำศัพท์ทั้งหมด

K คือ จำนวนกลุ่มที่เหมาะสมที่

$\varphi_{Z_{m,t}}$ คือ กลุ่มหัวข้อแฝงจากความน่าจะเป็นของหัวข้อแฝงที่อยู่ในบทวิจารณ์ที่ m ซึ่งได้จากการสุ่มครั้งที่ t

$W_{m,t}$ คือ คำศัพท์ที่สุ่มได้จากการสุ่มคำที่อยู่ในกลุ่มหัวข้อแฝงจากความน่าจะเป็นของหัวข้อแฝงของเอกสารลำดับที่ m ซึ่งได้จากการสุ่มครั้งที่ t

ตัวอย่างขั้นตอนการทำงานของโมเดล LDA

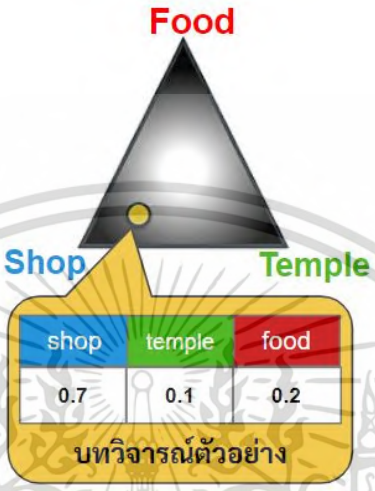


รูปที่ 2.3 ขั้นตอนการทำงานทั้งหมดของโมเดลการจัดสรรหัวข้อแฝง

ที่มา ปรับปรุงจาก Luis 2020

เนื่องจากการทำงานของโมเดล LDA มีความซับซ้อนจึงดังรูปที่ 2.3 จึงมีความจำเป็นที่จะต้องกำหนดตัวแปร และประโยคตัวอย่างที่ใช้ให้เหมาะสมกับตัวอย่าง ซึ่งตัวอย่างนี้ดัดแปลงมาจาก (Luis 2020) โดยกำหนดให้บทวิจารณ์ตัวอย่างที่ใช้คือ “Oh no!! the ornament ornament ornament ornament ornament ornament ornament monk eat eat” เมื่อประโยคนี้ผ่านกระบวนการ

การเตรียมข้อมูลในแล้วจะกลายเป็น “ornament”, “gold”, “gold”, “ornament”, “ornament”, “ornament”, “ornament”, “monk”, “eat”, “eat” ซึ่งทำให้บทวิจารณ์นี้มีมีความยาว 10 คำ และประกอบด้วยคำศัพท์ 4 คำศัพท์ได้แก่ “ornament”, “monk”, “eat” และ “gold” โดยมีจำนวนกลุ่มความสนใจ K เท่ากับ 3 กลุ่มคือ กลุ่ม shop กลุ่ม food และกลุ่ม temple ขั้นตอนการทำงานของ LDA มีวิธีการดังนี้

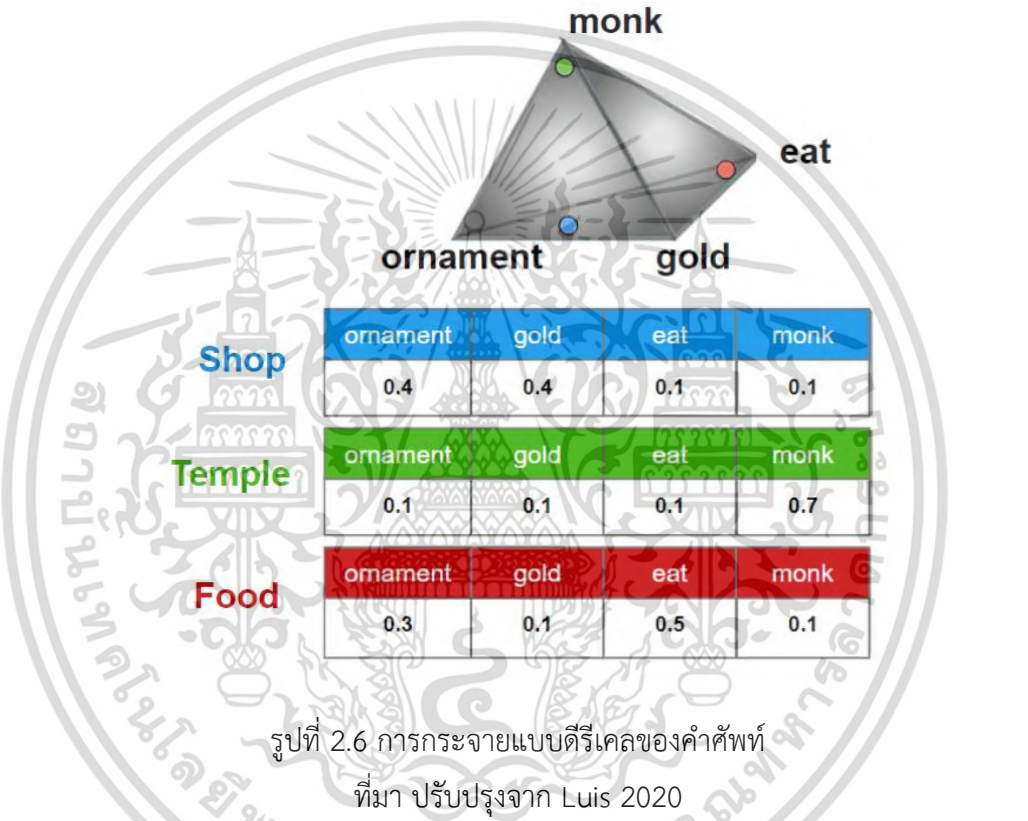


รูปที่ 2.4 การกระจายแบบดีรีเคลของกลุ่มความสนใจ
ที่มา ปรับปรุงจาก Luis 2020

1. จากค่า K เท่ากับ 3 ทำให้สามารถแสดงการกระจายแบบดีรีเคลเป็นสามมิติที่มีลักษณะเป็นรูปสามเหลี่ยม ซึ่งจุดสีเหลืองข้างในการกระจายคือ บทวิจารณ์ตัวอย่างที่เกิดขึ้น โดยบทวิจารณ์ตัวอย่างนี้ตั้งอยู่ในตำแหน่งที่ใกล้กับมุมของกลุ่มความสนใจ shop มากที่สุดซึ่งแสดงให้เห็นว่าบทวิจารณ์ฉบับนี้มีความน่าจะเป็นบทวิจารณ์ที่อยู่ในกลุ่ม shop ซึ่งสามารถเขียนความน่าจะเป็นของแต่ละกลุ่มได้ตามตารางในรูปที่ 2.4

รูปที่ 2.5 กลุ่มความสนใจที่ถูกสุ่มขึ้นมา
ที่มา ปรับปรุงจาก Luis 2020

2. จากรูปที่ 2.5 เมื่อได้ความน่าจะเป็นของแต่ละหัวข้อมาแล้วต่อไปจะเป็นการสุ่มกลุ่มความสนใจที่จะเกิดขึ้น เนื่องจากความยาวของคำในประโยคตัวอย่างเท่ากับ 10 ดังนั้น กลุ่มความสนใจจะถูกสุ่มออกมา 10 ครั้ง โดยได้ผลลัพธ์ดังรูปที่ 2.5 จากผลลัพธ์การสุ่มพบว่าได้ กลุ่มความสนใจ Shop ทั้งหมด 6 ครั้ง กลุ่มความสนใจ Food 3 ครั้ง กลุ่มความสนใจ Temple 1 ครั้ง ซึ่งจะเห็นว่าผลลัพธ์ที่เกิดจากการสุ่มมีความผิดพลาด ทำให้ไม่ได้เป็นไปตามค่าความน่าจะเป็นในการกระจายแบบดีรีเคล



รูปที่ 2.6 การกระจายแบบดีรีเคลของคำศัพท์

ที่มา ปรับปรุงจาก Luis 2020

3. จากการพิจารณาเนื้อหาที่อยู่ในบทวิจารณ์ตัวอย่างพบว่ามีคำศัพท์ทั้งหมด 4 คำเกิดขึ้นในบทวิจารณ์ได้แก่ “ornament”, “monk”, “eat” และ “gold” ทำให้เราสามารถสร้างการกระจายดีรีเคลแบบ 4 มิติที่มีลักษณะคล้ายพีรามิดได้ โดยในการกระจายนี้จะมีวงกลมสีต่างๆ ซึ่งตำแหน่งของวงกลมนี้แสดงถึงความน่าจะเป็นของคำที่จะเกิดขึ้นในกลุ่มความสนใจนั้น โดยวงกลมสีฟ้าคือกลุ่มความสนใจ Shop สีแดงคือกลุ่มความสนใจ Food และสีเขียวคือกลุ่มความสนใจ Temple ดังรูปที่ 2.6
4. จากรูปจะเห็นได้ว่าวงกลมสีฟ้ามีตำแหน่งอยู่ใกล้มุม “ornament” กับ “gold” มากกว่า ดังนั้นความน่าจะเป็นของคำว่า “ornament” กับ “gold” มากกว่าคำว่า “monk” และ “eat” สำหรับกลุ่ม shop ในทำนองเดียวกัน วงกลมสีเขียวอยู่ใกล้คำว่า “monk”

มากที่สุดทำให้มีความน่าจะเป็นของ “monk” มากที่สุดในกลุ่ม temple และวงกลมสีแดงอยู่ใกล้คำว่า “eat” มากที่สุดทำให้มีความน่าจะเป็นของคำว่า “eat” มากที่สุดในกลุ่ม food

5. จากกลุ่มความสนใจที่ถูกสุ่มขึ้นมาในรูปที่ 2.5 จะมีการสุ่มคำศัพท์ต่างๆ ที่จะเกิดขึ้นในแต่ละกลุ่มความสนใจจากการกระจายแบบดีรีเคลของคำศัพท์ในรูปที่ 2.5 ซึ่งผลลัพธ์ของคำศัพท์ในแต่ละกลุ่มความสนใจจะแสดงดังรูปที่ 2.7 พบว่าการสุ่มครั้งแรกได้คำว่า ornament ซึ่งเป็นคำที่มีความน่าจะเป็นสูงสุดในกลุ่มความสนใจ Shop และเมื่อสุ่มไปเรื่อยๆ จนถึงลำดับที่ 10 เราจะได้คำศัพท์มา 10 คำศัพท์ซึ่งจะเป็นจำนวนคำศัพท์เฉพาะ

รูปที่ 2.7 คำศัพท์ที่สุ่มขึ้นมาได้ในแต่ละกลุ่มความสนใจ

ที่มา ปรับปรุงจาก Luis 2020

ทำซ้ำขั้นตอนที่ 2 สำหรับบทวิจารณ์ถัดไป ซึ่งเมื่อทำครบทุกบทวิจารณ์แล้วเราจะได้คำศัพท์มาบทวิจารณ์ละ 10 คำ ดังนั้นเราสามารถนำ คำศัพท์ที่เกิดขึ้นในข้อที่ 4 มาหารกับจำนวนคำศัพท์นั้นๆ ที่เกิดขึ้นทั้งหมดได้ ซึ่งผลลัพธ์ที่เกิดขึ้นจะเป็น ค่าน้ำหนักของแต่ละคำซึ่งเป็นผลลัพธ์ของโมเดล

2.3 การเรียนรู้แบบมีผู้สอนเพื่อทำนายความรู้สึกของนักท่องเที่ยว

ในการทำนายความรู้สึกของนักท่องเที่ยวนั้นเราจำเป็นต้องใช้บทวิจารณ์และคำตอบ (Label) เพื่อที่จะมาฝึกสอน (Train) และทดสอบ (Test) ให้แก่โมเดลประเภทการเรียนรู้แบบมีผู้สอน โดยในงานวิจัยฉบับได้ใช้โมเดลทั้งหมด 3 โมเดลได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติคและ ซัพพอร์ตเวกเตอร์แมชชีน โดยใช้การแบ่งข้อมูลสำหรับการฝึกสอนเป็น 70% และทดสอบ 30%

2.3.1 นาอีฟเบย์ (Naïve Bayes)

นาอีฟเบย์เป็นโมเดลการเรียนรู้แบบมีผู้สอนประเภทการจำแนกซึ่งใช้หลักการความน่าจะเป็นและกฎของเบย์ในการจำแนกประเภทของข้อมูล ซึ่งสมมุติฐานของวิธีการจำแนกข้อมูลประเภทนี้คือ

คุณลักษณะ (Feature) ของข้อมูลต้องไม่ขึ้นต่อกันอย่างมีเงื่อนไข (Conditionally Independent) โดยผลลัพธ์ของนาอีฟเบย์คือเป็นการจำแนกประเภทของความรู้สึกเป็นบวกหรือลบ (Juan and Ney 2002 ; Géron 2019) โดยนาอีฟเบย์จะจำแนกประเภทของข้อมูลโดยเรียนรู้ความน่าจะเป็นของโอกาสเกิดประเภทของความรู้สึก และความน่าจะเป็นของคำที่ปรากฏในประโยค โดยตัวอย่างการคำนวณแสดงไว้ดังภาคผนวก ข

$$P(c_i \mid d_j) = \arg\max P(c_i) \prod_{j=1}^{N} P(d_j \mid c) \tag{2.6}$$

เมื่อ c_i คือ ประเภทของข้อมูล (Label) โดยที่ $i = POS, NEG$

d_j คือ ความถี่ของคำศัพท์คำที่ j โดยที่ $j = 1, 2, 3, \ldots, N$

N คือ จำนวนคำศัพท์ทั้งหมด

2.3.2 การถดถอยเชิงโลจิสติค (Logistic Regression)

การถดถอยเชิงโลจิสติดเป็นโมเดลการเรียนรู้แบบมีผู้สอนที่ใช้สำหรับการจำแนกประเภทข้อมูลโดยการนำฟังก์ชันโลจิสติคมาใช้กับผลลัพธ์ของโมเดลการถดถอยเชิงเส้นทำให้ผลลัพธ์ออกมากลายเป็นค่า 0 ถึง 1 ซึ่งเป็นการบอกประเภทของข้อมูลว่าเป็นข้อมูลประเภทใด (Russell and Norvig 2002 ; Murphy 2012 ; James et al. 2013)

จากสมการของการถดถอยเชิงเส้นดังสมการที่ (2.7)

$$y = w^T x \tag{2.7}$$

เมื่อ y คือ เวกเตอร์ของผลทำนาย

w คือ เวกเตอร์ของค่าน้ำหนัก

x คือ เวกเตอร์ของคำศัพท์

เมื่อนำฟังก์ชันโลจิสติค (Logistic) เพิ่มเข้าไปในสมการที่ (2.7) จะได้เป็นดังสมการที่ (2.8) และสมการที่ (2.9)

$$\hat{y} = Logit(w^T x) \tag{2.8}$$

$$\hat{y} = \frac{1}{1 + e^{-(w^T x)}} \tag{2.9}$$

สมการคำตอบของการถดถอยเชิงโลจิสติคจะเป็นค่าที่มีค่าความน่าจะเป็นสูงสุดในรูปของค่าความน่าแสดงไว้ดังนี้

$$P(y = 1 \mid x, w) = Logit(\hat{y}) \tag{2.10}$$
$$P(y = 0 \mid x, w) = Logit(1 - \hat{y})$$

โดยการถดถอยเชิงโลจิสติคมีฟังก์ชันต้นทุนคือ

$$loss = -\frac{1}{N}\sum_{i=1}^{N}[y_i Logit(\hat{y}_i) + (1 - y_i) Logit(1 - \hat{y}_i)] \tag{2.11}$$

เมื่อ $\hat{y}_i$ คือ ผลทำนายลำดับที่ i โดยที่ $i = 1,2,3,...,N$

y_i คือ คำตอบลำดับที่ i โดยที่ $i = 1,2,3,...,N$

N คือ จำนวนคำศัพท์ทั้งหมด

2.3.3 ซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น (Linear Support Vector Machine)

ซัพพอร์ตเวกเตอร์แมชชีน เป็นโมเดลการเรียนรู้แบบมีผู้สอนที่แบ่งได้เป็น 2 ประเภทคือ Support Vector Classifier ซึ่งเป็นโมเดลสำหรับการจำแนกประเภทข้อมูลและ Support Vector Regression ซึ่งเอาไว้สำหรับการทำนายข้อมูลที่มีค่าต่อเนื่อง โดยในงานวิจัยนี้จะใช้ Support Vector Classifier ในการจำแนกประเภทของซึ่งเป็นการจำแนกข้อมูลออกเป็นหลายๆ ประเภทด้วยระนาบหลายมิติ (Hyper Plane) โดยระนาบที่ถูกสร้างขึ้นนี้จะทำหน้าที่เป็นเส้นแบ่ง (Separator) ประเภทของข้อมูลโดยถ้าผู้ใช้ไม่มีความรู้ก่อนหน้า (Prior Knowledge) เกี่ยวกับข้อมูล ซัพพอร์ตเวกเตอร์แมชชีนถือว่าเป็นโมเดลตัวเลือกหนึ่งที่ควรใช้ในการจำแนกข้อมูล ซัพพอร์ตเวกเตอร์แมชชีนแบบเชิงเส้นนั้นจะได้ผลลัพธ์ที่ดีเมื่อข้อมูลมีลักษณะที่สามารถแบ่งแบบเชิงเส้น (Linear Separability) ได้เท่านั้น (อารยา 2556 ; Phienthrakul 2008) โดยการจำแนกประเภทข้อมูลจะมีประสิทธิภาพเมื่อข้อมูลมีการกระจายตัวแบบเชิงเส้นที่จะได้สร้างเส้นขอบ (Margin) ที่เหมาะสมเพื่อให้สามารถแยกข้อมูลได้ดีที่สุดซึ่งสมการของซัพพอร์ตเวกเตอร์แมชชีนสามารถเขียนได้ดังต่อไปนี้

$$\hat{y} = \begin{cases} 0; & w^T x + b < 0 \\ 1; & w^T x + b > 0 \end{cases} \tag{2.12}$$

จากสมการที่ 2.12 ซึ่งเป็นสมการเส้นตรงที่ใช้ใน คำตอบของการทำนายคือ 0 และ 1 ซึ่งสำหรับการจำแนกความรู้สึกของนักท่องเที่ยวนั้นเราจะใช้ 0 แทนความรู้สึกเชิงบวก (POS) และ 1 แทนความรู้สึกเชิงลบ (NEG) เราสามารถเขียนให้อยู่ในรูปของการจำแนกเชิงเส้นได้โดยใช้ฟังก์ชัน Signum ซึ่งสามารถเขียนได้เป็น

$$\hat{y} = sign(w^T x + b) \tag{2.13}$$

$$\hat{y} = \begin{cases} 0; & sign(w^T x + b) < 0 \\ 1; & sign(w^T x + b) > 0 \end{cases} \tag{2.14}$$

เมื่อ $\hat{y}$ คือ เวกเตอร์ของผลทำนาย

w คือ เวกเตอร์ของค่าน้ำหนัก

x คือ เวกเตอร์ของคำศัพท์

b คือ เวกเตอร์ของจุดตัดแกน

$sign$ คือ ฟังก์ชัน Signum ซึ่งมีค่าเป็น 0 เมื่อ $x > 0$ และ 1 เมื่อ $x < 0$

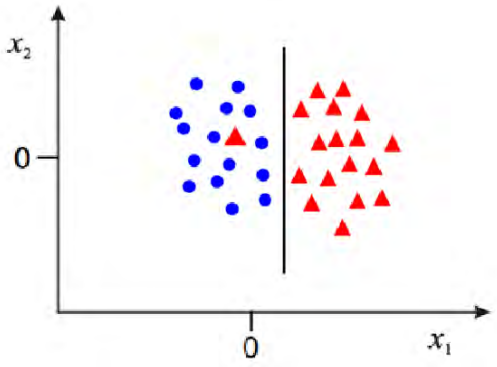
ซึ่งฟังก์ชันต้นทุนที่ซัพพอร์ตเวกเตอร์แมชชีนใช้คือ Hinge Loss (Géron 2019) ตามสมการ (2.15)

$$loss = \sum_{i=1}^{N} [1 - y_i(\hat{y}_i)] \tag{2.15}$$

เมื่อ y_i คือ ผลจริงของข้อมูลลำดับที่ i โดยที่ $i = 1,2,3,...N$

$\hat{y}_i$ คือ ผลทำนายของข้อมูลลำดับที่ i โดยที่ $i = 1,2,3,...N$

N คือ จำนวนคำศัพท์ทั้งหมด



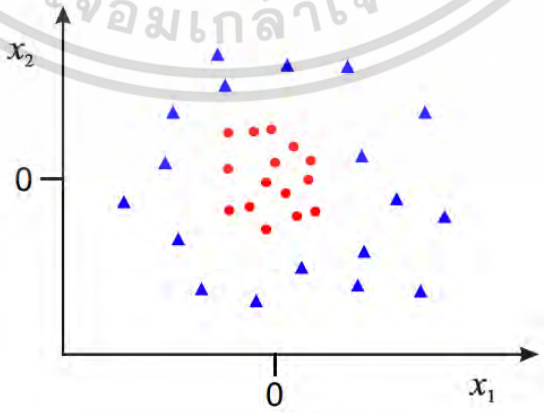
รูปที่ 2.8 การกระจายของข้อมูลที่แบ่งแบบเชิงเส้นได้

ที่มา Zisserman 2015

จากรูปที่ 2.8 มีข้อมูลอยู่สองประเภทคือข้อมูลทั้งสีน้ำเงินและสีแดง โดยข้อมูลลักษณะการกระจายที่ไม่ซับซ้อนและข้อมูลมีลักษณะที่สามารถแบ่งแบบเชิงเส้นได้ โดยพบว่าใช้ซัพพอร์ตเวกเตอร์แมชชีนแบบเชิงเส้น ซึ่งมีลักษณะเป็นเส้นตรงสีดำมาใช้แบ่งข้อมูลแล้วพบว่ามีเพียงแค่ข้อมูลสีแดง 1 ตัวที่มีการจำแนกที่ผิดพลาด

2.3.4 ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้น (Non-Linear Support Vector Machine)

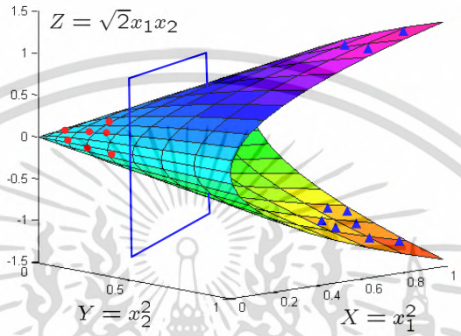
ในการจำแนกข้อมูลโดยใช้ซัพพอร์ตเวกเตอร์แมชชีนแบบเชิงเส้นนั้นข้อมูลที่นำมาใช้จำแนกต้องมีลักษณะที่สามารถแบ่งแบบเชิงเส้นได้ ซึ่งการใช้ซัพพอร์ตเวกเตอร์แมชชีนแบบเชิงเส้นกับข้อมูลที่ไม่สามารถจำแนกแบบเชิงเส้นได้นั้นจะทำให้ไม่สามารถทำการจำแนกข้อมูลได้อย่างมีประสิทธิภาพ ดังเช่นในรูปที่ 2.9 ซัพพอร์ตเวกเตอร์แมชชีนแบบเชิงเส้นไม่สามารถสร้างเส้นตรงมาแบ่งข้อมูลทั้งสองประเภทได้อย่างมีประสิทธิภาพดังนั้นจำเป็นต้องมีการแปลงปริภูมิของคุณลักษณะต่างๆ ทำให้คุณลักษณะที่มีจำนวนมิติหรืออยู่ในปริภูมิที่สูงขึ้นโดยในงานวิจัยฉบับนี้ได้เลือกวิธีเคอร์เนลในการแปลงข้อมูล



รูปที่ 2.9 การกระจายของข้อมูลที่ไม่สามารถแบ่งแบบเชิงเส้นได้

ที่มา Zisserman 2015

จากรูปที่ 2.10 มีข้อมูลอยู่สองประเภทคือข้อมูลทั้งสีน้ำเงินและสีแดง ซึ่งลักษณะการกระจายของข้อมูลแบบนี้มีความซับซ้อนกว่ารูปที่ 2.9 และไม่สามารถแบ่งแบบเชิงเส้นได้เนื่องจากไม่สามารถใช้ซัพพอรต์เวกเตอร์แมชชีนที่เป็นเส้นตรงมาจำแนกข้อมูลสองประเภทนี้ได้ ดังนั้นจึงต้องมีแปลงข้อมูลด้วยวิธีของเคอร์เนลดังกล่าวให้มีปริภูมิที่สูงขึ้นเพื่อช่วยในการจำแนกดังรูปที่ 2.11 ซึ่งเมื่อแปลงแล้วจะเห็นได้ว่าเราสามารถนำสมการเชิงเส้นที่มีมิติมากขึ้น (สี่เหลี่ยมสีนำเงิน) มาใช้ในการจำแนกข้อมูลได้ดังรูปที่ 2.10



รูปที่ 2.10 การแปลงข้อมูลให้อยู่ในปริภูมิที่สูงขึ้น

ที่มา Zisserman 2015

การทำเคอร์เนลใช้ในการสร้างฟังก์ชันการตัดสินใจที่ไม่เป็นเชิงเส้น (Non-Linear Decision Function) เพื่อให้ ซัพพอร์ตเวกเตอร์แมชชีน เชิงเส้นสามารถสร้างเส้น Hyper plane มาทำการจำแนกข้อมูลที่ถูกแปลงให้อยู่ในมิติที่สูงขึ้นดังสมการ (2.16)

$$\hat{y} = sign(\sum_{i=1}^{N} \alpha_i (\phi(x) \cdot \phi(x)) + b) \qquad (2.16)$$

เมื่อ $\hat{y}$ คือ เวกเตอร์ผลทำนาย

α_i คือ ค่าน้ำหนักของข้อมูลตัวที่ i โดย $i = 1,2,3,...,N$

$\phi(x)$ คือ คำศัพท์ที่ถูกแปลงด้วยวิธีเคอร์เนล

b คือ เวกเตอร์จุดตัดแกน

N คือ จำนวนคำศัพท์ทั้งหมด

$sign$ คือ ฟังก์ชัน Signum ซึ่งมีค่าเป็น 1 เมื่อ $x \geq 0$ และ 1 เมื่อ $x < 0$

อย่างไรก็ตามเนื่องจากฟังก์ชันการแจกแจงของซัพพอร์ตเวกเตอร์แมชชีนเกิดจากการทำผลคูณภายใน (Inner Product) ทำให้เพียงแค่กำหนดฟังก์ชันของเคอร์เนล (Kernel Function) ที่

ถูกต้องและเหมาะสมซึ่งเป็นฟังก์ชันที่ใช้ในการแปลงเพื่อเชื่อมปริภูมิมิติต่างๆ ก็สามารถทำการแยกประเภทข้อมูลที่ไม่สามารถแยกแบบเชิงเส้นได้วิธีการนี้ถูกเรียกว่า Kernel Trick ซึ่งเคอร์เนลฟังก์ชันที่ใช้ในงานวิจัยนี้จะมีด้วยกัน 3 ฟังก์ชันที่เป็นที่นิยม (Kononenko and Kukar 2007) ได้แก่ ฟังก์ชันเคอร์เนลแบบเชิงเส้น ฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยล และฟังก์ชันเคอร์เนลแบบ Radial Basis Function (RBF) โดยตัวแปรของฟังก์ชันเคอร์เนลจะถูกหาค่าที่เหมาะสมตาม (Hsu, Chang, and Lin 2003)

ฟังก์ชันเคอร์เนลแบบเชิงเส้น

$$K(x,x) = \phi(x) \cdot \phi(x) \tag{2.17}$$

ฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยล

$$K(x,x) = (\phi(x) \cdot \phi(x))^d \tag{2.18}$$

ฟังก์ชันเคอร์เนลแบบ Radial Basis Function

$$K(x,x) = \exp\ (-\gamma \,||\, \phi(x) - \phi(x) \,||) \tag{2.19}$$

เมื่อ $\phi(x)$ คือ ข้อมูลที่ถูกแปลงด้วยวิธีเคอร์เนล

d คือ ตัวแปรองศาของฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยล

γ คือ พารามิเตอร์ที่ใช้ปรับค่าความสัมพันธ์ของความอคติและความแปรปรวน

ซึ่งมีซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นฟังก์ชันต้นทุนคือ

$$loss = \sum_{i=1}^{N} [1 - y_i \, f_k(x_i)] \tag{2.20}$$

โดยที่ y_i คือ ผลทำนายลำดับที่ i โดยที่ $i = 1,2,3,...N$

x_i คือ คำศัพท์ลำดับที่ i โดยที่ $i = 1,2,3,...N$

f_k คือ ฟังก์ชันตัดสินใจที่ผ่านการแปลงเคอร์เนลแบบเชิงเส้น แบบโพลิโนเมี่ยล หรือแบบ Radial Basis Function

N คือ จำนวนคำศัพท์ทั้งหมด

2.3.5 การเรกูลาไรซ์ (Regularization)

การเรกูลาไรซ์คือ การทำให้ค่าน้ำหนักพารามิเตอร์น้ำหนักบางตัวภายในโมเดลไม่สูงมากจนเกินไปเพื่อลดการเกิด Overfitting (Bühlmann and Van De Geer 2011) ซึ่งการที่พารามิเตอร์มีน้ำหนักสูงอาจจะทำให้การตัดสินใจของโมเดลผิดไป เนื่องจากตัวโมเดลยึดติดกับข้อมูลที่ถูกนำไปใช้เรียนรู้มากเกินไป โดยการเรกูลาไรซ์จะเป็นการเพิ่มพจน์บทลงโทษ (Penalty) เข้าไปในสมการฟังก์ชันต้นทุน เพื่อลดค่าน้ำหนักที่มากเกินไปของพารามิเตอร์ โดยพจน์บทลงโทษที่นิยมแบ่งออกได้เป็น 2 ประเภทได้แก่ Ridge และ Lasso จากการทำเรกูลาไรซ์ทำให้ฟังก์ชันต้นทุนกลายเป็น Loss รวมกับพจน์ของบทลงโทษ (James et al. 2013) ดังแสดงในสมการที่ (2.22) และ (2.23)

$$J = loss(y, \hat{y}) + penelty \tag{2.21}$$

สมการการเรกูลาไรซ์ฟังก์ชั้นต้นทุนด้วย Lasso แสดงดังนี้

$$J = loss(y, \hat{y}) + (\lambda \sum_{i}^{N} || w_i ||) \tag{2.22}$$

สมการการเรกูลาไรซ์ฟังก์ชั้นต้นทุนด้วย Ridge แสดงดังนี้

$$J = loss(y, \hat{y}) + (\lambda \sum_{i}^{N} || w_i ||^2) \tag{2.23}$$

ในทางปฏิบัตินิยมใช้ตัวแปร C แทน λ ซึ่ง $C = \frac{1}{\lambda}$ เพื่อแสดงถึงความรุนแรงในการเรกูลไรซ์โดย C มีค่าน้อยจะหมายความว่ามีความรุนแรงของการเรกูลาไรซ์มาก ซึ่งการตั้งค่า C ควรตั้งให้มีค่าเพิ่มขึ้น และลดลงแบบเอกซ์โพเนนเชียล โดยการเลือกโฒเดลที่ดีที่นั้นเราจะเลือกจากค่า C ที่ทำให้โมเดลมีค่าความถูกต้องสูงสุด (Hsu et al 2003 ; Hastie et al. 2004)

ทำให้สมการที่ (2.24) และ (2.25) เป็น

$$J = loss(y, \hat{y}) + (\frac{1}{C} \sum_{i}^{N} || w_i ||) \tag{2.24}$$

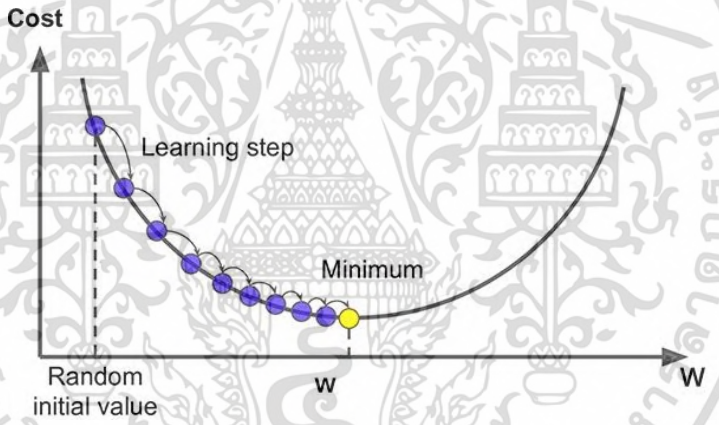
$$J = loss(y, \hat{y}) + (\frac{1}{C} \sum_{i}^{N} || w_i ||^2) \tag{2.25}$$

อย่างไรก็ตามสำหรับโมเดลเคอร์เนลซัพพอร์ตเวกเตอร์แมชชีน พจน์ของบทลงโทษสามารถเขียนในรูปของการรวมกันแบบเชิงเส้นของการทำวิธีของเคอร์เนล (Kimeldorf and Wahba 1971 ;

Hastie et al. 2004 ; Moguerza and Muñoz 2006 ; Bao 2012) ทำให้ไม่มีพจน์บทลงโทษสำหรับโมเดลเคอร์เนลซัพพอร์ตเวกเตอร์แมชชีน

2.3.6 การเคลื่อนลงตามความชัน (Gradient Descent)

การเคลื่อนลงตามความชันเป็นวิธีในการหาค่าที่เหมาะสมในการทำนายข้อมูลด้วยการเรียนรู้แบบมีผู้สอนให้กับฟังก์ชันต้นทุนของโมเดลใดๆ ที่กล่าวไว้ในหัวข้อที่ 2.31-2.34 โดนการเคลื่อนลงตามความชันนี้คำนวนหาค่าต้นทุนซ้ำๆ ตามกราฟของฟังก์ชันต้นทุน (Cost Function) ที่มีลักษณะเป็นเส้นเว้าโค้งดังแสดงในรูปที่ 2.12 การที่จะทำให้โมเดลสามารถจำแยกข้อมูลได้อย่างมีประสิทธิภาพนั้น โมเดลจะเริ่มจากการสุ่มค่าน้ำหนักซึ่งส่งผลให้ฟังก์ชันต้นทุนมีค่าสูง จากนั้นโมเดลจะทำการปรับเปลี่ยนค่าน้ำหนักไปเรื่อยๆ โดยการทำอนุพันธ์มีเป้าหมายเพื่อทำให้ผลลัพธ์ของสมการเข้าใกล้จุดต่ำสุดมากที่สุด (ผลลัพธ์ของการอนุพันธ์เป็นศูนย์) จนทำให้ค่าของฟังก์ชันต้นทุนมีค่าน้อยที่สุดเท่าที่จะเป็นไปได้ โดยในการปรับเปลี่ยนค่าน้ำหนักในแต่ละครั้งจะอ้างอิงจากสมการที่ (2.26)



รูปที่ 2.11 การปรับเปลี่ยนค่าน้ำหนักด้วยวิธีการเคลื่อนลงตามความชัน

ที่มา Saugat 2018

$$w_{i+1} = w_i - \alpha \frac{\partial}{\partial w_i} loss \qquad (2.26)$$

เมื่อ w_{i+1} คือ ค่าน้ำหนักใหม่

w_i คือ ค่าน้ำหนักเดิม

α คือ อัตราการเรียนรู้

ด้วยวิธีการนี้จะทำให้ได้ค่าน้ำหนักของแต่ละคำศัพท์ออกมาเพื่อใช้ในการทำนายความรู้สึกเชิงบวก และเชิงลบของแต่ละบทวิจารณ์

2.3.7 การแก้ปัญหาประเภทคำตอบไม่เท่ากัน (Class Balancing)

จากการนำบทวิจารณ์ของนักท่องเที่ยวมาทำการสร้างโมเดลเพื่อทำการวิเคราะห์ความรู้สึกในหัวข้อที่ 2.3 พบว่าข้อมูลเป็นบทวิจารณ์ที่เป็นความรู้สึกเชิงบวกจำนวน 2,942 บทวิจารณ์ และบทวิจารณ์ที่เป็นความรู้สึกเชิงลบจำนวน 1,050 บทวิจารณ์ ซึ่งทำให้การเรียนรู้ของโมเดลนั้นเกิดปัญหาคำตอบไม่เท่ากัน (Imbalanced Data)

ในงานวิจัยฉบับนี้ได้มีการแก้ปัญหาประเภทคำตอบไม่เท่ากันด้วยวิธี SMOTE (Synthetic Minority Over-sampling TEchnique) ซึ่งเป็นวิธีการสุ่มตัวอย่างซ้ำสำหรับขุดข้อมูลที่มีคำตอบไม่เท่ากัน (Chawla et al. 2002) ซึ่งวิธี SMOTE เป็นวิธีที่ใช้กันอย่างแพร่หลายในด้านการทำเหมืองข้อความเพื่อให้จำนวนคำตอบของข้อความ หรือบทวิจารณ์เชิงบวก และเชิงลบมีจำนวนเท่ากันของ (Liu and Zhang 2012)

2.4 การเปรียบเทียบประสิทธิภาพการทำนาย

2.4.1 เมทริกซ์ความสับสน (Confusion Matrix)

เมทริกซ์ความสับสนเป็นตารางที่ใช้สำหรับวัดผลโมเดลการจำแนกความรู้สึกที่เป็นเชิงบวกและเชิงลบ ซึ่งการเลือกผลทำนายจะทำได้โดยการวัดประสิทธิภาพของการทำนายด้วยตัวชี้วัดหลากหลายชนิดซึ่งสามารถคำนวณได้จากเมทริกซ์ความสับสนดังตารางที่ 2.2

ตารางที่ 2.2 เมทริกซ์ความสับสน

ผลทำนาย / ค่าจริง	ผลบวก	ผลลบ
ผลบวก	TP	FN
ผลลบ	FP	TN

เมื่อ TP คือ ผลทำนายเป็นเชิงบวกและข้อมูลเป็นเชิงบวก (ผลบวกจริง) หรือเรียกว่า True Positive

FP คือ ผลทำนายเชิงบวกแต่ข้อมูลเป็นเชิงลบ (ผลบวกลวง) หรือเรียกว่า False Positive

FN คือ ผลทำนายเชิงลบแต่ข้อมูลเป็นบวก (ผลลบลวง) หรือเรียกว่า False Negative

TN คือ ผลทำนายเป็นเชิงลบและข้อมูลเป็นเชิงลบ (ผลลบจริง) หรือเรียกว่า True Negative

จากผลบวก และผลลบในตารางที่ 2.2 ในงานวิจัยนี้ใช้เทคนิคการทำเหมืองข้อความโดยจำแนกบทวิจารณ์โดยให้ผลบวกเป็นความรู้สึกเชิงบวก และผลลบเป็นความรู้สึกเชิงลบ และในส่วนของตัวชี้วัด งานวิจัยนี้ได้ใช้ตัวชี้วัด 3 ชนิดได้แก่ ค่าความถูกต้อง (Accuracy) ความระลึก (Precision) และความแม่นยำ (Recall) และ ซึ่งสามารถคำนวณได้ตามสมการที่ (2.27) ถึง (2.29)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.27}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.28}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.29}$$

จากตารางที่ 2.2 เมื่อให้โมเดลทำนายประโยค “I love yaowarat” ซึ่งเป็นประโยคที่เป็นความรู้สึกเชิงบวก พบว่าเมื่อโมเดลทำนายได้ถูกต้องจะเรียกว่า “ผลบวกจริง (TP)” ซึ่งถ้าโมเดลทายผิดเราจะเรียกว่า “ผลลบลวง (FN)” ในทางตรงกันข้ามในส่วนของประโยค “I hate yaowarat” เป็นประโยคที่เป็นความรู้สึกเชิงลบเมื่อโมเดลทำนายได้ถูกต้องเราจะเรียกว่า “ผลลบจริง (TN)” ซึ่งตรงกันข้ามกับ “ผลบวกลวง (FP)” โดยค่าจริงเป็นคะแนนจากนักท่องเที่ยว ซึ่งแสดงได้ดังตารางที่ 2.3

ตารางที่ 2.3 ตัวอย่างการหาผลบวกจริง (TP) ผลลบจริง (TN) ผลบวกลวง (FP) และผลลบลวง (FN)

ตัวอย่างบทวิจารณ์	**ค่าจริง**	**ผลทำนาย**	
		ผลบวก	**ผลลบ**
“I love yaowarat”	**ผลบวก**	TP	FN
“I hate yaowarat”	**ผลลบ**	FP	TN

2.4.2 การเลือกโมเดลที่ดีที่สุด (Model Selection)

การเลือกโมเดลที่ดีที่สุดคือการเลือกโมเดลย่อยๆ จากโมเดลหลักทั้ง 3 ชนิดได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอรต์เวกเตอร์แมชชีน มาใช้ในการตัดสินใจร่วมกันแบบเสียงข้างมาก โดยการเลือกโมเดลที่จะมาทำการตัดสินใจร่วมกันนั้นจะเลือกจากการพิจารณาค่าความถูกต้องที่สูงที่สุดของแต่ละโมเดลย่อย ซึ่งการนำโมเดลย่อยมาช่วยกันตัดสินใจจะทำให้สามารถเพิ่มประสิทธิภาพของการตัดสินใจของโมเดลได้ดีขึ้น (Salini, A, U Jeyapriya 2018)

2.5 การวิเคราะห์ความเด่นและความแพร่หลาย (Salience Valence Analysis)

การวิเคราะห์ความเด่น (Salience) และความแพร่หลาย (Valence) คือแนวคิดที่ใช้ในการวิเคราะห์หาจุดเด่น และจุดด้อยของกลุ่มความสนใจ โดยในงานวิจัยของ Taecharungroj และ Mathayomchan (2019) ได้นำวิธีการนี้มาประยุกต์ใช้ในเชิงของการท่องเที่ยวเพื่อใช้สำหรับตีความกลุ่มความสนใจและ คำศัพท์ที่ปรากฏในกลุ่มความสนใจนั้นๆ เพื่อวิเคราะห์หาว่าจุดเด่น จุดด้อยของแต่ละกลุ่มความสนใจ เพื่อช่วยในการสร้างนโยบายเพื่อปรับปรุงเยาวราชให้สอดคล้องกับความสนใจของนักท่องเที่ยวได้ โดยการวิเคราะห์แบ่งออกเป็น 2 ระดับได้แก่ การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม และการวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์

2.5.1 การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม (Dimensional Salience Valence Analysis)

การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่มคือ การเปรียบเทียบความพึงพอใจของนักท่องเที่ยวในแต่ละกลุ่มต่างๆ ซึ่งประกอบด้วย 2 ส่วนได้แก่

ความเด่นเชิงกลุ่ม (Dimensional Salience) คือค่าสัดส่วนจำนวนของบทวิจารณ์ที่นักท่องเที่ยววิจารณ์ไว้ในแต่ละกลุ่มสนใจเทียบกับจำนวนบทวิจารณ์ทั้งหมดทุกกลุ่มความสนใจ

ความแพร่หลายเชิงกลุ่ม (Dimensional Valence) คือค่าที่บ่งบอกถึงความรู้สึกของนักท่องเที่ยวในกลุ่มนั้นๆ ซึ่งสามารถคำนวณได้จากสมการที่ (2.30) และ (2.31)

$$Dimensional\ salience = \left(\frac{r_{POS}}{R} \right) \times 100 \tag{2.30}$$

$$Dimensional\ valence = \left(\frac{r_{POS} - e_{POS}}{r} \right) \times 100 \tag{2.31}$$

$$e_{POS} = \frac{r \times R_{POS}}{R} \tag{2.32}$$

เมื่อ r_{POS} คือ จำนวนบทวิจารณ์เชิงบวกในกลุ่มความสนใจใดๆ

r คือ จำนวนบทวิจารณ์ทั้งหมดในกลุ่มความสนใจใดๆ

e_{POS} คือ จำนวนบทวิจารณ์เชิงบวกคาดหวังในกลุ่มความสนใจใดๆ

R คือ จำนวนบทวิจารณ์ทั้งหมด

R_{POS} คือ จำนวนบทวิจารณ์เชิงบวกทั้งหมด

2.5.2 การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์ (Lexical Salience Valence Analysis)

การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์เป็นการนำผลลัพธ์ของโมเดลการจัดสรรหัวข้อแฝงมาแปลงให้อยู่ในรูปของความเด่นเชิงคำศัพท์ (Term Salience) และค่าความแพร่หลายเชิงคำศัพท์ (Term Valence) ซึ่ง 2 ค่านี้สามารถนำมาใช้ในการบ่งบอกจุดเด่น และจุดด้อยของสถานที่ท่องเที่ยวในแต่ละกลุ่มได้ โดยค่าความเด่นเชิงคำศัพท์ คือค่าความถี่ที่เกิดจากการใช้คำของนักท่องเที่ยว ยิ่งนักท่องเที่ยวมีการใช้คำศัพท์เฉพาะนี้มากเพียงใดคำศัพท์นี้ก็จะมีค่าความเด่นจะมีค่ามากขึ้นเท่านั้น ส่วนค่าความแพร่หลายเชิงคำศัพท์ คือค่าที่บ่งบอกถึงความรู้สึกของนักท่องเที่ยวที่มีต่อคำศัพท์เฉพาะ ซึ่งค่าความเด่น และความแพร่หลายสามารถคำนวณได้ตามสมการที่ (2.33) และ (2.34)

$$Term\ salience = \log_{10}(t) \tag{2.33}$$

$$Term\ valence = \frac{\bar{x}_{POS} - \bar{x}_{NEG}}{\bar{x}_{POS} + \bar{x}_{NEG}} \tag{2.34}$$

เมื่อ $\bar{x}_{POS}$ คือ จำนวนเฉลี่ยของคำศัพท์เฉพาะที่ปรากฏในบทวิจารณ์เชิงบวก

$\bar{x}_{NEG}$ คือ จำนวนเฉลี่ยของคำศัพท์เฉพาะที่ปรากฏในบทวิจารณ์เชิงลบ

t คือ จำนวนคำศัพท์เฉพาะที่เกิดขึ้น

จากการนำทั้ง 2 ปัจจัยมาสร้างตารางไขว้เพื่อทำให้ความหมายจากคำโดยค่าความเด่นและความแพร่หลายที่คำนวณออกมานั้นมีความหมายดังตารางที่ 2.4

ตารางที่ 2.4 ความหมายของค่าความเด่นและความแพร่หลาย

ความเด่น / ความแพร่หลาย	ความเด่นเป็นบวก	ความเด่นเป็นลบ
ความแพร่หลายเป็นบวก	นักท่องเที่ยวให้ความสนใจและมีความรู้สึกเชิงบวก	นักท่องเที่ยวให้ความสนใจน้อยแต่มีความรู้สึกเชิงบวก
ความแพร่หลายเป็นลบ	นักท่องเที่ยวให้ความสนใจแต่มีความรู้สึกเชิงลบ	นักท่องเที่ยวให้ความสนใจน้อยและมีความรู้สึกเชิงลบ

เมื่อทราบค่าความเด่นและความแพร่หลายของแต่ละกลุ่มความสนใจแล้ว สามารถนำค่าเหล่านี้ไปสร้างแผนภูมิฟองสบู่ของการวิเคราะห์ความเด่น และความแพร่หลายเพื่อหาจุดเด่น และจุดด้อยของสถานที่ท่องเที่ยวนั้นๆ ได้

2.5 เว็ปแอพพลิเคชั่น

เว็ปแอพพลิเคชั่นเป็นโปรแกรมคอมพิวเตอร์ประเภทหนึ่ง ที่ให้ผู้ใช้บริการสามารถใช้งานผ่านเว็บเบราว์เซอร์ต่างๆ ที่สามารถเชื่อมต่อเข้ากับระบบเซิร์ฟเวอร์ได้ โดยในปัจจุบันมีการการพัฒนาเว็บแอปพลิเคชั่นให้มีความทันสมัยมากขึ้นโดยสามารถวิเคราะห์ข้อมูลจากลูกค้าได้

ปัจจุบันด๊อกเกอร์ เป็นโปรแกรมที่ช่วยในการจำลองระบบปฏิบัติการ เพื่อใช้ในการทำงานของโปรแกรมแอพพลิเคชั่นได้อย่างทันท่วงที และมีเสถียรภาพสูงเนื่องจากสามารถเปิดใช้งานได้อย่างรวดเร็วเพื่อตอบสนองกับผู้รับบริการที่มีจำนวนมากขึ้นในปัจจุบัน โดยข้อดีของด๊อกเกอร์คือการช่วยประหยัดเวลา และความยุ่งยากในการลงโปรแกรมต่างๆ ที่จำเป็นเนื่องจากด๊อกเกอร์ได้รวบรวมโปรแกรมทุกอย่างที่จำเป็นในงานเอาไว้ในด๊อกเกอร์ทั้งหมด

ในงานวิจัยฉบับนี้ใช้ด๊อกเกอร์ เป็นโปรแกรมพื้นฐานเพื่อใช้ในการจำลองระบบเซิร์ฟเวอร์ในการประมวลผลเว็ปแอพพลิเคชั่นตัวอย่างควบคู่ไปกับการประมวลผลโมเดลการเรียนรู้ของเครื่อง

2.6 งานวิจัยที่เกี่ยวข้อง

ในงานวิจัยนี้ผู้เขียนได้รวบรวมงานวิจัยที่เกี่ยวข้องกับการใช้เทคนิคต่างๆ เพื่อวิเคราะห์บทวิจารณ์ที่เกี่ยวข้องกับการท่องเที่ยวจากเว็ปไซต์ Tripadvisor ด้วยวิธีการเรียนรู้ของเครื่องในแบบต่างๆ

ในปัจจุบันการวิเคราะห์ความรู้สึกของลูกค้าได้ถูกนำไปประยุกต์ใช้ในหลากหลายธุรกิจ โดยเฉพาะอย่างยิ่งในอุตสาหกรรมการท่องเที่ยวโดยมีเว็บไซต์ที่น่าเชื่อถืออย่าง TripAdvisor ซึ่งเป็นแพลตฟอร์มที่ให้นักท่องเที่ยวจากทั่วโลกสามารถแสดงความคิดเห็นต่างๆ โดย Tripadvisor มีระบบคัดกรองนักท่องเที่ยวที่วิจารณ์สถานที่ท่องเที่ยวเป็นระดับต่างๆ ซึ่งบ่งบอกถึงความน่าเชื่อถือของผู้วิจารณ์ทำให้ ข้อมูลที่ได้จากเว็ปไซต์ Tripadvisor มีความน่าเชื่อถือ (Filieri, Alguezaui, and McLeay 2015)

จากงานวิจัยของ Zhang et al. (2010) ได้นำข้อมูลบทวิจารณ์ร้านอาหารจาก Tripadvisor ทั้งหมด 1,242 บทวิจารณ์มาทำการเปรียบเทียบระหว่างบทวิจารณ์ของผู้เชี่ยวชาญกับบทวิจารณ์ต่อความนิยมของร้านอาหารด้วยวิธีการถดถอยเชิงเส้น เพื่อที่จะเปรียบเทียบความเห็นของลูกค้ากับความเห็นของนักวิจารณ์อาหาร โดยพบว่าบทวิจารณ์ที่มาจากลูกค้ามีความสำคัญอย่างมากต่อความนิยมของร้านอาหาร โดยเฉพาะในส่วนของรสชาติ โดยจากการทดลองพบว่าบทวิจารณ์มาจากลูกค้าได้ค่าเบต้า 0.0515 และ 0.0144 สำหรับบทวิจารณ์ของผู้เชี่ยวชาญ ซึ่งแสดงให้เห็นว่าบทวิจารณ์หรือความคิดเห็นที่มาจากตัวลูกค้าโดยตรงนั้นมีความสำคัญมากกว่าบทวิจารณ์ของผู้เชี่ยวชาญ

ในงานวิจัยของ Bi, Yang, Zhi-Ping and Jin (2019) ใช้วิธีการเรียนรู้ของเครื่องในการวิเคราะห์ปัจจัยที่ส่งผลคะแนนความนิยมของโรงแรมสองโรงแรมโดยข้อมูลที่ใช้มีจำนวน 4,276 บทวิจารณ์จาก Tripadvisor มาวิเคราะห์ด้วยโมเดลการจัดสรรข้อมูลแฝงเพื่อสกัดข้อมูลออกมาเป็นคำเฉพาะ และใช้ซัพพอร์ตเวกเตอร์แมชชีนในการวิเคราะห์ความรู้สึก โดยให้ 1-3 คะแนนเป็นความรู้สึกเชิงลบ และ 4-5 คะแนนเป็นความรู้สึกเชิงบวกเพื่อจำแนกความรู้สึก จากนั้นนำมาสร้างกราฟความสัมพันธ์ระหว่างความพึงพอใจของลูกค้า (Performance หรือ Satisfaction) และ ความสำคัญ (Importance) ของแต่ละปัจจัยออกมาจากโมเดลการจัดสรรหัวข้อแฝงซึ่งจากผลลัพธ์สามารถอธิบายได้ว่าปัจจัยที่เกี่ยวกับการตรงต่อเวลา ความพร้อม ความเป็นกันเอง ความเป็นระเบียบของการแต่งกายของพนักงานส่งผลให้ลูกค้ามีความพึงพอใจ และโรงแรมได้รับความนิยมเพิ่มขึ้น

จากงานวิจัยของ Geetha, Singha, and Sinha (2017) ได้ทำการวิเคราะห์ความรู้สึกของลูกค้าผู้เข้ามาพักในโรงแรมโดยใช้บทวิจารณ์ออนไลน์ ซึ่งทำการวิเคราะห์บทวิจารณ์ของโรงแรมทั้ง

496 โรงแรมในเมือง Gao ประเทศอินเดีย โดยข้อมูลที่ใช้เป็นการสุ่มตัวอย่าง 40 ตัวอย่างจาก 496 โรงแรม โดยงานวิจัยนี้แบ่งออกเป็น 2 ส่วนคือการวิเคราะห์การวิเคราะห์กลุ่มแบบลำดับชั้นเพื่อนำบทวิจารณ์มาแบ่งเป็นกลุ่มโดยการวิเคราะห์กลุ่มแบบลำดับชั้น (Hierarchical Clustering) เพื่อแบ่งกลุ่มข้อมูลซึ่งพบว่าบทวิจารณ์ที่นำมาใช้สามารถแบ่งได้เป็น 2 กลุ่มได้แก่ กลุ่มค่าใช้จ่าย และกลุ่มความหรูหรา จากนั้นได้ใช้นาอีฟเบย์ในการช่วยวิเคราะห์ความรู้สึกในแต่ละกลุ่มพบว่าในกลุ่มที่เกี่ยวกับค่าใช้จ่ายมีความรู้สึกเชิงบวก 55% ความรู้สึกเชิงลบ 25% และความรู้สึกเป็นกลาง 20% ส่วนกลุ่มความหรูหรา ความรู้สึกเชิงบวก 72% ความรู้สึกเชิงลบ 21% และความรู้สึกเป็นกลาง 7%

ในงานวิจัยของ Xiang et al. (2017) ได้ทำการเปรียบเทียบข้อมูลเชิงลึกของแพลตฟอร์มท่องเที่ยวยอดนิยมทั้ง 3 แพลตฟอร์มได้แก่ TripAdvisor Expedia และ Yelp โดยใช้ข้อมูล 438,890 บทวิจารณ์จาก Tripadvisor, 480,589 บทวิจารณ์จาก Expedia และ 30,816 บทวิจารณ์จาก Yelp เพื่อค้นหาข้อมูลเชิงลึกเกี่ยวกับการท่องเที่ยวโดยให้ผู้เชี่ยวชาญเป็นคนตรวจสอบว่าบทวิจารณ์เป็นบทวิจารณ์เชิงบวกหรือเชิงลบ พบว่าจากการใช้โฒเดลการจัดกรรหัวข้อแฝง ที่หาจำนวนกลุ่มที่เหมาะสด้วยวิธีเคมีน พบว่าข้อมูลสามารถแบ่งได้เป็น 6 กลุ่มคือ การบริการพื้นฐาน ความคุ้มค่า ความดึงดูดของสถานที่ ประสบการณ์ที่ได้รับ ความพึงพอใจ และใช้นาอีฟเบย์ในการทำนายความรู้สึกซึ่งได้ประสิทธิภาพจากการวัดด้วยค่าความระลึกที่ 95%

Kuhamanee et al. (2017) ได้ใช้ข้อมูลจาก Twitter 10,000 บทวิจารณ์มาวิเคราะห์ความรู้สึกของนักท่องเที่ยวต่างประเทศที่วิจารณ์เกี่ยวกับกรุงเทพในปี 2017 โดยมีประเภทคำตอบ 3 ประเภทได้แก่ความรู้สึกเชิงบวก ความรู้สึกเชิงลบ และความรู้สึกเป็นกลางซึ่งแบ่งโดยใช้คน ด้วยโมเดลการเรียนรู้ของเครื่องทั้งหมด 4 ชนิดได้แก่ ต้นไม้ตัดสินใจ นาอีฟเบย์ ซัพพอร์ตเวกเตอร์แมชชีน และโครงข่ายประสาทเทียม มาวิเคราะห์ความรู้สึกของชาวต่างชาติที่มีต่อกรุงเทพมหานคร ประเทศไทย ซึ่งได้ค่าความถูกต้องของโมเดลเป็น 79.83% 55.66% 80.11% และ 80.33% ตามลำดับ อย่างไรก็ตามงานวิจัยของ Korovkinas and Garšva (2018) การถดถอยเชิงโลจิสติคมีค่าความแม่นยำสูงกว่านาอีฟเบย์หนึ่งครั้งดังนั้นในงานวิจัยฉบับนี้จึงนำการถดถอยเชิงโลจิสติคมาใช้ในการวิเคราะห์ความรู้สึกด้วย

ในงานวิจัยของ Taecharungroj and Mathayomchan (2019) ได้นำบทวิจารณ์ของนักท่องเที่ยวที่มีต่อจังหวัดภูเก็ต ประเทศไทย บนเว็ปไซต์ Tripadvisor จำนวน 65,079 บทวิจารณ์ซึ่งประกอบด้วย 25,458 บทวิจารณ์ชายหาด 12,584 บทวิจารณ์เกาะต่างๆ 3,514 บทวิจารณ์ตลาด 1,300 ถนนคนเดิน และ 10,519 บทวิจารณ์ โดยให้ 1-3 คะแนนเป็นความรู้สึกเชิงลบ และ 4-5

คะแนนเป็นความรู้สึกเชิงบวกเพื่อจำแนกความรู้สึก ซึ่งข้อมูลจะถูกสกัดออกมาเป็นปัจจัยในรูปของคำศัพท์เฉพาะที่ใช้การจัดกลุ่มข้อมูลแบบเคมีนในการหาค่าจำนวนกลุ่มที่เหมาะสมเพื่อนำจำนวนกลุ่มที่เหมาะสมนี้ไปใช้ในโมเดล LDA และจากนั้นนำผลลัพธ์จาก LDA ไปใช้ร่วมกับผลลัพธ์ของนาอีฟเบย์ ซึ่งเป็นโมเดลในการวิเคราะห์ความรู้สึกและนำไปสร้างแผนภูมิฟองสบู่เพื่อใช้ดูข้อมูลเชิงลึกของภูเก็ต ซึ่งการใช้โมเดลการเรียนรู้ของเครื่องสำหรับงานด้านการท่องเที่ยวควรมีค่าความถูกต้องมากกว่า 70% ขึ้นไป โดยงานวิจัยฉบับนี้ใช้ค่าความถูกต้องในการวัดผลลัพธ์ของโมเดลซึ่งดีที่สุดอยู่ที่ 78%

จากการทบทวนวรรณกรรมที่เกี่ยวข้องกับการวิเคราะห์บทวิจารณ์ที่เกี่ยวกับการท่องเที่ยวด้วยวิธีการต่างๆทั้งหมด ผู้วิจัยได้มีการนำงานวิจัยและโมเดลต่างๆ มาสรุปได้ดังตารางที่ 2.5 โดยแบ่งกลุ่มโมเดลที่ใช้เป็น 2 กลุ่ม คือการเรียนรู้แบบไม่มีผู้สอน และการเรียนรู้แบบมีผู้สอน

งานวิจัยฉบับนี้ได้มีการนำโมเดล และวิธีการจากการทบทวนวรรณกรรมต่างๆ มาพัฒนาต่อยอดโดยมีการใช้พัฒนาต่อยอดในด้านของการเรียนรู้แบบมีผู้สอนโดยการเพิ่มจำนวนโมเดล การทำเรกูราไลซ์ และการแก้ปัญหาคำตอบไม่เท่ากัน ซึ่งเป็นวิธีที่แตกต่างจากงานวิจัยข้างต้น เพื่อให้ผลลัพธ์มีความเที่ยงตรง และเพื่อลดการเกิด Overfitting ได้ดียิ่งขึ้นอีกทั้งมีการนำโมเดลมามาใช้สร้างตัวอย่างเพื่อให้เห็นเป็นแนวทางในการนำปัญญาประดิษฐ์มาใช้จริงในรูปของเว็ปแอพพลิเคชั่น

ตารางที่ 2.5 งานวิจัยที่เกี่ยวข้อง

ชื่อผู้แต่ง	ชื่อเรื่อง	เทคนิคที่ใช้								
		การเรียนรู้แบบไม่มีผู้สอน			การเรียนรู้แบบมีผู้สอน					
		HC	LDA	KM	LiREG	NB	SVM	DT	NN	LoREG
Zhang et al.	The impact of e-word-of-mouth on the online popularity of restaurants: A comparison of consumer reviews and editor reviews				✓					
YuvinaTileng, Herry Utomo, and Latuperissa	Analysis of Service Quality using Servqual Method and Importance Performance Analysis (IPA) in Population Department, Tomohon City		✓				✓			
Guo, Barnes, and Jia	Mining meaning from online ratings and reviews: Tourist satisfaction analysis using latent dirichlet allocation		✓							

ชื่อผู้แต่ง	ชื่อเรื่อง	เทคนิคที่ใช้								
		การเรียนรู้แบบไม่มีผู้สอน			การเรียนรู้แบบมีผู้สอน					
		HC	LDA	KM	LiREG	NB	SVM	DT	NN	LoREG
Korovkinas and Garšva	Selection of Intelligent Algorithms for Sentiment Classification Method Creation					✓				✓
Kuhamanee et al.	Sentiment analysis of foreign tourists to Bangkok using data mining through online social network					✓	✓	✓	✓	
Taecharungroj, Viriya, and Boonyanit Mathayomchan	Analysing TripAdvisor reviews of tourist attractions in Phuket, Thailand		✓			✓				
Xiang et al.	A comparative analysis of major online review platforms: Implications for social media analytics in hospitality and tourism		✓	✓		✓				

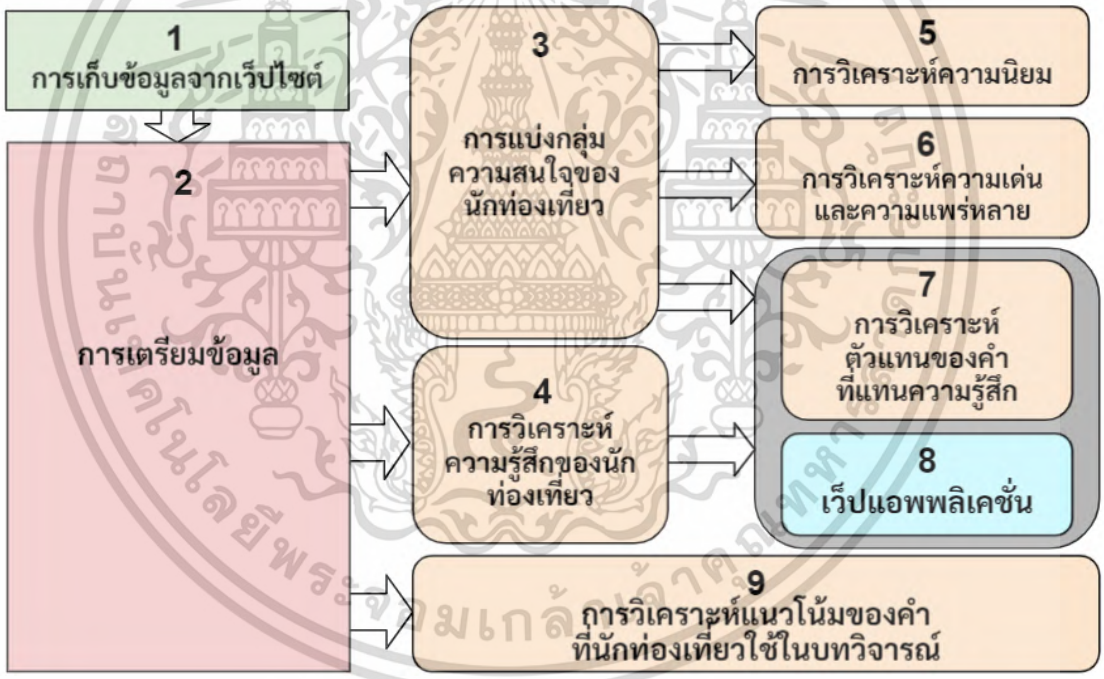
ชื่อผู้แต่ง	ชื่อเรื่อง	เทคนิคที่ใช้								
		การเรียนรู้แบบไม่มีผู้สอน			การเรียนรู้แบบมีผู้สอน					
		HC	LDA	KM	LiREG	NB	SVM	DT	NN	LoREG
Geetha, Singha, and Sinha	Relationship between customer sentiment and online customer ratings for hotels - An empirical analysis	✓				✓				

เมื่อ HC คือ การวิเคราะห์กลุ่มแบบลำดับชั้น (Hierarchical Clustering)
LDA คือ การจัดสรรหัวข้อแฝง (Latent Dirichlet Allocation)
KM คือ การจัดกลุ่มแบบเคมีน (K-means Clustering)
} การเรียนรู้แบบไม่มีผู้สอน

LiREG คือ การถดถอยเชิงเส้น (Linear Regression)
NB คือ นาอีฟเบย์ (Naïve Bayes)
SVM คือ ซัพพอร์ดเวกเตอร์แมชชีน (Support Vector Machine)
DT คือ ต้นไม้ตัดสินใจ (Decision Tree)
NN คือ โครงข่ายประสาทเทียม (Neural Network)
LoREG คือ การถดถอยเชิงโลจิสติค (Logistic Regression)
} การเรียนรู้แบบมีผู้สอน

บทที่ 3
วิธีการดำเนินงานวิจัย

ในงานวิจัยฉบับนี้ได้กำหนดระเบียบวิธีวิจัยได้พัฒนาแนวทางการกำหนดระเบียบวิธีวิจัยจาก Taecharungroj and Mathayomchan (2019) ซึ่งเป็นการวิเคราะห์ความสนใจและความรู้สึกของนักท่องเที่ยวที่เดินทางมายังจังหวัดภูเก็ต ประเทศไทย ที่พัฒนาด้วยโปรแกรม KNIME โดยผู้วิจัยได้นำมาปรับปรุงระบบการวิเคราะห์ให้มีความสมบูรณ์มากขึ้นโดยมีการแก้ปัญหาข้อมูลไม่เท่ากัน (Imbalanced Data) และเพิ่มเติมโมเดลสำหรับการวิเคราะห์ความรู้สึก โดยระเบียบวิจัยประกอบไปด้วย 9 ส่วนดังแสดงในรูปที่ 3.1 แบ่งออกเป็น 4 ส่วนหลักได้แก่ สีเขียวเป็นการเก็บรวมรวมข้อมูล สีแดงเป็นการเตรียมข้อมูล สีส้มเป็นการวิเคราะห์ข้อมูลและการประเมินผลลัพธ์ และสีฟ้าเป็นส่วนของการพัฒนาเว็ปแอพพลิเคชั่น

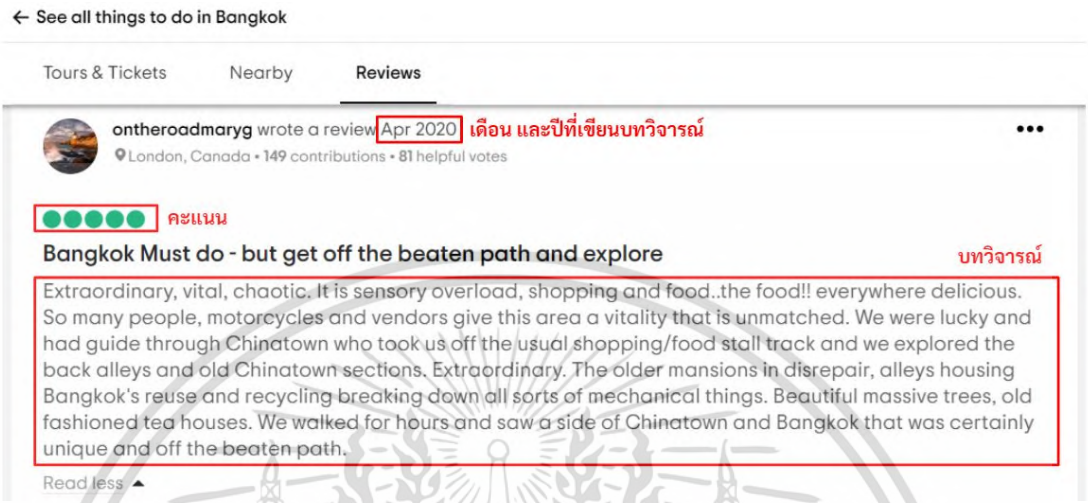


รูปที่ 3.1 โครงสร้างของงานวิจัย

3.1 การเก็บข้อมูลจากเว็ปไซต์

ในการเก็บข้อมูลจากเว็ปไซต์ Tripadvisor ในงานวิจัยฉบับนี้ได้ใช้ภาษาไพธอนเป็นเครื่องมือในการเก็บข้อมูลโดยอัตโนมัติ โดยไลบรารี่ที่สำคัญสำหรับการเก็บข้อมูลผ่านเว็ปได้แก่ Beautifulsoup4 และ Selenium ซึ่งเป็นไลบรารี่ที่ช่วยในการค้นหาข้อมูลส่วนที่เราต้องการในหน้าเว็ปไซต์ต่าง โดยไลบรารี่ตัวนี้จะทำการค้นหาข้อความส่วนที่เราต้องการจากโครงสร้างของ

หน้าเว็ปไซต์ที่เราต้องการ ซึ่งในงานวิจัยฉบับนี้จะทำการเก็บข้อมูล 4 อย่างได้แก่ เดือนที่เขียนบทวิจารย์ ปีที่เขียนบทวิจารณ์ คะแนน และเนื้อหาบทวิจารณ์ดังรูปที่ 3.2 โดยหลังจากเก็บข้อมูลเสร็จแล้วข้อมูลจะถูกบันทึกเป็นตารางในรูปของไฟล์ตารางเอกเซลดังรูปที่ 3.3



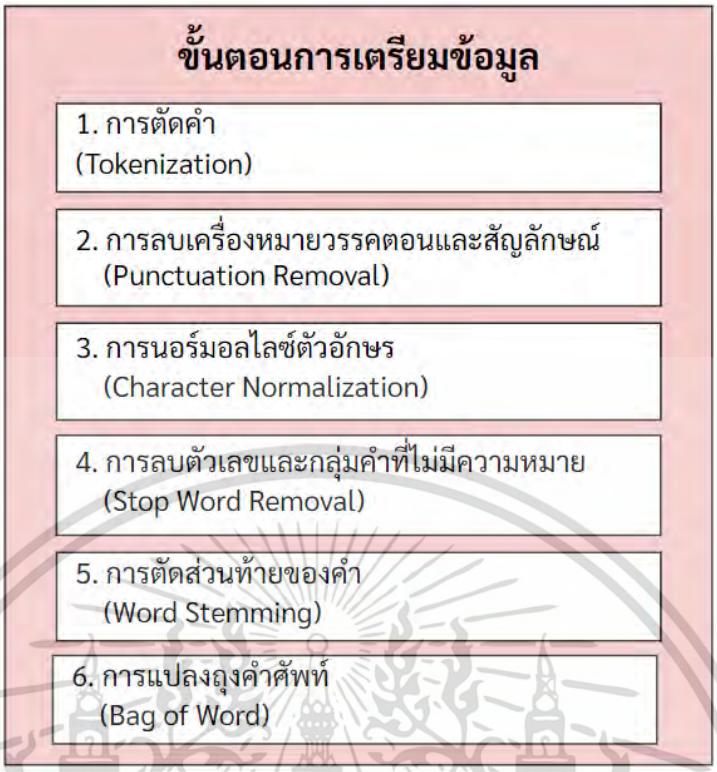
รูปที่ 3.2 ตัวอย่างบทวิจารณ์

เดือนที่เขียน	ปีที่เขียน	คะแนน	บทวิจารณ์
Apr	2020	5	Extraordinary, vital, chaotic. It is
Mar	2020	5	Bangkok Chinatown is very easy t
Mar	2020	4	I visited so many times here. Foo
Mar	2020	4	Catch a BTS to Saphan Taksin stat
Mar	2020	5	Chinatown, is an important landr
Mar	2020	4	Good food and entertainment... t
Mar	2020	4	In some cities, the area doesn't q
Mar	2020	5	We went to Chinatown in a Sund
Mar	2020	4	Wandered around Chinatown for
Mar	2020	5	It's a have to see it to believe it k
Feb	2020	4	Even if you don't want to eat or s

รูปที่ 3.3 ไฟล์เอกเซลที่เก็บข้อมูล

3.2 การเตรียมข้อมูล

หลังจากทำการเก็บข้อมูลจากเว็ปไซต์เรียบร้อยแล้วจากหัวข้อที่ 3.1 ข้อมูลรูปแบบของข้อความนั้นไม่สามารถนำไปใช้ในการวิเคราะห์ได้ทันที จึงต้องมีการนำข้อความทั้งบทวิจารณ์มาเข้าสู่ขั้นตอนการแปลงข้อมูลต่างๆ ให้มีรูปแบบเป็นเวกเตอร์โดยมีกระบวนการแบ่งเป็น 6 ขั้นตอนแสดงดังรูปที่ 3.4



รูปที่ 3.4 ขั้นตอนการเตรียมข้อมูล

ที่มา Sarkar 2019

ตัวอย่างการเตรียมข้อมูล

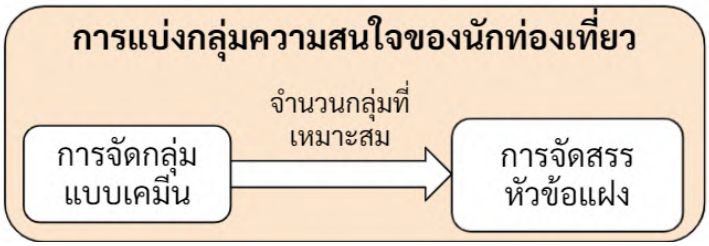
ในตัวอย่างการเตรียมข้อมูลนี้ ประโยคที่ใช้คือประโยค “Bangkok Chinatown is very easy to get around !” ซึ่งจะต้องผ่านขั้นตอนการเตรียมข้อมูลทั้ง 6 ขั้นตอน ได้แก่ การตัดคำ การลบเครื่องหมายวรรคตอนและสัญลักษณ์ออก การนอร์มอลไลซ์ตัวอักษร ลบตัวเลขและกลุ่มคำที่ไม่มีความหมายออก การตัดส่วนท้ายของคำ และการแปลงถุงคำศัพท์ ซึ่งจากประโยคตัวอย่างดังกล่าวเมื่อนำมาผ่านขั้นตอนการเตรียมข้อมูลทั้ง 6 ขั้นตอน ผลลัพธ์แต่ละขั้นตอนสามารถแสดงได้ดังตารางที่ 3.1

ตารางที่ 3.1 ตัวอย่างขั้นตอนการเตรียมข้อมูล

ขั้นตอน	คำอธิบาย	ผลลัพธ์
1. การตัดคำ	นำประโยคมาแบ่งออกเป็นคำ	“Bangkok”, “Chinatown”, “is”, “very”, “easy”, “to”, “get”, “around”, “!”
2. การลบเครื่องหมายวรรคตอนและสัญลักษณ์ออก	ลบเครื่องหมาย “!” ออก	“Bangkok”, “Chinatown”, “is”, “very”, “easy”, “to”, “get”, “around”
3. การนอร์มอลไลซ์ตัวอักษร	ทำให้เป็นตัวอักษรเล็กทั้งหมด	“bangkok”, “chinatown”, “is”, “very”, “easy”, “to”, “get”, “around”,
4. การลบตัวเลขและกลุ่มคำที่ไม่มีความหมายออก	ลบ “is” กับ “to” ที่ไม่มีความหมายออก	“bangkok”, “chinatown”, “easy”, “get”, “around”
5. การตัดส่วนท้ายของคำ	การทำให้คำศัพท์อยู่ในรูปปกติโดยในตัวอย่างนี้คือการทำให้ “getting” เป็น “get”	“bangkok”, “chinatown”, “easy”, “get”, “around”
6. การแปลงถุงคำศัพท์	แปลงให้เป็นเวกเตอร์ซึ่งเป็นการนับความถี่ที่เกิดขึ้นของคำ	[1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, ...]

3.3 การแบ่งกลุ่มความสนใจของนักท่องเที่ยว

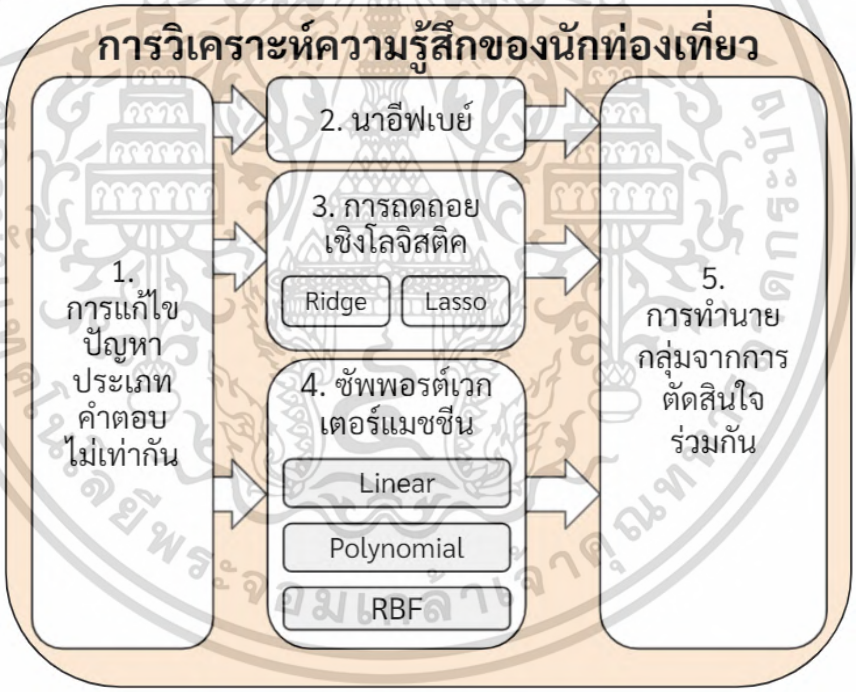
หลังจากที่ทำการแปลงบทวิจารณ์ทั้งหมดเป็นเวกเตอร์ด้วยภาษาไพธอนแล้ว ข้อมูลเวกเตอร์เหล่านี้จะถูกนำมาเข้ากระบวนการแบ่งกลุ่มแบบเคมีนโดยกำหนดจำนวนกลุ่มที่เหมาะสมสำหรับข้อมูลชุดนั้นด้วยวิธีการข้อศอก (Elbow) ตามที่กล่าวไว้ในหัวข้อที่ 2.2.1 ซึ่งเมื่อได้ค่าจำนวนกลุ่มที่เหมาะสม K กลุ่มแล้วจะนำค่า K นี้ไปใช้ในการกำหนดกลุ่มความสนใจของนักท่องเที่ยวในโมเดลการจัดสรรหัวข้อแฝง (Latent Dirichlet Allocation) เพื่อหาคำศัพท์เฉพาะในแต่ละกลุ่มดังแสดงในรูปที่ 3.5



รูปที่ 3.5 ขั้นตอนการแบ่งกลุ่มนักท่องเที่ยว

3.4 การวิเคราะห์ความรู้สึกของนักท่องเที่ยว

โมเดลการเรียนรู้แบบมีผู้สอนในงานวิจัยฉบับนี้จะใช้ทั้งหมด 3 โมเดลหลักได้แก่ นาอีฟเบย์ (Naïve Bayes) การถดถอยเชิงโลจิสติค (Logistic Regression) และซัพพอร์ตเวกเตอร์แมชชีน (Support Vector Machine: SVM) โดยใช้ไลบรารี่ Sklearn ซึ่งเป็นไลบรารี่สำหรับสร้างโมเดลการเรียนรู้ของเครื่องในภาษาไพธอน ซึ่งหลังจากข้อมูลผ่านกระบวนการเตรียมข้อมูลในหัวข้อที่ 3.2 แล้ว ข้อมูลทั้งหมดจะถูกนำไปสร้างโมเดล โดยมีขั้นตอนดังรูปที่ 3.6



รูปที่ 3.6 ขั้นตอนการวิเคราะห์ความรู้สึกของนักท่องเที่ยว

3.4.1 การแก้ไขปัญหาประเภทคำตอบไม่เท่ากัน

จากปัญหาข้อมูลบทวิจารณ์ในงานวิจัยฉบับนี้มีจำนวนบทวิจารณ์เชิงลบ และเชิงบวกไม่เท่ากัน ทำให้เกิดปัญหาประเภทของคำตอบในชุดข้อมูลไม่เท่ากัน ดังนั้นในงานวิจัยฉบับนี้จะใช้วิธี SMOTE มาช่วยในการแก้ปัญหาข้อมูลที่มีคำตอบไม่เท่ากัน ในงานทางด้านการจำแนกความรู้สึก (Chawla NV and et al 2002; Sarakit P. and et al. 2015) ซึ่งในภาษาไพธอนสามารถเรียกใช้ได้จากไลบรารี่ SMOTE ได้โดยตรง

การแก้ไขปัญหาประเภทคำตอบไม่เท่ากันจะเริ่มจากการนำข้อมูลบทวิจารณ์เชิงลบมาทำการสุ่มซ้ำด้วยวิธี SMOTE เพื่อให้ได้จำนวนบทวิจารณ์เชิงลบเท่ากับจำนวนบทวิจารณ์เชิงบวก จากนั้นจะนำข้อมูลทั้งหมดมาทำการแบ่งส่วนเพื่อใช้ในการฝึกสอน และทดสอบ โดยแบ่งเป็น 70% สำหรับการฝึกสอน และ 30% สำหรับการทดสอบโมเดล

3.4.2 นาอีฟเบย์

การสร้างโมเดลนาอีฟเบย์จะใช้ไลบรารี่ Sklearn.naive_bayes ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับการสร้างโมเดลนาอีฟเบย์สำหรับงานประเภทการจำแนก (Classification) โดยหลักการ และทฤษฎีในการทำงานดำเนินการตามหัวข้อที่ 2.3.1

3.4.3 การถดถอยเชิงโลจิสติค

การสร้างการถดถอยเชิงโลจิสติคจะใช้ไลบรารี่ Sklearn.linear_model.LogisticRegression ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับโมเดลเชิงเส้น โดยการถดถอยเชิงโลจิสติคของ Sklearn จะมีการปรับค่าบทลงโทษในแบบต่างๆ โดยหลักการ และทฤษฎีในการทำงานดำเนินการตามหัวข้อที่ 2.3.2 ซึ่งในงานวิจัยนี้ได้ใช้ 2 แบบได้แก่ บทลงโทษแบบ Ridge และบทลงโทษแบบ Lasso (James et al. 2013) ซึ่งสามารถกำหนดได้จากพารามิเตอร์ Penalty ในไลบรารี่ดังกล่าว

3.4.4 ซัพพอร์ตเวกเตอร์แมชชีน

การสร้างซัพพอร์ตเวกเตอร์แมชชีนจะแบ่งเป็น 2 ส่วนได้แก่ ซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น และซัพพอร์ตเวกเตอร์แมชไม่เชิงเส้น โดยซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นจะใช้ไลบรารี่ Sklearn.svm.LinearSVC ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับสร้างโมเดลซัพพอร์ตเวกเตอร์แมชชีนสำหรับการจำแนก (Linear Support Vector Classifier) ส่วนซัพพอร์ตเวกเตอร์แมชไม่เชิงเส้นจะใช้ไลบรารี่ Sklearn.svm.SVC ซึ่งเป็นหนึ่งในไลบรารี่ย่อยของ Sklearn ที่ใช้สำหรับสร้างโมเดลซัพพอร์ตเวกเตอร์แมชชีนแบบไม่เชิงเส้น ซึ่งสำหรับซัพพอร์ตเวกเตอร์แมชไม่เชิงเส้นนั้นจะมีการเพิ่มเคอร์เนลที่ใช้ในการแปลงซึ่งในงานวิจัยนี้ใช้ทั้งหมด 3 แบบได้แก่ แบบเชิงเส้น แบบโพลิโนเมียล และแบบ Radial Basis Function ซึ่งสามารถกำหนดได้จากพารามิเตอร์ Kernel

3.4.5 การทำนายกลุ่มจากการตัดสินใจร่วมกัน

หลังจากทำการสร้างโมเดลในแต่ละโมเดลเสร็จ แต่ละโมเดลจะนำมาถูกวัดสมรรถภาพ เพื่อหาโมเดลที่ดีที่สุด 3 โมเดลเพื่อใช้ในการสร้างโมเดลสำหรับการตัดสินใจร่วมกันแบบเสียงข้างมาก ซึ่งในงานวิจัยฉบับนี้จะเป็นการนำโมเดลที่ดีที่สุดของ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอรต์เวกเตอร์แมชชีน มาทำการตัดสินใจร่วมกัน

3.5 การวิเคราะห์ความนิยม (Popularity Analysis)

เมื่อทำการแบ่งกลุ่มบทวิจารณ์ตามกลุ่มความสนใจแล้ว บทวิจารณ์ในแต่ละกลุ่มนั้นจะถูกนำมาวิเคราะห์ความนิยมซึ่งเป็นการวิเคราะห์ค่าคะแนนเฉลี่ยของบทวิจารณ์ในกลุ่มโดยคะแนนเฉลี่ยเป็นคะแนนจริงจากข้อมูลที่เก็บมาจากเว็ปไซต์ Tripadvisor ซึ่งมีตั้งแต่ 1 (ความพอใจต่ำสุด) ถึง 5 (ความพอใจสูงที่สุด) ดังนั้นการนำบทวิจารณ์ที่อยู่ในกลุ่มความสนใจต่างๆ มาหาค่าเฉลี่ยเพื่อใช้ในการดูคะแนนภาพรวมของแต่ละกลุ่มความสนใจจะสามารถบ่งบอกถึงค่าความนิยมของของนักท่องเที่ยวที่มีต่อกลุ่มความสนใจได้ในรูปของคะแนนเฉลี่ย

3.6 การวิเคราะห์ความเด่นและความแพร่หลาย (Salience-Valence Analysis)

3.6.1 การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม

การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่มเป็นการนำจำนวนบทวิจารณ์ที่ถูกแบ่งกลุ่มความสนใจในแต่ละกลุ่มความสนใจเพื่อมาหาค่าความเด่นเชิงกลุ่ม (Dimensional Salience) และความแพร่หลายเชิงกลุ่ม (Dimensional Salience) ซึ่งนำมาใช้ในการเปรียบเทียบ และตีความความพึงพอใจของนักท่องเที่ยวในแต่ละกลุ่มความสนใจต่างๆ

ความเด่นเชิงกลุ่ม คือการอธิบายสัดส่วนความสนใจของนักท่องเที่ยวในแต่ละกลุ่มเทียบกับความสนใจทั้งหมด โดยนำจำนวนบทวิจารณ์ในแต่ละกลุ่มเทียบกับจำนวนบทวิจารณ์ทั้งหมด

ความแพร่หลายเชิงกลุ่มจะใช้ในการบ่งบอกความรู้สึกของนักท่องเที่ยวในกลุ่มความสนใจต่างๆ เพื่อใช้ในการอธิบายความพึงพอใจโดยรวมที่นักท่องเที่ยวมีต่อกลุ่มความสนใจใดๆ

ตัวอย่างการคำนวณความเด่นและความแพร่หลายเชิงกลุ่ม

ในตัวอย่างการคำนวณความเด่นและความแพร่หลายเชิงกลุ่มนั้นจะใช้ข้อมูลตัวอย่างจากตารางที่ 3.2 ซึ่งมีข้อมูลของจำนวนบทวิจารณ์ทั้งหมด และจำนวนบทวิจารณ์เชิงบวกในกลุ่มความสนใจอาหาร

ตารางที่ 3.2 ตัวอย่างข้อมูลสำหรับการคำนวณความเด่นและความแพร่หลายเชิงกลุ่ม

กลุ่มความสนใจ	จำนวนบทวิจารณ์เชิงบวก	จำนวนบทวิจารณ์เชิงลบ	จำนวนบทวิจารณ์ทั้งหมด
อาหาร	1,769	656	2,425
อื่นๆ	1,133	434	1,567
รวม	**2,902**	**1,090**	**3,992**

จากข้อมูลในตารางที่ 3.2 สามารถคำนวณค่าความเด่นได้โดยใช้สมการที่ (3.1)

$$Dimensional\ salience = \left(\frac{r_{POS}}{R} \right) \times 100 \tag{3.1}$$

ซึ่งเมื่อแทนค่าลงไปจะได้ว่า

$$Dimensional\ salience_{food} = \left(\frac{1,769}{3,992} \right) \times 100 = 60.75 \tag{3.2}$$

สำหรับค่าความแพร่หลายเชิงกลุ่มเราสามารถคำนวณได้จากสมการที่ (3.3)

$$Dimensional\ valence = \left(\frac{r_{POS} - e_{POS}}{r} \right) \times 100 \tag{3.3}$$

และค่าความคาดหวังคำนวณได้จากสมการที่ (3.4)

$$e_{POS} = \frac{r \times R_{POS}}{R} \tag{3.4}$$

ทำให้สามารถคำนวณจำนวนบทวิจารณ์เชิงบวกคาดหวัง e_{POS} ได้ดังนี้

$$e_{POS} = \frac{2,425 \times 2,902}{3,922} = 1,763 \tag{3.5}$$

ซึ่งเมื่อได้จำนวนบทวิจารณ์เชิงบวกคาดหวังแล้วทำให้สามารถคำนวณความแพร่หลายเชิงกลุ่มได้ดังนี้

$$Dimensional\ valence = \left(\frac{1,769 - 1,763}{2,425} \right) \times 100 = 0.2531 \tag{3.6}$$

โดยการที่ค่าความแพร่หลายเป็นบวกนั้นสามารถตีความได้ว่านักท่องเที่ยวมีความรู้สึกเชิงบวกกับกลุ่มความสนใจอาหารนี้

3.6.2 การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์

การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์เป็นการนำคำศัพท์เฉพาะทั้ง 10 คำของทั้ง 4 กลุ่มความสนใจมาคำนวณหาค่าความเด่นและความแพร่หลายตามสมการที่กล่าวไว้ในบทที่แล้วดังสมการที่ (2.33) และ (2.34) เพื่อบ่งบอกเอกลักษณ์ของสถานที่ท่องเที่ยว

ค่าความเด่นเชิงคำศัพท์ (Term Salience) คือค่าความถี่ที่เกิดจากการใช้คำของนักท่องเที่ยว ยิ่งนักท่องเที่ยวมีการใช้คำศัพท์เฉพาะนี้มากเพียงใดคำศัพท์นี้ก็จะมีค่าความเด่นจะมีค่ามากขึ้นเท่านั้น

ส่วนค่าความแพร่หลายเชิงคำศัพท์ (Term Valence) เป็นค่าที่บ่งบอกถึงความรู้สึกของนักท่องเที่ยวที่มีต่อคำศัพท์เฉพาะ ซึ่งสามารถตีความได้ว่าโดยเฉลี่ยแล้วคำศัพท์เฉพาะเหล่านี้ถูกนักท่องเที่ยวใช้ในการเขียนบทวิจารณ์ในเชิงบวกหรือเชิงลบมากน้อยเท่าใด

ตัวอย่างการคำนวณความเด่นและความแพร่หลาย

กำหนดให้คำว่า “food” และ “market” เป็นคำศัพท์เฉพาะที่จะหาค่าความเด่นและความแพร่หลายเชิงคำศัพท์ โดยกำหนดตัวอย่างบทวิจารณ์ดังตารางที่ 3.3

ตารางที่ 3.3 บทวิจารณ์ตัวอย่างสำหรับการคำนวณความเด่นและความแพร่หลายเชิงคำศัพท์

บทวิจารณ์	ความรู้สึก	กลุ่มความสนใจ
The food from yaowarat road is the best.	POS	อาหาร
There are a lot of stinky foods.	NEG	อาหาร
Great night market and cheap food. A lot of people in most of the time of the day.	POS	อาหาร

ในตารางที่ 3.3 เป็นตัวอย่างของบทวิจารณ์ที่ถูกแบ่งให้ไปอยู่ในกลุ่มความสนใจทั้ง 4 กลุ่ม โดยในตัวอย่างนี้จะเป็นประโยคตัวอย่างที่อยู่ในกลุ่มของความสนใจอาหาร ซึ่งมีค่าความรู้สึกเกิดมาจากการแปลงค่าคะแนน 1-3 คะแนนเป็นความรู้สึกเชิงลบ (NEG) และ 4-5 คะแนนความรู้สึกเชิงบวก ดังนั้นจากสมการ

$$Term\ salience = \log_{10}(t) \tag{3.67}$$

ซึ่ง t คือค่าความถี่ของคำศัพท์เฉพาะที่เกิดขึ้นซึ่งจากตารางที่ 3.3 พบว่า มีคำว่า food อยู่ 3 คำและมีคำว่า market 1 คำดังนั้น

$$Term\ salience_{food} = \log_{10}(3) = 0.47 \tag{3.7}$$

$$Term\ salience_{market} = \log_{10}(1) = 0 \tag{3.8}$$

สำหรับค่าความแพร่หลายเชิงคำศัพท์เราสามารถคำนวณได้จากสมการที่ (3.9)

$$Term\ valence = \frac{\bar{x}_{POS} - \bar{x}_{NEG}}{\bar{x}_{POS} + \bar{x}_{NEG}} \tag{3.8}$$

โดย $\bar{x}_{POS}$ คือ จำนวนเฉลี่ยของคำศัพท์เฉพาะที่ปรากฏในบทวิจารณ์ที่มีรู้สึกเชิงบวก และ $\bar{x}_{NEG}$ คือ จำนวนเฉลี่ยของคำศัพท์เฉพาะที่ปรากฏในบทวิจารณ์ที่รู้สึกเชิงลบดังนั้นเราจะสามารถคำนวณค่าความแพร่หลายของคำศัพท์ food และ market ได้ซึ่งเราพบคำว่า food อยู่ในบทวิจารณ์ทความรู้สึกเชิงลบ 1 ครั้งจากบทวิจารณ์ความรู้สึกเชิงลบ 1 บทวิจารณ์ ดังนั้น $\bar{x}_{NEG}$ จะเท่ากับ $\frac{1}{1}$ ในทางตรงกันข้ามคำว่า food อยู่ในบทวิจารณ์เชิงบวกทั้งหมด 2 ครั้งจากบทวิจารณ์เชิงบวกทั้งหมด 2 บทวิจารณ์ดังนั้น $\bar{x}_{POS}$ จะเท่ากับ $\frac{2}{2}$ ทำให้สามารถคำนวณค่าความแพร่หลายได้เป็น

$$Valence_{food} = \frac{\frac{2}{2} - \frac{1}{1}}{\frac{2}{2} + \frac{1}{1}} = 0 \quad (3.9)$$

สำหรับคำศัพท์คำว่า market ในบทวิจารณ์เชิงลบทั้งหมด 1 บทวิจารณ์เราไม่พบคำว่า market ดังนั้น $\bar{x}_{NEG}$ จะเท่ากับ $\frac{0}{1}$ ในทางตรงกันข้ามในบทวิจารณ์เชิงบวกเราพบคำนี้อยู่ 1 ครั้งจากทั้งหมด 1 บทวิจารณ์เชิงบวกทำให้ $\bar{x}_{POS}$ จะเท่ากับ $\frac{1}{1}$ ดังนั้นค่าความแพร่หลายจะเท่ากับ

$$Valence_{market} = \frac{\frac{1}{1} - \frac{0}{1}}{\frac{1}{1} + \frac{2}{2}} = 0.5 \quad (3.10)$$

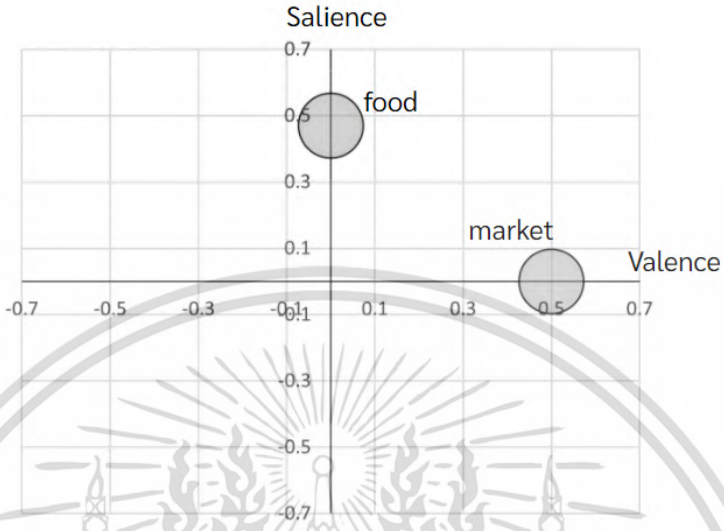
จากนั้นทำให้เราสามารถนำค่าความเด่นและความแพร่หลายทั้งหมดมาสร้างเป็นตารางได้ดังตารางที่ 3.4

ตารางที่ 3.4 ตัวอย่างค่าความเด่นและความแพร่หลายเชิงคำศัพท์

กลุ่มแหล่งอาหาร		
คำศัพท์	**ความเด่น**	**ความแพร่หลาย**
Food	0.47	0
Market	0	0.5

จากตารางที่ 3.4 เราพบเราสามารถนำค่าความเด่นและความแพร่หลายเชิงคำศัพท์ไปสร้างเป็นความสัมพันธ์ของทั้ง 2 ค่าได้ในรูปแผนภูมิฟองสบู่ (Bubble Chart) ดังรูปที่ 3.5 ซึ่งจากรูปจะ

พบว่า “food” มีค่าความเด่นมากกว่า “market” ซึ่งหมายความว่า “food” เป็นคำที่เกิดขึ้นบ่อยกว่า “market” ในบทวิจารณ์ ในทางกลับกัน “market” มีความแพร่หลายมากกว่า “food” ซึ่งบ่งบอกถึงนักท่องเที่ยวมีความรู้สึกดีกับ “market” มากกว่าดังรูปที่ 3.7



รูปที่ 3.7 ตัวอย่างแผนภูมิฟองสบู่

3.7 การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก

การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึกเป็นการนำบทวิจารณ์มาแบ่งเป็นประโยคและนำไปวิเคราะห์ด้วยโมเดลวิเคราะห์ความรู้สึกที่มีการตัดสินใจร่วมกันด้วยเสียงข้างมากเพื่อจำแนกความรู้สึก โดยผลลัพธ์จะออกมาเป็นคำศัพท์ต่างๆ ที่ปรากฏอยู่ในความรู้สึกเชิงบวก และความรู้สึกเชิงลบ

หลังจากทำการหาคำศัพท์ต่างๆ ที่ปรากฏอยู่ในความรู้สึกเชิงบวกและเชิงลบแล้วจะนำคำศัพท์ที่ปรากฏเหล่านั้นมาพิจารณาว่ามีคำศัพท์ใดบ้างที่เหมาะสมที่จะนำมาเป็นข้อดีและข้อเสียในกลุ่มความสนใจใดบ้าง

ตัวอย่างการวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก

กำหนดให้ตัวอย่างประโยคต่อไปนี้อยู่ในกลุ่มของความสนใจเกี่ยวกับอาหาร “Visit china town after 6pm in the evening to see the real beauty. It's a place where you can get the real shark fin. However, the dog made me creepy when eating.”

จากตัวอย่างข้อความที่กำหนดไว้ข้างต้นสามารถนำมาวิเคราะห์ตัวแทนของคำที่แทนความรู้สึกได้ดังนี้

1. นำบทวิจารณ์ของนักท่องเที่ยวมาตัดแบ่งแยกประโยคโดยการแบ่งด้วยจุด Full Stop ซึ่งเป็นจุดที่แสดงถึงการจบประโยคในภาษาอังกฤษดังนั้นสามารถแบ่งวิจารณ์ได้เป็น 3 ประโยคได้แก่

    1.1 Visit china town after 6pm in the evening to see the real beauty.

    1.2 It's a place where you can get the real shark fin.

    1.3 However, the dog made me creepy when eating.

2. นำทั้งสามประโยคนี้มาผ่านกระบวนการเตรียมข้อมูลแล้วจะประโยคผลลัพธ์ดังนี้

    2.1 Visit china town evening see real beauty.

    2.2 Place get real shark fin.

    2.3 Dog make creepy eat.

3. นำประโยคมาวิเคราะห์ด้วยโมเดลวิเคราะห์ความรู้สึกแบ่งประเภทความรู้สึกของแต่ละประโยคออกเป็นประโยคที่มีความรู้สึกเชิงบวก หรือเชิงลบ

ตารางที่ 3.5 ผลลัพธ์การวิเคราะห์ความรู้สึกของประโยค

ประโยค	ประเภทความรู้สึก	กลุ่มความสนใจ
Visit china town evening see real beauty.	POS	อาหาร
Place get real shark fin.	POS	
Dog make creepy eat.	NEG	

4. นำคำศัพท์มาเทียบกันเพื่อหาคำที่ไม่ซ้ำในแต่ละประเภทความรู้สึก เพื่อนำไปพิจารณาในการหาคำที่เหมาะสม

จากผลลัพธ์ในตารางที่ 3.5 เราสามารถแบ่งคำศัพท์ที่เกิดในตามประเภทความรู้สึกได้ดังนี้ ประเภทความรู้สึกเชิงบวก (POS) มีคำศัพท์ที่เกิดขึ้นในได้แก่ “visit” “china” “town” “evening” “see” “real” “beauty” “eat” “shark” “fin” ประเภทความรู้สึกเชิงลบ (NEG) ได้แก่ “dog” “make” “creepy” “eat” ซึ่งจากการพิจารณาคำศัพท์พบว่ามีคำศัพท์ที่เกี่ยวข้อกับกลุ่มอาหารดังนี้ สำหรับความรู้สึกเชิงบวกมีคำที่เกี่ยวข้อง 3 คำได้แก่ “eat” “shark” “fin” ส่วนความรู้สึกเชิงลบมี 2 คำได้แก่ “eat” “dog” “creepy” อย่างไรก็ตามคำว่า “eat” ไม่ควรนำมาใช้เนื่องจากเป็นคำเกิดขึ้นทั้งในความรู้สึกเชิงบวกและเชิงลบ ซึ่งไม่สามารถบอกได้ว่าเป็นสิ่งที่นักท่องเที่ยวมีความรู้สึกเชิงบวกหรือเชิงลบกับคำๆ นี้

3.8 เว็ปแอพพลิเคชั่นสำหรับการวิเคราะห์กลุ่มความสนใจและความรู้สึก

เครื่องมือวิเคราะห์ความสนใจและความรู้สึกสำหรับนักท่องเที่ยวในงานวิจัยฉบับนี้เป็นเครื่องมือที่ให้ผู้ใช้งานใส่บทวิจารณ์ภาษาอังกฤษของนักท่องเที่ยวที่มีต่อเยาวราชลงไป และกำหนดให้กดเพื่อทำนายเพื่อทำนายว่าบทวิจารณ์ดังกล่าวมีความรู้สึกเป็นอย่างไรและจัดอยู่ในกลุ่มความสนใจใด โดยเว็ปแอพพลิเคชั่นนี้จะถูกนำเสนอในรูปแบบของตัวแบบเว็บแอพพลิเคชั่นที่พัฒนาด้วยภาษาไพธอน และโปรแกรมด๊อกเกอร์ โดยไพธอนจะใช้ในการพัฒนาหน้าเว็ปแอปพลิเคชั่นเพื่อนำไปใช้กับนักวิเคราะห์ข้อมูลที่ต้องการวิเคราะห์บทวิจารณ์ของนักท่องเที่ยวชาวต่างชาติที่วิจารณ์ถึงเยาวราชสามารถเข้าถึงได้ง่าย และอาจนำไปต่อยอดเพื่อวิเคราะห์ความสนใจของนักท่องเที่ยวเพื่อทำนโยบายทางการตลาดต่อไป

3.9 การวิเคราะห์แนวโน้มของคำ (Word Trend Analysis)

การวิเคราะห์แนวโน้มของคำจะเป็นการนำคำศัพท์ที่นักท่องเที่ยวใช้เขียนบทวิจารณ์จากการนับจำนวนคำตามความถี่ของคำที่เกิดขึ้นมากที่สุด 10 คำแรกมาทำการวิเคราะห์การสัดส่วนการเปลี่ยนแปลงที่เกิดขึ้นในแต่ละปี โดยคำศัพท์เหล่านั้นจะถูกนำมาสร้างเป็นกราฟเส้น (Line chart) ที่แสดงถึงหว่างสัดส่วนของคำศัพท์ที่เกิดขึ้นเทียบกันในแต่ละปี เพื่อนำไปใช้ในการอธิบายการเปลี่ยนแปลงของการใช้คำศัพท์ของนักท่องเที่ยวในแต่ละบทวิจารณ์ และเพื่อดูแนวโน้มความสนใจของนักท่องเที่ยวที่มีต่อเยาวราชจากการวิเคราะห์บทวิจารณ์จากคำศัพท์ทั้ง 10 คำ

บทที่ 4
ผลการวิจัยและการอภิปรายผล

จากบทที่ 3 ที่ได้มีการนำเสนอขั้นตอนการดำเนินงานวิจัยและได้นำเสนอดังรูปที่ 3.1 สำหรับบทที่ 4 นี้จะเป็นผลการศึกษาในส่วนต่างๆ ซึ่งประกอบไปด้วย การแบ่งกลุ่มความสนใจของนักท่องเที่ยว การวิเคราะห์ความรู้สึกของนักท่องเที่ยว การวิเคราะห์ความนิยม การวิเคราะห์ความเด่นและความแพร่หลาย การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก การสร้างเว็ปแอพพลิเคชั่น และการวิเคราะห์แนวโน้ม

4.1 ลักษณะของข้อมูลที่ใช้ในการทดลอง

ข้อมูลที่ใช้ในการศึกษาเป็นบทวิจารณ์ที่เขียนด้วยภาษาอังกฤษภาษาอังกฤษของนักท่องเที่ยวผ่านเว็ปไซต์ Tripadvisor ที่เดินทางมายังเยาวราชระหว่างปีค.ศ. 2012 ถึง 2020 โดยมีจำนวนบทวิจารณ์ทั้งสิ้น 3,992 บทวิจารณ์ โดยจำแนกจำนวนบทวิจารณ์ตามคะแนนความพึงพอใจที่มีต่อถนนเยาวราช ตามลำดับคะแนน 1 ถึง 5 ซึ่งแสดงความพึงพอใจจากน้อยไปมากดังตารางที่ 4.1

ตารางที่ 4.1 จำนวนบทวิจารณ์ที่มีต่อถนนเยาวราชจำแนกตามปีและคะแนน

ปี (ค.ศ.)	คะแนน					จำนวนบทวิจารณ์	คะแนนเฉลี่ย
	1	2	3	4	5		
2012	4	6	24	39	42	115	3.9
2013	7	17	27	70	55	176	3.8
2014	8	15	58	110	69	260	3.8
2015	19	40	117	248	201	625	3.9
2016	16	42	178	373	305	914	4.0
2017	26	46	169	358	317	916	4.0
2018	18	25	108	207	226	584	4.0
2019	9	10	54	109	160	342	4.2
2020	0	0	7	28	25	60	4.3
ทั้งหมด	**107**	**201**	**742**	**1,542**	**1,400**	**3,992**	**36.0**

จากตารางที่ 4.1 พบว่าว่าบทวิจารณ์ทั้ง 3,992 บทวิจารณ์ส่วนใหญ่มีระดับคะแนนอยู่ระหว่าง 4-5 คะแนน ซึ่งส่งผลให้ภาพรวมระดับค่าคะแนนเฉลี่ยของแต่ละปีค่อนข้างสูงและมีแนวโน้มเพิ่มขึ้น

ระดับคะแนนในแต่ละปีนี้จะถูกจำแนกเป็นความรู้สึกเชิงบวก และเชิงลบโดยใช้เกณฑ์ค่าคะแนน 4-5 คะแนนแสดงความรู้สึกเชิงบวก และ 1-3 คะแนนแสดงความรู้สึกเชิงลบโดยมีจำนวนบทวิจารณ์ที่เป็นความรู้สึกเชิงบวกจำนวน 2,942 บทวิจารณ์ และบทวิจารณ์ที่เป็นความรู้สึกเชิงลบจำนวน 1,050 บทวิจารณ์

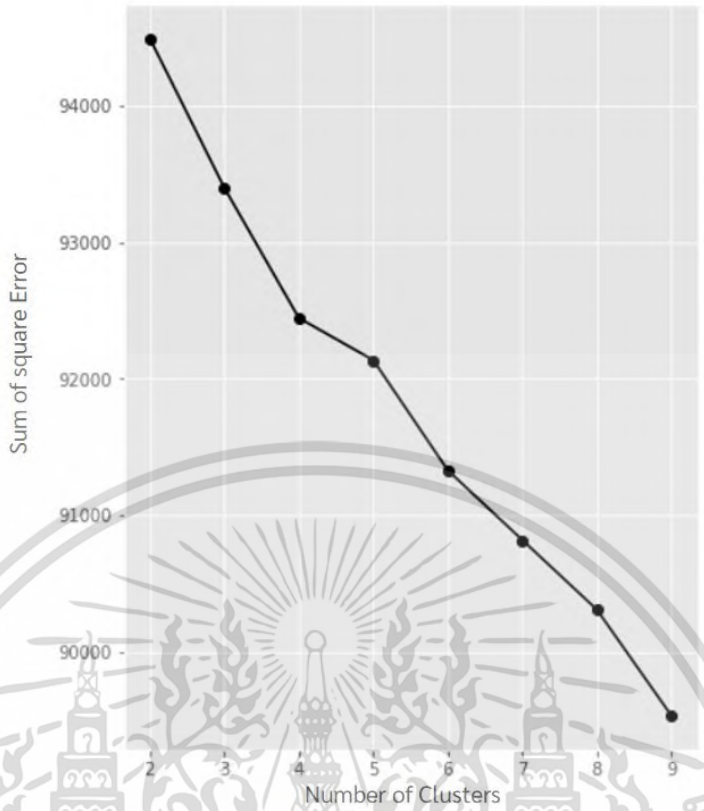
สำหรับการปี 2563 (ค.ศ. 2020) ได้เกิดการแพร่ระบาดของโรคโควิด-19 ทำให้ประเทศไทยมีมาตราการป้องกันการแพร่เชื้อโรคระบาดโดยการปิดรับนักท่องเที่ยวซึ่งทำให้การท่องเที่ยวของประเทศไทยหยุดชะงักตั้งแต่เดือนมีนาคม จึงทำให้ในปี 2563 นี้มีบทวิจารณ์เพียงแค่ 60 บทวิจารณ์

4.2 ผลลัพธ์การแบ่งกลุ่มความสนใจของนักท่องเที่ยว

การแบ่งกลุ่มความสนใจของนักท่องเที่ยวจะเริ่มจากการหาค่ากลุ่มที่เหมาะสมโดยใช้วิธีการจัดกลุ่มแบบเคมีนด้วยการหาจำนวนกลุ่มที่เหมาะสมด้วยวิธีข้อศอก ซึ่งจำนวนกลุ่มที่เหมาะสมนี้จะถูกนำไปใช้ในการกำหนดจำนวนกลุ่มความสนใจของนักท่องเที่ยวที่เหมาะสมในโมเดลการจัดสรรหัวข้อแฝง โดยผลลัพธ์ส่วนนี้จะเป็นคำตอบที่สอดคล้องกับวัตถุประสงค์ในข้อที่ 1 ของงานวิจัยนี้

4.2.1 ผลลัพธ์การวิเคราะห์ข้อมูลแบบเคมีน

การกำหนดจำนวนกลุ่มที่เหมาะสมในการแบ่งกลุ่มความสนใจของนักท่องเที่ยวที่มีต่อเยาวราชสามารถพิจารณาได้จากกราฟความสัมพันธ์ระหว่างผลรวมค่าความคลาดเคลื่อนกำลังสอง (Sum of Square Error) เทียบกับจำนวนกลุ่มที่กำหนดแบบข้อศอกดังรูปที่ 4.1 พบว่าเมื่อจำนวนกลุ่มเปลี่ยนจาก 3 เป็น 4 ทำให้ค่าความคลาดเคลื่อนเกิดจุดหักที่มีลักษณะคล้ายข้อศอกที่หักขึ้นมา ตามหลักวิธีการหาค่ากลุ่มที่เหมาะสมแบบข้อศอก ซึ่งมีความหมายว่าเมื่อเพิ่มจำนวนกลุ่มมากขึ้นอัตราการลดลงของค่าผลรวมความคลาวเคลื่อนกำลังสองลดลงในอัตราที่น้อยลง ดังนั้นกลุ่มความสนใจของนักท่องเที่ยวจะเท่ากับ 4 กลุ่มความสนใจ



รูปที่ 4.1 ผลลัพธ์การแบ่งกลุ่มแบบเคมีน

4.2.2 ผลลัพธ์โมเดลการจัดสรรหัวข้อแฝง

จากการหาจำนวนกลุ่มที่เหมาะสมเท่ากับ 4 กลุ่มจากหัวข้อที่ 4.2.1 ขั้นตอนต่อไปคือการแบ่งกลุ่มความสนใจของนักท่องเที่ยวด้วยการวิเคราะห์หัวข้อแฝงในแต่ละกลุ่มเพื่อกำหนดความสนใจทั้ง 4 กลุ่ม จากผลการวิเคราะห์พบว่าผลลัพธ์ขคำศัพท์ และค่าน้ำหนักซึ่งเป็นคำที่แสดงถึงความสำคัญของคำศัพท์ในแต่ละกลุ่มความสนใจที่แตกต่างกันแสดงตารางที่ 4.2

ตารางที่ 4.2 คำศัพท์ที่กลุ่มความสนใจของนักท่องเที่ยวจากโมเดลการจัดสรรหัวข้อแฝง

ลำดับ	กลุ่มที่ 1		กลุ่มที่ 2		กลุ่มที่ 3		กลุ่มที่ 4	
	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก
1	Shop	0.037	Street	0.023	Food	0.065	Shop	0.024
2	Street	0.023	Walk	0.020	Street	0.042	Food	0.023
3	Cheap	0.020	Food	0.020	Shop	0.024	Good	0.020
4	Area	0.019	Market	0.019	Great	0.016	Street	0.017
5	Good	0.019	Shop	0.019	Good	0.016	Great	0.015
6	Walk	0.017	Go	0.015	Town	0.014	Market	0.014
7	Gold	0.015	Town	0.015	Market	0.013	Road	0.013
8	Food	0.014	People	0.013	Lot	0.013	Restaurant	0.013
9	Time	0.012	Time	0.013	Restaurant	0.012	Stall	0.012
10	Visit	0.011	Night	0.013	Stall	0.012	Taxi	0.011
11	Price	0.003	Crowd	0.005	Fresh	0.006	Look	0.007
12	Yaowarat	0.002	People	0.005	Cheap	0.004	Busy	0.004
13	Sell	0.002	Good	0.005	Recommend	0.004	Interest	0.004
14	Item	0.002	City	0.004	Interest	0.004	Gold	0.004
15	Wholesale	0.002	Best	0.004	Area	0.003	Wander	0.004
16	Cheap	0.002	Time	0.004	Sell	0.003	Crowd	0.003
17	Festival	0.002	Sell	0.003	Real	0.003	People	0.002
18	Comfort	0.002	Interest	0.003	Little	0.003	Like	0.002
19	Time	0.002	Small	0.003	Kind	0.003	Explore	0.002
20	Descript	0.002	Lot	0.003	Image	0.002	Look	0.002

จากตารางที่ 4.2 นำเสนอคำศัพท์ที่เรียงตามน้ำหนักของคำศัพท์ที่เกิดขึ้นในแต่ละกลุ่มความสนใจ พบว่าค่าน้ำหนักของคำที่ 11 เป็นต้นไปมีค่าน้อยกว่า 0.10 และมีค่าต่างจากลำดับที่ 10 ค่อนข้างมากดังนั้นคำศัพท์ที่มีน้ำหนักมากที่สุด 10 คำแรกเรียงตามค่าน้ำหนักจากมากไปน้อยจะถูกเลือกขึ้นมาเพื่อกำหนดกลุ่มความสนใจของนักท่องเที่ยวที่มีต่อเยาวราชดังตารางที่ 4.3

ตารางที่ 4.3 คำศัพท์ที่ถูกเลือกกลุ่มความสนใจของนักท่องเที่ยวจากโมเดลการจัดสรรหัวข้อแฝง

ลำดับ	กลุ่มที่ 1		กลุ่มที่ 2		กลุ่มที่ 3		กลุ่มที่ 4	
	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก	คำศัพท์	ค่าน้ำหนัก
1	Shop	0.037	Street	0.023	Food	0.065	Shop	0.024
2	Street	0.023	Walk	0.020	Street	0.042	Food	0.023
3	Cheap	0.020	Food	0.020	Shop	0.024	Good	0.020
4	Area	0.019	Market	0.019	Great	0.016	Street	0.017
5	Good	0.019	Shop	0.019	Good	0.016	Great	0.015
6	Walk	0.017	Go	0.015	Town	0.014	Market	0.014
7	Gold	0.015	Town	0.015	Market	0.013	Road	0.013
8	Food	0.014	People	0.013	Lot	0.013	Restaurant	0.013
9	Time	0.012	Time	0.013	Restaurant	0.012	Stall	0.012
10	Visit	0.011	Night	0.013	Stall	0.012	Taxi	0.011

การตั้งชื่อกลุ่มความสนใจของนักท่องเที่ยวทั้ง 4 กลุ่มจะพิจารณาจาก ค่าน้ำหนักของคำศัพท์ที่ปรากฏทั้ง 4 กลุ่ม ซึ่งเป็นสิ่งที่นักท่องเที่ยวนิยมใช้คำศัพท์เหล่านี้บ่อย ซึ่งมีคำศัพท์เช่น “street”, “food”, “shop” เป็นต้น

จากการนำคำศัพท์ในตารางที่ 4.3 มาตั้งชื่อกลุ่มความสนใจพบว่า ชื่อกลุ่มความสนใจสามารถตั้งได้โดยการนำคำที่เป็นเอกลักษณ์ในแต่ละกลุ่มมาตั้งชื่อรวมกัน ซึ่งสามารถแสดงชื่อกลุ่มได้ดังแสดงในตารางที่ 4.4

ตารางที่ 4.4 ตารางการตั้งชื่อกลุ่มความสนใจ

ชื่อกลุ่ม	คำศัพท์ที่เกิดขึ้นในกลุ่ม
Shopping Place (แหล่งช้อปปิ้ง)	“Shop”, “Street”, “Cheap”, “Area”, “Good”, “Walk”, “Gold”, “Food”, “Time”, “Visit”
Night Street Food Market (ตลาดอาหารริมทางยามค่ำ)	“Street”, “Walk”, “Food”, “Market”, “Shop”, “Go”, “Town”, “People”, “Time”, “Night”
Food (อาหาร)	“Food”, “Street”, “Shop”, “Great”, “Good”, “Town”, “Market”, “Lot”, “Restaurant”, “Stall”
Sightseeing (การเที่ยวชมเมือง)	“Shop”, “Food”, “Good”, “Street”, “Great”, “Market”, “Road”, “Restaurant”, “Stall”, “Taxi”

4.3 ผลลัพธ์การวิเคราะห์ความรู้สึกของนักท่องเที่ยว

ผลลัพธ์ในส่วนนี้จะเป็นการทำนายความรู้สึกของบทวิจารณ์ต่างๆ ตามวัตถุประสงค์ข้อที่ 2 โดยโมเดลที่ใช้ในการวิเคราะห์ได้แก่โมเดลนาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีน เพื่อที่จำแนกความรู้สึกของนักท่องเที่ยวจากบทวิจารณ์ โดยข้อมูลทั้งหมด 3,992 บทวิจารณ์ จะแบ่งเป็นบทวิจารณ์เชิงบวก 2,942 บทวิจารณ์ และบทวิจารณ์เชิงลบ 1,050 บทวิจารณ์ ซึ่งในงานวิจัยนี้ได้มีการใช้เทคนิค SMOTE เพื่อช่วยทำให้ประเภทของข้อมูลมีจำนวนเท่ากันทำให้บทวิจารณ์เชิงลบมีจำนวนเพิ่มขึ้นเป็น 2,942 บทวิจารณ์ ดังนั้นบทวิจารณ์ทั้งหมดในการทดลองจะมีทั้งหมด 5,884 บทวิจารณ์ โดยการทดลองการทดลองนี้จะเบ่งข้อมูลออกเป็น 2 ส่วนคือ ข้อมูลสำหรับการฝึกสอน 70% หรือ 4,119 บทวิจารณ์ และ 30% สำหรับการทดสอบหรือ 1,765 บทวิจารณ์

ข้อมูลสำหรับการฝึกสอนนั้นจะถูกนำไปใช้สอนโมเดลต่างๆ เพื่อให้โมเดลเกิดการเรียนรู้เพื่อให้สามารถจำแนกประเภทบทวิจารณ์เชิงบวก และเชิงลบ ซึ่งโมเดลที่ผ่านการเรียนรู้นั้นจะถูกนำมาทดสอบด้วยข้อมูลทดสอบ 30% โดยผลลัพธ์จะออกมาในรูปของค่าน้ำหนักของ 5 คำศัพท์แรกที่ส่งผลต่อโมเดลมากที่สุด โดยการวัดผลสำหรับโมเดลจะแบ่งเป็นการวัดผลสำหรับการฝึกสอน และการทดสอบซึ่งใช้ตัววัด 3 แบบได้แก่ ค่าความถูกต้อง ค่าความระลึก และค่าความแม่นยำ

4.3.1 ผลลัพธ์โมเดลนาอีฟเบย์

การฝึกสอนโมเดลนาอีฟเบย์มีจุดประสงค์เพื่อหาค่าความน่าจะเป็นแบบมีเงื่อนไขตามสมการที่ (2.6) อย่างไรก็ตามเนื่องจากในการทำเหมืองข้อความมีการแปลงคำศัพท์ที่พบเจอให้อยู่ในรูปของเวกเตอร์ด้วยวิธีการถุงคำศัพท์ทำให้ไม่สามารถแสดงค่าความน่าจะเป็นออกมาของคำศัพท์ทุกคำออกมาได้ทั้งหมด ดังนั้นในส่วนนี้จะเป็นการแสดงผลลัพธ์ของการฝึกสอนของคำศัพท์ที่มีค่าความน่าจะเป็นแบบมีเงื่อนไขมากที่สุด 5 คำจากทั้งหมด 3,798 คำศัพท์ โดยคำตอบของแต่ละประเภทจะแบ่งเป็น 2 ส่วน ได้แก่ความรู้สึกเชิงบวก (POS) และความรู้สึกเชิงลบ (NEG)

คำศัพท์ที่มีความน่าจะเป็นแบบมีเงื่อนไขสูงสุด 5 อันดับแรกของความรู้สึกเชิงลบ และความรู้สึกเชิงบวกแสดงได้ดังตารางที่ 4.5 และ 4.6

ตารางที่ 4.5 คำศัพท์ที่มีความน่าจะเป็นแบบมีเงื่อนไขสูงสุด 5 อันดับแรกของความรู้สึกเชิงลบ

คำศัพท์	**ความน่าจะเป็นแบบมีเงื่อนไข**
“not”	0.04941
“food”	0.03289
“chinatown”	0.02617
“street”	0.02386
“place”	0.02168

ตารางที่ 4.6 คำศัพท์ที่มีความน่าจะเป็นแบบมีเงื่อนไขสูงสุด 5 อันดับแรกของความรู้สึกเชิงบวก

คำศัพท์	**ความน่าจะเป็นแบบมีเงื่อนไข**
“food”	0.03060
“street”	0.02038
“place	0.01817
“chinatown”	0.01781
“good”	0.01380

ผลการทดสอบสอบประสิทธิภาพของนาอีฟเบย์ด้วยข้อมูลฝึกสอนสามารถแสดงได้ดังตารางที่ 4.7 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 1,885 ครั้ง ผลบวกลวงเท่ากับ 157 ครั้ง ผลลบลวงเท่ากับ 179 ครั้ง และผลลบจริงเท่ากับ 1,898 ครั้ง

ตารางที่ 4.7 ผลการทดสอบของนาอีฟเบย์ด้วยชุดข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	**POS**	**NEG**	**รวม**
POS	1,885	179	**2,064**
NEG	157	1,898	**2,055**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.7 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ตัววัดได้ดังนี้

1. ค่าความถูกต้องสามารถคำนวณได้ดังนี้

$$Accuracy = \frac{1,885 + 1,898}{1,885 + 1,898 + 157 + 179} = 0.9184$$

2. ค่าความระลึกสามารถคำนวณได้ดังนี้

$$Precision = \frac{1,885}{1,885+179} = 0.9133$$

3. ค่าความแม่นยำสามารถคำนวณได้ดังนี้

$$Recall = \frac{1,885}{1,885+157} = 0.9231$$

จากการใช้นาอีฟเบย์ในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลทดสอบ แสดงดังตารางที่ 4.8 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 713 ครั้งผลบวกลวงเท่ากับ 175 ครั้ง ผลลบลวงเท่ากับ 170 ครั้ง และผลลบจริงเท่ากับ 707 ครั้ง

ตารางที่ 4.8 ผลการทดสอบของนาอีฟเบย์ด้วยชุดข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	713	170	**883**
NEG	175	707	**882**
รวม	**888**	**877**	**1,765**

จากตารางที่ 4.8 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ได้ดังนี้

1. ค่าความถูกต้อง 0.8045
2. ค่าความระลึก 0.8075
3. ค่าความแม่นยำ 0.8029

4.3.2 ผลลัพธ์การถดถอยเชิงโลจิสติค

การฝึกสอนโมเดลการถดถอยเชิงโลจิสติคมีจุดประสงค์เพื่อหาค่าน้ำหนักของแต่ละคุณลักษณะซึ่งในที่นี้คือคำศัพท์ต่างๆ เพื่อให้สอดคล้องกับคำศัพท์ที่ใช้ในโมเดลนาอีฟเบย์ ค่าน้ำหนักสูงสุด 5 อันดับของการถดถอยเชิงโลจิสติคที่มีค่าน้ำหนักสูงสุด 5 ค่าแสดงได้ดังตารางที่ 4.9

ตารางที่ 4.9 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของการถดถอยเชิงโลจิสติค

คำศัพท์	น้ำหนัก
“awesome”	0.0068
“serve”	0.0058
“alot”	0.0058

“bike”	0.0058
“fascinating”	0.0056

จากการใช้ถดถอยเชิงโลจิสติคในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลฝึกสอนในการทดสอบดังตารางที่ 4.10 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 2,030 ครั้ง ผลบวกลวงเท่ากับ 12 ครั้ง ผลลบลวงเท่ากับ 13 ครั้ง และผลลบจริงเท่ากับ 2,064 ครั้ง

ตารางที่ 4.10 ผลการทดสอบการถดถอยเชิงโลจิสติคด้วยชุดข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	POS	NEG	**รวม**
POS	2,030	13	**2,043**
NEG	12	2,064	**2,076**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.10 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ได้ดังนี้

1. ค่าความถูกต้อง 0.9939
2. ค่าความระลึก 0.9936
3. ค่าความแม่นยำ 0.9941

จากการใช้การถดถอยเชิงโลจิสติคในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลทดสอบในการทดสอบดังตารางที่ 4.11 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 737 ครั้ง ผลบวกลวงเท่ากับ 151 ครั้ง ผลลบลวงเท่ากับ 142 ครั้ง และผลลบจริงเท่ากับ 735 ครั้ง

ตารางที่ 4.11 ผลการทดสอบการถดถอยเชิงโลจิสติคด้วยชุดข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	**รวม**
POS	737	142	**879**
NEG	151	735	**886**
รวม	**888**	**877**	**1,765**

จากตารางที่ 4.11 สามารถคำนวณตัววัดประสิทธิภาพทั้ง 3 ได้ดังนี้

1. ค่าความถูกต้อง 0.8340

2. ค่าความระลึก 0.8385
3. ค่าความแม่นยำ 0.8300

4.3.3 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น

การฝึกสอนโมเดลซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นมีจุดประสงค์เพื่อหาค่าน้ำหนักของแต่ละคุณลักษณะคล้ายกับการถดถอยเชิงโลจิสติค โดยค่าน้ำหนักสูงสุด 5 อันดับของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น แสดงได้ดังตารางที่ 4.12

ตารางที่ 4.12 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้น

คำศัพท์	**น้ำหนัก**
“gear”	0.0208
“squishie”	0.0193
“labyrinth”	0.0191
“theme”	0.0189
“shoplot”	0.0178

ผลการทดสอบสอบประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลฝึกสอนสามารถแสดงได้ดังตารางที่ 4.13 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 2,029 ครั้ง ผลบวกลวงเท่ากับ 13 ครั้ง ผลลบลวงเท่ากับ 17 ครั้ง และผลลบจริงเท่ากับ 2,060 ครั้ง

ตารางที่ 4.13 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	**POS**	**NEG**	**รวม**
POS	2,029	17	**2,046**
NEG	13	2,060	**2,073**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.13 สามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.9927
2. ค่าความระลึก 0.9917
3. ค่าความแม่นยำ 0.9936

ผลการทดสอบสอบประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลฝึกสอน สามารถแสดงได้ดังตารางที่ 4.14 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 717 ครั้ง ผลบวกลวงเท่ากับ 163 ครั้ง ผลลบลวงเท่ากับ 137 ครั้ง และผลลบจริงเท่ากับ 748 ครั้ง

ตารางที่ 4.14 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	**รวม**
POS	717	137	**854**
NEG	163	748	**911**
รวม	**880**	**885**	**1,765**

จากตารางที่ 4.14 สามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.8300
2. ค่าความระลึก 0.8396
3. ค่าความแม่นยำ 0.8148

4.3.4 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้น

ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นเป็นการใช้ฟังก์ชันต่างๆ เพื่อแปลงคุณลักษณะให้อยู่ในมิติที่สูงขึ้นเพื่อทำให้ซัพพอร์ตเวกเตอร์แมชชีนมีความสามารถจำแนกข้อมูลที่ไม่เป็นเชิงเส้นได้มีประสิทธิภาพมากขึ้น อย่างไรก็ตามเนื่องจากการแปลงคุณลักษณะให้อยู่มิติที่สูงขึ้นนั้นจะทำให้การอภิปรายผลค่าน้ำหนักมีความซับซ้อนมากขึ้นตามไปด้วย โดยในโปรแกรมภาษาไพธอนสามารถแสดงผลลัพธ์ของค่าน้ำหนักของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงที่ใช้ฟังก์ชันเคอร์เนลแบบเชิงเส้นเท่านั้นจึงทำให้ในส่วนของอีก 2 เคอร์เนลฟังก์ชันที่เหลือได้แก่ ฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยล และฟังก์ชันเคอร์เนลแบบเชิงเส้นแบบ RBF ไม่สามารถแสดงผลค่าน้ำหนักได้

จากการฝึกสอนซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นพบว่าคำศัพท์ที่มีน้ำหนักสูงสุด 5 อันดับ ซึ่งแสดงได้ดังตารางที่ 4.15

ตารางที่ 4.15 คำศัพท์ที่มีค่าน้ำหนักสูงสุด 5 อันดับแรกของซัพพอร์ตเวกเตอร์แมชชีนเชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิง

คำศัพท์	**น้ำหนัก**
“exist”	0.01431
“wow”	0.01325
“market”	0.01222
“good”	0.01121
“awesome”	0.01045

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลฝึกสอนในการทดสอบดังตารางที่ 4.16 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 1,975 ครั้ง ผลบวกลวงเท่ากับ 67 ครั้ง ผลลบลวงเท่ากับ 49 ครั้ง และผลลบจริงเท่ากับ 2,028 ครั้ง

ตารางที่ 4.16 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นด้วยข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	**POS**	**NEG**	**รวม**
POS	1,975	49	**2,024**
NEG	67	2,028	**2,095**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.16 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.9718
2. ค่าความระลึก 0.9758
3. ค่าความแม่นยำ 0.9672

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลทดสอบในการทดสอบดังตารางที่ 4.17 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 724 ครั้ง ผลบวกลวงเท่ากับ 164 ครั้ง ผลลบลวงเท่ากับ 135 ครั้ง และผลลบจริงเท่ากับ 742 ครั้ง

ตารางที่ 4.17 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบเชิงเส้นด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	724	135	**859**
NEG	164	742	**906**
รวม	**888**	**877**	**1,765**

จากตารางที่ 4.17 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.8306
2. ค่าความระลึก 0.8428
3. ค่าความแม่นยำ 0.8153

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลฝึกสอนในการทดสอบดังตารางที่ 4.18 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 2,007 ครั้ง ผลบวกลวงเท่ากับ 35 ครั้ง ผลลบลวงเท่ากับ 1,389 ครั้ง และผลลบจริงเท่ากับ 688 ครั้ง

ตารางที่ 4.18 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลด้วยข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	2,007	1,389	**3,396**
NEG	35	688	**723**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.18 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.6543
2. ค่าความระลึก 0.5910
3. ค่าความแม่นยำ 0.9829

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลทดสอบในการทดสอบดังตารางที่ 4.19 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 860 ครั้ง ผลบวกลวงเท่ากับ 28 ครั้ง ผลลบลวงเท่ากับ 779 ครั้ง และผลลบจริงเท่ากับ 98 ครั้ง

ตารางที่ 4.19 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบโพลิโนเมี่ยลด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	860	779	**1,639**
NEG	28	98	**126**
รวม	**888**	**877**	**1,765**

จากตารางที่ 4.19 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.5428
2. ค่าความระลึก 0.5247
3. ค่าความแม่นยำ 0.9685

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลฝึกสอนในการทดสอบดังตารางที่ 4.20 พบว่าจากบทวิจารณ์สำหรับฝึกสอน 4,119 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 1,978 ครั้ง ผลบวกลวงเท่ากับ 64 ครั้ง ผลลบลวงเท่ากับ 49 ครั้ง และผลลบจริงเท่ากับ 2,028 ครั้ง

ตารางที่ 4.20 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ด้วยข้อมูลฝึกสอน

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	1,978	49	**2,027**
NEG	64	2,028	**2,092**
รวม	**2,042**	**2,077**	**4,119**

จากตารางที่ 4.20 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.9726
2. ค่าความระลึก 0.9758
3. ค่าความแม่นยำ 0.9687

จากการใช้ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ในการจำแนกประเภทความรู้สึกโดยใช้ข้อมูลทดสอบในการทดสอบดังตารางที่ 4.21 พบว่าจากบทวิจารณ์สำหรับทดสอบ 1,765 บทวิจารณ์ มีค่าผลบวกจริงเท่ากับ 746 ครั้ง ผลบวกลวงเท่ากับ 142 ครั้ง ผลลบลวงเท่ากับ 138 ครั้ง และผลลบจริงเท่ากับ 739 ครั้ง

ตารางที่ 4.21 ผลการทดสอบซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF ด้วยข้อมูลทดสอบ

ผลทำนาย / ผลจริง	POS	NEG	รวม
POS	746	138	**884**
NEG	142	739	**881**
รวม	**888**	**877**	**1,765**

จากตารางที่ 4.21 ซึ่งสามารถคำนวณเป็นตัววัดประสิทธิภาพทั้ง 3 ได้เป็น

1. ค่าความถูกต้อง 0.8390
2. ค่าความระลึก 0.8416
3. ค่าความแม่นยำ 0.8378

จากการทดลองทดสอบซัพพอร์ตเวกเตอร์แมชชีนด้วยข้อมูลทดสอบไม่เชิงเส้นทั้ง 3 โมเดล พบว่าซัพพอร์เวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF มีค่าชี้วัดความถูกต้อง ความแม่นยำและ ความระลึกมากที่สุด

4.3.5 การเพิ่มประสิทธิภาพโมเดล

ในงานวิจัยฉบับนี้มีการเพิ่มประสิทธิภาพของโมเดลด้วยการกำหนดค่าพารามิเตอร์ที่เหมาะสมเพื่อทำให้โมเดลมีประสิทธิภาพสูงสุด โดยในการเปรียบเทียบเพื่อหาพารามิเตอร์ที่ดีที่สุดนั้นกำหนดโดยการเปรียบเทียบค่าประสิทธิภาพได้แก่ ความถูกต้อง ความระลึก และความแม่นยำตามหัวข้อที่ 4.3

จากหัวข้อที่ 4.3.1 ถึง 4.3.4 เมื่อเราวัดผลโมเดลด้วยตัววัดทั้ง 3 แบบแล้วทำให้ได้ตัวแทนของโมเดลทั้ง 3 โมเดลมาได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF แต่เนื่องจากนาอีฟเบย์ไม่มีการปรับค่าพารามิเตอร์ดังนั้น การเพิ่มประสิทธิภาพของโมเดลจะมีเพียง 2 โมเดลได้แก่ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF

4.3.5.1 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติค

จากการใช้ทฤษฎีในหัวข้อที่ 2.3.5 ผลลัพธ์ของการถดถอยเชิงโลจิสติคจะแบ่งเป็น 2 ส่วน ได้แก่ การถดถอยเชิงโลจิสติคปกติซึ่งได้ผลลัพธ์ความถูกต้องเท่ากับ 0.8316 หรือ 83.16% โดยเมื่อนำโมเดลมาทำการเลกูลาร์ไรซ์แล้วสามารถแสดงได้ดังตารางที่ 4.22

การถดถอยเชิงโลจิสติคด้วยการเรกูลาไรซ์แบบ Lasso และ Ridge ซึ่งทำได้จากการเปลี่ยนพารามิเตอร์ penalty และการค่าปรับ C ซึ่งเป็นค่าความรุนแรงของเรกูลาร์ไรซ์ ที่ปรับได้จากพารามิเตอร์ “C” ในไลบรารี่ โดยการปรับค่าพารามิเตอร์ต่างๆ ผลลัพธ์การเพิ่มประสิทธิภาพด้วยชุดข้อมูลฝึกหัด และชุดข้อมูลทดสอบของการเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติคแสดงไว้ในตารางที่ 4.22 และ 4.23

ตารางที่ 4.22 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลฝึกสอน

C	Lasso			Ridge		
	ความถูกต้อง	**ความระลึก**	**ความแม่นยำ**	**ความถูกต้อง**	**ความระลึก**	**ความแม่นยำ**
2^{-15}	0.6664	0.6499	0.6665	0.7344	0.7509	0.7510
2^{-13}	0.6420	0.6603	0.6524	0.7583	0.7400	0.7504
2^{-11}	0.6483	0.6718	0.6702	0.8086	0.7851	0.8070
2^{-9}	0.6272	0.6721	0.6496	0.9092	0.8643	0.8867
2^{-7}	0.6534	0.6682	0.6718	0.9728	0.8911	0.8644
2^{-5}	0.8389	0.8204	0.8277	0.9820	0.9205	0.9393
2^{-3}	0.9451	0.9498	0.9495	0.9942	0.9940	0.9978
2^{-1}	0.9440	0.9549	0.9309	0.9013	0.8904	0.8904
2^{1}	0.9579	0.9521	0.9374	0.8856	0.8914	0.8914
2^{3}	0.9443	0.9716	0.9484	0.9063	0.8790	0.8790
2^{5}	0.9331	0.9330	0.9066	0.8722	0.8723	0.8723
2^{7}	0.9528	0.9660	0.9633	0.9108	0.8976	0.8976
2^{9}	0.9523	0.9642	0.9656	0.9148	0.8819	0.8819
2^{11}	0.9923	0.9852	0.9960	0.8743	0.9094	0.9094
2^{13}	0.9389	0.9364	0.9320	0.8846	0.8871	0.8871
2^{15}	0.9299	0.9108	0.9172	0.8764	0.8955	0.8955

จากการนำโมเดลการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลฝึกสอนดังตารางที่ 4.22 พบว่าการเพิ่มประสิทธิภาพการถดถอยเชิงโลจิสติคแบบ Lasso ที่มีค่า C เท่ากับ 2^{11} ให้ค่าความถูกต้อง ความระลึก และความแม่นยำสูงที่สุดในการเพิ่มประสิทธิภาพ และสำหรับในส่วนของการถดถอยเชิงโลจิสติคแบบ Ridge เมื่อ C เท่ากับ 2^{-3} จะทำให้ความถูกต้อง ความระลึก และความแม่นยำสูงสุด

เมื่อทำการเปรียบเทียบการเพิ่มประสิทธิภาพทั้ง 2 โมเดล พบว่าเมื่อใช้ข้อมูลสำหรับฝึกสอนในการเรียนรู้โมเดลวิธีแบบ Ridge สามารถเพิ่มประสิทธิภาพของโมเดลได้ดีกว่าแบบ Lasso เนื่องจากเมื่อเปรียบเทียบ ผลลัพธ์ความถูกต้อง ความระลึก และความแม่นยำระหว่างวิธี Ridge และ Lasso แล้ว ผลลัพธ์ทั้ง 3 ค่าของวิธี Ridge มีค่ามากกว่าวิธีของ Lasso

ตารางที่ 4.23 การเพิ่มประสิทธิภาพของการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลทดสอบ

C	Lasso			Ridge		
	ความถูกต้อง	**ความระลึก**	**ความแม่นยำ**	**ความถูกต้อง**	**ความระลึก**	**ความแม่นยำ**
2^{-15}	0.4966	0.5382	0.4646	0.5811	0.5423	0.6615
2^{-13}	0.4966	0.5548	0.4800	0.5946	0.6768	0.6551
2^{-11}	0.4966	0.4099	0.5268	0.6334	0.7026	0.6820
2^{-9}	0.4966	0.5089	0.5866	0.7337	0.7569	0.7138
2^{-7}	0.4966	0.5328	0.5397	0.8226	0.7393	0.7886
2^{-5}	0.6785	0.7161	0.6013	0.8401	0.8198	0.8111
2^{-3}	0.7922	0.8089	0.8007	0.8412	0.8388	0.8467
2^{-1}	0.8148	0.7304	0.8163	0.8378	0.7422	0.7741
2^{1}	0.8193	0.8056	0.7714	0.8294	0.8100	0.7644
2^{3}	0.8142	0.8040	0.7692	0.8255	0.7619	0.7920
2^{5}	0.8063	0.8067	0.7969	0.8221	0.7740	0.7262
2^{7}	0.8024	0.7558	0.8045	0.8238	0.8915	0.7780
2^{9}	0.8187	0.7189	0.8059	0.8249	0.7472	0.7585
2^{11}	0.8243	0.8116	0.8245	0.8210	0.8301	0.7511
2^{13}	0.7900	0.7458	0.8028	0.8148	0.7189	0.7997
2^{15}	0.7703	0.6823	0.6891	0.8125	0.8019	0.7448

จากการนำโมเดลการถดถอยเชิงโลจิสติคโดยใช้ข้อมูลทดสอบดังตารางที่ 4.23 พบว่าการเพิ่มประสิทธิภาพการถดถอยเชิงโลจิสติคแบบ Lasso ที่มีค่า C เท่ากับ 2^{11} ให้ค่าความถูกต้อง ความระลึก และความแม่นยำสูงที่สุดในการเพิ่มประสิทธิภาพ และสำหรับในส่วนของการถดถอยเชิงโลจิสติคแบบ Ridge เมื่อ C เท่ากับ 2^{-3} จะทำให้ความถูกต้อง ความระลึก และความแม่นยำสูงสุดเช่นกัน ซึ่งสอดคล้องกับการวัดผลโมเดลในชุดข้อมูลฝึกสอน

เมื่อทำการเปรียบเทียบการเพิ่มประสิทธิภาพโมเดลทั้ง 2 พบว่าเมื่อใช้ข้อมูลสำหรับทดสอบโมเดลวิธีแบบ Ridge สามารถเพิ่มประสิทธิภาพของโมเดลได้ดีกว่าแบบ Lasso เนื่องจากเมื่อเปรียบเทียบ ผลลัพธ์ความถูกต้อง ความระลึก และความแม่นยำระหว่างวิธี Ridge และ Lasso แล้วผลลัพธ์ทั้ง 3 ค่าของวิธี Ridge มีค่ามากกว่าวิธีของ Lasso ซึ่งสอดคล้องกับการวัดผลโมเดลในชุด

ข้อมูลฝึกสอน และชุดทดสอบ ดังนั้นการถดถอยเชิงโลจิสติคแบบเรกูลาไรซ์ด้วย Ridge จะเป็นตัวแทนของโมเดลการถดถอยเชิงโลจิสติคที่ใช้ในการสร้างโมเดลการตัดสินใจร่วมกันแบบเสียงข้างมาก

4.3.5.2 การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีน

จากการวัดค่าความถูกต้องของซัพพอร์ตเวกเตอร์แมชชีนทั้งหมดพบว่า ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบ RBF ซึ่งจะมีค่าพารามิเตอร์ 2 ตัวให้ปรับได้แก่ C ซึ่งเป็นค่าความรุนแรงของเรกูลาร์ไรซ์ และ γ คือความสัมพันธ์ของความอคติและความแปรปรวน ซึ่งสามารถปรับได้จากพารามิเตอร์ “C” และ “gamma” ในไลบรารี่ ผลลัพธ์การเพิ่มประสิทธิภาพด้วยข้อมูลฝึกสอนและข้อมูลทดสอบแสดงไว้ในตารางที่ 4.24 และ 4.25 โดยผลลัพธ์ในตารางจะแสดงตารางของค่าพารามิเตอร์ที่ทำให้ค่าความถูกต้องสูงสุดเท่านั้น สำหรับพารามิเตอร์ค่าอื่นๆ สามารถอ้างอิงได้จากภาคผนวก ก

ตารางที่ 4.24 การเพิ่มประสิทธิภาพซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF โดยใช้ข้อมูลฝึกสอน

C	γ	**ความถูกต้อง**	**ความระลึก**	**ความแม่นยำ**
2^{1}	2^{-15}	0.5562	0.6051	0.5072
	2^{-13}	0.5739	0.5520	0.4771
	2^{-11}	0.8514	0.8636	0.9002
	2^{-9}	0.8905	0.8745	0.8765
	2^{-7}	0.9707	0.9627	0.9755
	2^{-5}	0.9837	0.9889	0.9857
	2^{-3}	0.9811	0.9485	0.9698
	2^{-1}	0.8880	0.8211	0.8998
	2^{1}	0.7337	0.7543	0.6224
	2^{3}	0.6441	0.5668	0.6479
	2^{5}	0.6294	0.5455	0.5663
	2^{7}	0.6359	0.5874	0.7081
	2^{9}	0.6163	0.6623	0.5733
	2^{11}	0.6104	0.5292	0.5479
	2^{13}	0.6060	0.6880	0.6801
	2^{15}	0.6185	0.6627	0.5387

จากตารางที่ 4.24 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF พบว่าหลังจากปรับค่าพารามิเตอร์ทั้งสองตัวได้แก่ C ซึ่งเป็นพารามิเตอร์ของการเรกูลาร์ไรซ์ และพารามิเตอร์ของฟังก์ชันเนอร์เนล γ ตั้งแต่ 2^{-15} ถึง 2^{15} พบว่าเมื่อกำหนดค่าพารามิเตอร์ C และ γ เป็น 2^{1} และ 2^{-5} ตามลำดับจะทำให้โมเดลซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF ที่เรียนรู้ด้วยข้อมูลฝึกสอนมีค่าความถูกต้อง ความระลึก และความแม่นยำสูงที่สุดดังนั้นเราจะนำค่า C เท่ากับ 2^{1} และ γ เท่ากับ 2^{-5} ไปใช้ในการทำโมเดลที่ทดสอบด้วยข้อมูลทดสอบเช่นกัน

ตารางที่ 4.25 การเพิ่มประสิทธิภาพซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF โดยใช้ข้อมูลทดสอบ

C	γ	**ความถูกต้อง**	**ความระลึก**	**ความแม่นยำ**
2^{1}	2^{-15}	0.5034	0.5309	0.4319
	2^{-13}	0.5068	0.4787	0.4080
	2^{-11}	0.7838	0.7903	0.8016
	2^{-9}	0.8204	0.8185	0.8035
	2^{-7}	0.8378	0.8282	0.8140
	2^{-5}	0.8446	0.8563	0.8356
	2^{-3}	0.8305	0.8503	0.8244
	2^{-1}	0.8029	0.7404	0.8218
	2^{1}	0.6408	0.6627	0.5685
	2^{3}	0.5507	0.4999	0.5925
	2^{5}	0.5507	0.4734	0.5095
	2^{7}	0.5507	0.5346	0.6440
	2^{9}	0.5507	0.5710	0.4852
	2^{11}	0.5507	0.4519	0.4619
	2^{13}	0.5507	0.6377	0.5812
	2^{15}	0.5507	0.5694	0.4569

จากตารางที่ 4.25 ผลลัพธ์ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBF โดยใช้ข้อมูลทดสอบ พบว่าหลังจากปรับค่าพารามิเตอร์ทั้งสองตัวได้แก่ C และ γ ตั้งแต่ 2^{-15} ถึง 2^{15} พบว่าเมื่อกำหนดค่าพารามิเตอร์ C และ γ เป็น 2^{1} และ 2^{-5} ตามผลการทดสอบโมเดลด้วยชุดข้อมูลเรียนรู้ทำให้ผลลัพธ์มีความถูกต้อง ความระลึก และความแม่นยำสูงที่สุด ทำให้สรุปได้ว่ากำหนดให้ค่าพารามิเตอร์ C และ γ เป็น 2^{1} และ 2^{-5} ตามลำดับจะทำให้โมเดลซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันของเคอร์เนลแบบ RBFมีประสิทธิภาพสูงที่สุด

4.3.6 ผลลัพธ์การเลือกโมเดลมาช่วยในการตัดสินใจ

เมื่อนำผลลัพธ์ของโมเดลที่ที่ดีสุดของทั้ง 3 โมเดลได้แก่นาอีฟเบย์ การถดถอยเชิงโลจิสแบบเรกูลาไรซ์ด้วย Ridge และซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยเคอร์เนล RBF กำหนดเลือกโมเดลร่วมกันตัดสินใจแบบเสียงข้างมาก โดยจะเริ่มจากการนำโมเดลทั้ง 3 มาทำการจำแนกความรู้สึกเป็นเชิงบวกหรือเชิงลบ และนำผลคำตอบทั้ง 3 มาทำนายร่วมกัน โดยจะเลือกจำแนกตามผลของโมเดลอย่างน้อย 2 ใน 3 ของโมเดล ซึ่งเป็นหลักการของการจำแนกแบบเสียงข้างมากดังแสดงในตารางที่ 4.26 และ 4.27

ตารางที่ 4.26 ค่าความถูกต้องของโมเดลทั้ง 3 เมื่อเทียบกับโมเดลร่วมกันตัดสินใจแบบเสียงข้างมาก โดยใช้ข้อมูลชุดฝึกสอน

โมเดล	ความถูกต้อง	ความระลึก	ความแม่นยำ
นาอีฟเบย์	0.9184	0.9133	0.9231
การถดถอยเชิงโลจิสติค	0.9942	0.9940	0.9978
ซัพพอร์ตเวกเตอร์แมชชีน	0.9837	0.9889	0.9857
โมเดลร่วมกันตัดสินใจแบบเสียงข้างมาก	0.9812	0.9888	0.9913

ตารางที่ 4.27 ค่าความถูกต้องของโมเดลทั้ง 3 เมื่อเทียบกับโมเดลร่วมกันตัดสินใจแบบเสียงข้างมาก โดยใช้ข้อมูลทดสอบ

โมเดล	ความถูกต้อง	ความระลึก	ความแม่นยำ
นาอีฟเบย์	0.8045	0.8075	0.8029
การถดถอยเชิงโลจิสติค	0.8412	0.8388	0.8467
ซัพพอร์ตเวกเตอร์แมชชีน	0.8446	0.8563	0.8356
โมเดลร่วมกันตัดสินใจแบบเสียงข้างมาก	0.8862	0.8741	0.8632

จากการใช้ข้อมูลสำหรับฝึกสอนในการวัดผลในตารางที่ 4.26 พบว่าเมื่อให้โมเดลทั้ง 3 ตัดสินใจร่วมกัน ผลลัพธ์ของโมเดลการตัดสินใจร่วมกันมีความถูกต้อง ความระลึก และความแม่นยำมีทั้งต่ำกว่า และสูงกว่าเมื่อเทียบมากกว่าโมเดลเดี่ยวทั่วไป อย่างไรก็ตามเมื่อใช้ข้อมูลทดสอบในการวัดผมโมเดลพบว่า โมเดลร่วมกันตัดสินใจแบบเสียงข้างมากให้ผลลัพธ์ที่ดีกว่าทั้ง 3 ตัววัดซึ่งสามารถสรุปได้ว่าเมื่อนำโมเดลทั้ง 3 มาตัดสินใจร่วมกันแบบเสียงข้างมากจะทำให้มีประสิทธิภาพมากกว่าโมเดลเดี่ยวๆ ทั่วไป ซึ่งแปลว่าโมเดลการตัดสินใจร่วมกันแบบเสียงข้างมากมีประสิทธิภาพที่ดีกว่า

4.4 ผลลัพธ์การวิเคราะห์ความนิยม

การวิเคราะห์ความนิยมที่นำเสนอด้วยคะแนนความพึงพอใจที่นักท่องเที่ยวให้ไว้ในแต่ละบทวิจารณ์โดยนำบทวิจารณ์เฉพาะกลุ่มในแต่ละกลุ่มความสนใจมาเปรียบเทียบความพึงพอใจในภาพรวมของนักท่องเที่ยวที่ได้นำเสนอไปในหัวข้อที่ 4.2 มาเป็นแนวทางในปรับปรุงเยาวราชซึ่งเป็นวัตถุประสงค์ข้อที่ 3

จากผลลัพธ์คะแนนเฉลี่ยตวามความพึงพอใจของงานวิจัยนี้ พบว่าผู้มีความสนใจด้านอาหารมีคะแนนสูงสุดที่ 4.02 คะแนน จากจำนวน 2,425 บทวิจารณ์ รองลงมาคือผู้ที่สนใจด้านแหล่งช้อปปิ้งมีคะแนน 3.98 คะแนน จาก 753 บทวิจารณ์ ผู้ที่สนใจด้านการเที่ยวชมเมืองมีคะแนน 3.96 จาก 229 บทวิจารณ์ และสุดท้ายผู้มีความสนใจด้านตลาดอาหารริมทางยามค่ำ 3.95 คะแนน จาก 585 บทวิจารณ์ดังแสดงในตารางที่ 4.28

ตารางที่ 4.28 คะแนนเฉลี่ยของแต่ละกลุ่มความสนใจ

กลุ่มความสนใจ	จำนวนบทวิจารณ์	คะแนนเฉลี่ย
อาหาร	2,425	4.02
แหล่งช้อปปิ้ง	753	3.98
การเที่ยวชมเมือง	229	3.96
ตลาดอาหารริมทางยามค่ำ	585	3.95
รวม	3,992	15.91

4.5 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลาย

4.5.1 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลายเชิงกลุ่ม

ถนนเยาวราชที่มีการแบ่งกลุ่มความสนใจออกเป็น 4 กลุ่มความสนใจ จากการวิเคราะห์ค่าความเด่นเชิงกลุ่มซึ่งบ่งบอกถึงสัดส่วนจำนวนของบทวิจารณ์ที่นักท่องเที่ยวในแต่ละกลุ่มเทียบกับ

จำนวนบทวิจารณ์ทั้งหมด และความแพร่หลายเชิงกลุ่มจะบ่งบอกถึงความรู้สึกของนักท่องเที่ยวในกลุ่มนั้นๆ เทียบกับจำนวนบทวิจารณ์เชิงบวกที่คาดหวังโดยค่าความเด่นและความแพร่หลายสามารถแสดงได้ดังตารางที่ 4.29

ตารางที่ 4.29 ค่าความเด่นและค่าความแพร่หลายจำนวนกลุ่มความสนใจ

กลุ่มความสนใจ	จำนวนบทวิจารณ์เชิงบวก	จำนวนบทวิจารณ์เชิงบวกที่คาดหวัง	จำนวนบทวิจารณ์ทั้งหมด	ความเด่น	ความแพร่หลาย
อาหาร	1,769	1,763	2,425	60.75	0.25
แหล่งช้อปปิ้ง	548	547	753	18.86	0.08
ตลาดอาหารริมทางยามค่ำ	417	425	585	14.65	-1.41
เที่ยวชมเมือง	168	166	229	5.74	0.67

เมื่อเรียงค่าความเด่นจากมากไปน้อยพบว่า กลุ่มอาหาร มีค่าความเด่น 60.75% แหล่งช้อปปิ้ง 18.86% ตลาดอาหารริมทางยามค่ำ 14.65% เที่ยวชมเมือง 5.74% ซึ่งสามารถตีความได้ว่านักท่องเที่ยวที่เดินทางมายังเยาวราชมีเขียนบทวิจารณ์เกี่ยวกับกลุ่มความสนใจอาหารมากที่สุด และกลุ่มแหล่งช้อปปิ้ง กลุ่มตลาดอาหารริมทางยามค่ำ และกลุ่มเที่ยวชมเมืองตามลำดับ

ในด้านของค่าความแพร่หลายพบว่า ความแพร่หลายที่มีค่าเป็นบวกมีทั้งหมด 3 กลุ่มได้แก่ กลุ่มอาหาร กลุ่มแหล่งช้อปปิ้ง และกลุ่มเที่ยวชมเมือง โดยกลุ่มเที่ยวชมเมืองได้ค่าความแพร่หลายมากที่สุดอยู่ที่ 0.67 กลุ่มอาหาร 0.25 และกลุ่มแหล่งช้อปปิ้ง 0.08 ซึ่งแสดงให้เห็นว่านักท่องเที่ยวมีความรู้สึกในแง่บวกกับ 3 กลุ่มนี้ อย่างไรก็ตามกลุ่มตลาดอาหารริมทางยามค่ำนั้นมีค่าความแพร่หลายเป็นลบซึ่งแสดงถึงนักท่องเที่ยวมีความรู้สึกแง่ลบในกลุ่มความสนใจนี้ โดยสาเหตุของความไม่ชอบใจนี้ในงานวิจัยนี้จะอธิบายด้วยการวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์ซึ่งเป็นการหาข้อมูลเชิงลึกในแต่ละกลุ่ม

4.5.2 ผลลัพธ์การวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์

ผลลัพธ์ของการวิเคราะห์ความเด่นและความแพร่หลายเชิงคำศัพท์จะแสดงอยู่ในรูปของตารางซึ่งแสดงด้วยคำศัพท์เฉพาะทั้ง 10 คำที่คัดเลือกมาจากผลลัพธ์ของโมเดลการจัดสรรหัวข้อแฝงในหัวข้อที่ 4.2.1 โดยความเด่น และความแพร่หลายของคำศัพท์ในแต่ละกลุ่มความสนใจซึ่งเป็นการวิเคราะห์มาจากบทวิจารณ์ทั้งหมด 3,992 บทวิจารณ์ โดยค่าความเด่นเชิงคำศัพท์คือค่าความถี่ที่เกิดจากการใช้คำของนักท่องเที่ยว ยิ่งนักท่องเที่ยวมีการใช้คำศัพท์เฉพาะนี้มากเพียงใดก็จะมีค่าความเด่น

จะมีค่ามากขึ้นเท่านั้น ส่วนค่าความแพร่หลายเชิงคำศัพท์นั้นจะเป็นค่าที่บ่งบอกถึงความรู้สึกของนักท่องเที่ยวที่มีต่อคำศัพท์เฉพาะนั้นๆ ซึ่งมีความหมายว่าโดยเฉลี่ยแล้วศัพท์คำเฉพาะเหล่านี้ถูกนักท่องเที่ยวนำมาใช้ในการเขียนบทวิจารณ์ในเชิงบวก (4-5 คะแนน) หรือเชิงลบ (1-3 คะแนน) กันแน่ โดยผลลัพธ์ของค่าความเด่นและความแพร่หลายสามารถแสดงได้ดังตารางที่ 4.30

ตารางที่ 4.30 ค่าความเด่นและความแพร่หลายของกลุ่มความสนใจทั้ง 4 กลุ่ม

กลุ่มความสนใจแหล่งช้อปปิ้ง				**กลุ่มความสนใจตลาดอาหารริมทางยามค่ำ**			
คำศัพท์	**น้ำหนัก**	**ความเด่น**	**ความแพร่หลาย**	**คำศัพท์**	**น้ำหนัก**	**ความเด่น**	**ความแพร่หลาย**
Shop	0.037	3.09	0.01	Street	0.023	3.25	0
Street	0.023	3.25	0	Walk	0.020	2.9	0
Cheap	0.020	2.66	-0.02	Food	0.020	3.41	0.02
Area	0.019	2.77	-0.02	Market	0.019	2.87	0.01
Good	0.019	3.06	0.01	Shop	0.019	3.09	0.01
Walk	0.017	2.9	0	Go	0.015	3.35	0.03
Gold	0.015	2.48	0.04	Town	0.015	2.85	-0.03
Food	0.014	3.41	0.02	People	0.013	2.73	0
Time	0.012	2.76	0	Time	0.013	2.76	0
Visit	0.011	3.03	0	Night	0.013	2.74	-0.01
กลุ่มความสนใจอาหาร				**กลุ่มความสนใจเที่ยวชมเมือง**			
คำศัพท์	**น้ำหนัก**	**ความเด่น**	**ความแพร่หลาย**	**คำศัพท์**	**น้ำหนัก**	**ความเด่น**	**ความแพร่หลาย**
Food	0.065	3.41	0.02	Shop	0.024	3.09	0.01
Street	0.042	3.25	0	Food	0.023	3.41	0.02
Shop	0.024	3.09	0.01	Good	0.020	3.06	0.01
Great	0.016	2.86	0.04	Street	0.017	3.25	0
Good	0.016	3.06	0.01	Great	0.015	2.86	0.04
Town	0.014	2.85	-0.03	Market	0.014	2.87	0.01
Market	0.013	2.87	0.01	Road	0.013	2.56	-0.03
Lot	0.013	2.97	0	Restaurant	0.013	2.77	0
Restaurant	0.012	2.77	0	Stall	0.012	2.75	0
Stall	0.012	2.75	0	Taxi	0.011	2.28	-0.03

ตารางที่ 4.31 ค่าความเด่นและความแพร่หลาย

ความแพร่หลาย / ความเด่น	ความแพร่หลายสูง	ความแพร่หลายต่ำ
ความเด่นสูง	นักท่องเที่ยวให้ความสนใจมากและมีความรู้สึกเชิงบวก	นักท่องเที่ยวให้ความสนใจมากแต่มีความรู้สึกเชิงลบ
ความเด่นต่ำ	นักท่องเที่ยวให้ความสนใจน้อยแต่มีความรู้สึกเชิงบวก	นักท่องเที่ยวให้ความสนใจน้อยและมีความรู้สึกเชิงลบ

จากตารางที่ 4.31 สามารถทำให้เราสามารถวิเคราะห์หาจุดเด่นหรือจุดด้อยของสถานที่ท่องเที่ยวได้ โดยนำมาสร้างกราฟระหว่างความเด่น (Salience) และความแพร่หลาย (Valence) ซึ่งแนวตั้งจะแสดงถึงค่าความเด่น และกราฟแนวนอนแสดงถึงค่าความแพร่หลาย ส่วนขนาดของฟองสบู่จะแสดงถึงค่าน้ำหนักของแต่ละคำศัพท์ ทำให้สามารถสร้างแผนภาพฟองสบู่ที่แบ่งได้เป็น 4 จตุภาค (Quadrant) ดังรูปที่ 4.2

รูปที่ 4.2 แผนภาพฟองสบู่ของการวิเคราะห์ตวามเด่นและความแพร่หลาย

จากรูปที่ 4.2 ในเชิงคำศัพท์ในหลายมิติ และนำเสนอคำศัพท์เฉพาะทั้ง 10 คำศัพท์แรกของแต่ละกลุ่มความสนใจในแต่ละจตุภาค ตามหลักการของ Taecharungroj and Mathayomchan (2019) ซึ่งจะเป็นการอธิบายเชิงจตุภาคในภาพรวมของทุกกลุ่มความสนใจ

ในแต่ละจตุภาคจะบ่งบอกลักษณะข้อมูลความสนใจที่แตกต่างกันซึ่งเราจะวิเคราะห์ความหมายของแต่ละจตุภาคที่บ่งบอกถึงข้อเด่นข้อด้อยของเยาวราชได้ดี ซึ่งในที่นี้คำที่เป็นสีฟ้าแสดงถึงกลุ่มแหล่งช้อปปิ้ง สีชมพูแสดงถึงกลุ่มตลาดอาหารริมทางยามค่ำ สีเขียวแสดงถึงกลุ่มอาหาร และสี

ส้มแสดงถึงกลุ่มการเที่ยวชมเมือง อย่างไรก็ตามเนื่องจากในแต่ละกลุ่มความสนใจมีคำศัพท์ซ้ำในหลายกลุ่มทำให้อาจมีการแสดงสีที่แตกต่างออกไปจาก 4 สีข้างต้น เมื่อทำการวิเคราะห์ได้ผลดังนี้

จตุภาคที่ 1 คือ จตุภาคที่มีความเด่นและความแพร่หลายสูงซึ่งบ่งบอกถึงสิ่งที่โดดเด่นและน่าหลงไหลของเยาวราช โดยในจตุภาคนี้ประกอบไปด้วยคำศัพท์ 6 คำซึ่งมาจากทั้ง 4 กลุ่มความสนใจดังต่อไปนี้ เมื่อเรียงตามค่านักหนักแล้วคำศัพท์ที่มาจากกลุ่มความสนใจแหล่งช้อปปิ้งได้แก่ “shop” “good” และ “food” ที่มาจากกลุ่มความสนใจตลาดอาหารริมทางยามค่ำได้แก่ “food” “market” “shop” และ “go” คำศัพท์ที่มาจากกลุ่มความสนใจอาหารได้แก่ “food” “shop” “great” “good” และ “market” คำศัพท์ที่มาจากกลุ่มความสนใจการเที่ยวชมเมืองได้แก่ “shop” “food” “good” “great” และ “market”

จากการตีความกลุ่มความสนใจอาหารกับข้อมูลที่นักท่องเที่ยววิจารณ์เช่น “The sights, sounds and smell of Bangkok’s Chinatown are an assault to the senses. For anyone with a sense of adventure, a day lost among the many market alleys and street food vendors can be the most memorable of any spent in Bangkok.” สามารถตีความได้ว่าถนนเยาวราชมีมีความโดดเด่นและน่าชื่นชมในเรื่องนี้เนื่องจาก ถนนเยาวราชเป็นแหล่งขายอาหารที่สำคัญโดยเฉพาะอาหารริมทาง และมีตลาดที่ขายของหลากหลาย อาหารริมทางต่างๆ ในราคาที่เหมาะสม อีกทั้งยังมีแหล่งช้อปปิ้งที่จำหน่ายสินค้าต่างๆ ทำให้สิ่งเหล่านี้เป็นสิ่งที่น่าชื่นชมของถนนเยาวราช

จตุภาคที่ 2 คือ จตุภาคที่มีความเด่นสูงแต่มีความแพร่หลายต่ำ ในจตุภาคนี้บ่งบอกถึงนักท่องเที่ยวหลายคนที่กล่าวถึงเยาวราชในแง่ที่ไม่พึงพอใจต่อถนนเยาวราช เนื่องจากการที่มีความเด่นสูงหมายถึงสิ่งที่นักท่องเที่ยวกล่าวถึงบ่อย และเมื่อรวมกับความแพร่หลายต่ำซึ่งหมายถึงเป็นสิ่งที่มีความรู้สึกเชิงลบอย่างไรก็ตามในจตุภาคนี้มีเพียงศัพท์คำเดียวคือคำว่า “town” ที่มาจากกลุ่มอาหาร และ ตลาดอาหารริมทางยามค่ำซึ่งบ่งบอกถึงการที่นักท่องเที่ยวมีความรู้สึกที่ไม่ค่อยพอใจนักกับตัวเมืองเยาวราชด้วยค่าความแพร่หลาย -0.3

ตัวอย่างของบทวิจารณ์เชิงลบที่อยู่ในกลุ่มตลาดอาหารริมทางยามค่ำที่กล่าวถึงตัวเมืองของเยาวราชมีดังนี้ “Disappointed!!! Worth going to visit. The lights, the busy streets are cool, but I prefer cleaner small town areas.” หรือ “chinatown? messy town and smelly town!! nothing much to expect from this chinatown. Lots of confusion, loads of smelly street stalls selling everything is also on sale in any street market, except from strange and smelly food that no one dares to buy.” พบว่าตัวเมืองของเยาวราชมีความวุ่นวาย มีสิ่งสกปรกและกลิ่นอันไม่พึงประสงค์ซึ่งเป็นสิ่งที่นักท่องเที่ยวให้ความรู้สึกเชิงลบ

จตุภาคที่ 3 คือ จตุภาคที่มีความเด่นและความแพร่หลายต่ำ ซึ่งบ่งบอกถึงสิ่งที่นักท่องเที่ยวไม่พอใจแต่ไม่ได้เป็นที่พูดถึงกันมาก โดยคำศัพท์ที่เกิดขึ้นในจตุภาคนี้ประกอบด้วย 5 คำได้แก่ “road” “taxi” ที่มาจากกลุ่มเยี่ยมชมเมือง

ตัวอย่างบทวิจารณ์ที่อยู่ในกลุ่มมีดังนี้ “I visited on Sunday morning. Took a taxi from Central World area for about B80. Most of the taxi drivers refused to use meter or even refused to go there. They quoted B200 or more.” จากการพิจารณาคำและบทวิจารณ์ตัวอย่าง พบว่าแท็กซี่ที่เก็บราคาค่าโดยสารไม่เหมาะสมกับระยะทางเป็นสิ่งที่นักท่องเที่ยวมีความรู้สึกเชิงลบ คำศัพท์ “cheap” “area” ที่มาจากกกลุ่มแหล่งช้อปปิ้ง ซึ่งเมื่อวิเคราะห์ประกอบกับบทวิจารณ์เช่น “My family and I went to China town and it was dirty and overcrowded. People weren't as friendly. It was mostly food, toys and clothes. We didn't end up buying much because most of it looked like cheap junk.” พบว่าราคาของที่ถูกแต่บางครั้งเป็นของปลอมที่ไม่มีคุณภาพ และสุดท้ายคำศัพท์ “night” ที่มาจากกกลุ่มตลาดอาหารริมทางยามค่ำ ซึ่งสามารถตีความได้ว่านักท่องเที่ยวส่วนหนึ่งไม่ค่อยชอบใจกับ เยาวราชยามค่ำคืน ซึ่งจากข้อมูลที่นักท่องเที่ยวท่านหนึ่งวิจารณ์ไว้ว่า “My partner made the mistake of booking our last 9 days here and it's probably the biggest mistake of our trip, every day we have to leave to somewhere else as there is no where nice to eat, chaotic no room to walk on the footpath you have to walk on the road and dodge the cars and no shopping.” ซึ่งแสดงให้เห็นถึงความแออัดยามค่ำคืนของเยาวราชทำให้นักท่องเที่ยวไม่มีที่ในการเดินทำให้ต้องเดินที่ถนนแทน

จตุภาคที่ 4 คือจตุภาคที่มีความเด่นต่ำแต่ความแพร่หลายสูง มีความหมายว่าคำศัพท์ที่อยู่ในจตุภาคนี้เป็นเสน่ห์ของเยาวราชแต่นักท่องเที่ยวชาวต่างชาติไม่ค่อยพูดถึงกันซึ่งมีเพียงคำศัพท์เดียวคือคำว่า “gold” จากกลุ่มแหล่งช้อปปิ้ง ซึ่งสามารถตีความได้ว่านักท่องเที่ยวมีรู้สึกในแง่บวกกับทองคำของเยาวราช

จากการอ้างอิงจากข้อมูลตัวอย่างบทวิจารณ์ของนักท่องเที่ยว “We spent a couple of hours wandering through Chinatown. There was a 3 day food festival on and we were able to sample many different types of foods. There are many gold shops, restaurants and general shops to see. A great place to buy gold” พบว่าเยาวราชนั้นเป็นสถานที่ที่เหมาะแก่การซื้อทองคำ ยังไม่เป็นที่แพร่หลายในชาวต่างชาติ

4.6 ผลลัพธ์การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก

จากการนำบทวิจารณ์มาตัดแบ่งเป็นประโยค และนำมาวิเคราะห์ด้วยการวิเคราะห์ความรู้สึกโดยใช้การตัดสินใจของโมเดลการตัดสินใจร่วมกันแบบเสียงข้างมาก ซึ่งทำให้ได้คำศัพท์ที่เป็นลักษณะเฉพาะของแต่ละกลุ่มความสนใจ Liu and Zhang (2012) ซึ่งสามารถแบ่งได้เป็นกลุ่มความสนใจของสิ่งที่นักท่องเที่ยวมีความรู้สึกเชิงบวก และสิ่งที่นักท่องเที่ยวมีความรู้สึกเชิงลบ ดังตารางที่ 4.32

ตารางที่ 4.32 ผลลัพธ์การตัวแทนของคำที่แทนความรู้สึก

กลุ่มความสนใจ	ความรู้สึก	คำศัพท์ที่ผ่านการพิจารณาจากผู้เชี่ยวชาญทางธุรกิจ
แหล่งช้อปปิ้ง	POS	accessory, product, toy, gift, jewelry, haggle, decor, barter, merchandise, shirt, fashion, bargain, chestnut, herb, inexpensive, cheap, retail, trinket, package, gaud, goldsmith, pickle, merchant, ornament, dozen, aroma, inexpensive, cheap
	NEG	quantity, copy, jam, pigeon, junk, worthless, perplex, grimily, trotte, odor, polit, cockroach, trap, rude, stink,
ตลาดอาหารริมทางยามค่ำ	POS	fruit, bird, soup, nest, shark, tasty, noodle, snack, pork, duck, cuisine, dessert, soak, bargain, beer, durian, teochew, rice, chestnut, rice, lobster, chicken, mandarin, crab, flame, dozen, pricey, friendly
	NEG	poor, cockroach, assault, reluct, stink, overflow, dirtiest, sewer, wastewater, noisy, complaint, fear
อาหาร	POS	chicken, desert, kitchen, pork, crab, lobster, priceless, bargain
	NEG	filthiness, sewer, smoke, joke, huddle, crappy, trash, clutter, dog, cat
การเที่ยวชมเมือง	POS	atmosphere, kid, hotel, fascinate, alleyway, wander, sidewalk, hawker
	NEG	smelly, multitude, plight, struggle, complaint, environ, sewer, dirtiest

จากผลการวิเคราะห์ความรู้สึกของนักท่องเที่ยวที่มีต่อสิ่งต่างๆ ทั้งเชิงบวก และเชิงลบที่อยู่ในแต่กลุ่มความสนใจสามารถแสดงได้ดังตารางที่ 4.32 สามารถสรุปได้ดังนี้

1. กลุ่มความสนใจแหล่งช้อปปิ้ง นักท่องเที่ยวมีความรู้สึกที่ดีต่อสินค้าประเภทต่างๆ ได้แก่ เครื่องประดับ สินค้าตกแต่ง สมุนไพร และสินค้าที่มีลักษณะขายส่ง ซึ่งข้อดีของสินค้าต่างๆ เหล่านี้คือสามารถทำการต่อรองราคาได้กับผู้ขายได้ อย่างไรก็ตามสิ่งที่นักท่องเที่ยวไม่พอใจต่อกลุ่มความสนใจนี้คือปัญหาเรื่องสินค้าปลอม และสินค้าคุณภาพต่ำอีกทั้งสถานที่เดินเลือกซื้อสินค้ามีกลิ่นอันไม่พึงประสงค์และความสกปรกจากขยะและสัตว์ต่างๆ เช่น นกพิราบ และแมลงสาป
2. กลุ่มความสนใจตลาดอาหารริมทางยามค่ำ นักท่องเที่ยวมีความรู้สึกที่ดีต่ออาหารริมทางที่จำหน่ายที่ถนนเยาวราชช่วงเวลาค่ำโดยสิ่งที่นักท่องเที่ยวชื่นชอบได่แก่ ผลไม้ต่างๆ หูฉลาม รังนก ก๋วยเตี๋ยว อาหารดอง และอาหารทะเลซึ่งสามารถต่อรองราคาค่าอาหารได้ โดยสิ่งที่นักท่องเที่ยวไม่ชอบในกลุ่มความสนใจนี้จะเกี่ยวกับความสกปรกจากแมลงสาป กลิ่นไม่พึงประสงค์ต่างๆ จากท่อน้ำเสียและสิ่งรบกวน
3. กลุ่มความสนใจอาหารนักท่องเที่ยวมีความรู้สึกที่ดีต่ออาหารต่างๆ คล้ายกับในกลุ่มของตลาดอาหารริมทางยามค่ำซึ่งจะมีอาหารเพิ่มเติมที่นักท่องเที่ยวชื่นชอบเพิ่มเติมคืออาหารจำพวกแมลงทอด โดยสิ่งที่นักท่องเที่ยวไม่พึงพอใจประกอบด้วย ความสกปรกและสัตว์ชนิดต่างๆ เช่น สุนัข และแมว
4. กลุ่มความสนใจที่เกี่ยวกับการเที่ยวชมเมืองของเยาวราชนักท่องเที่ยวชื่นชอบเกี่ยวกับทัศนียภาพของคนที่หาบเร่ขายของ และบรรยากาศของความเป็นจีนในตัวเมือง โดยสิ่งที่นักท่องเที่ยวไม่พอใจคือความสกปรก และท่อน้ำในตัวเมือง

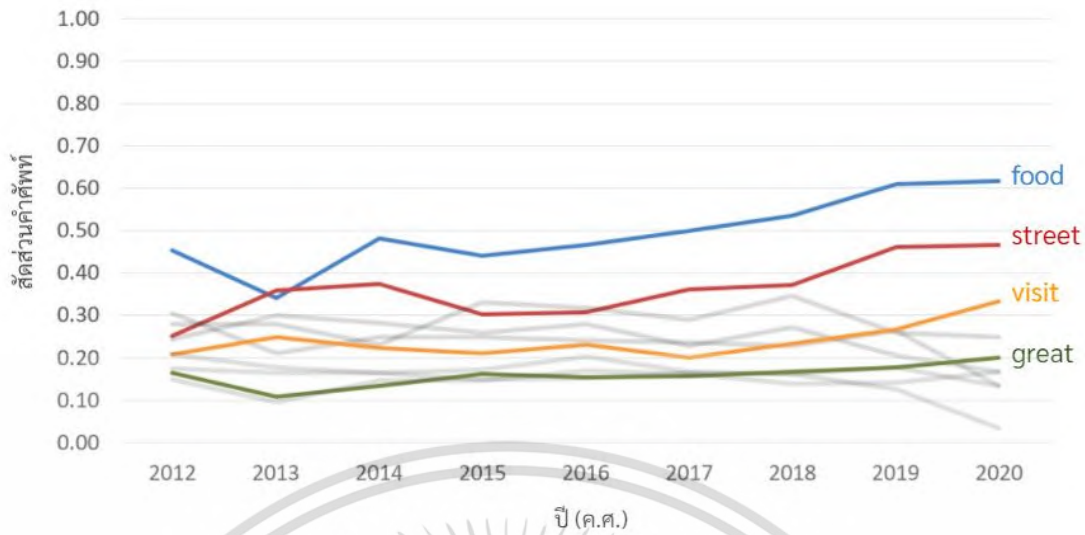
4.7 ผลลัพธ์การวิเคราะห์แนวโน้มของคำที่นักท่องเที่ยวใช้ในบทวิจารณ์

เมื่อนำคำศัพท์ของคำที่เกิดขึ้นบ่อยที่สุด 10 คำจากทุกกลุ่มได้แก่ “food” “street” “place” “shop” “good” “visit” “walk” “market” “great” “town” มาทำการหาอัตราส่วนระหว่างจำนวนบทวิจารณ์ที่พบคำศัพท์ต่อจำนวนบทวิจารณ์ในแต่ละปี ผลลัพธ์สามารถแสดงได้ดังตารางที่ 4.33

ตารางที่ 4.33 จำนวนบทวิจารณ์มีคำศัพท์ในแต่ละปี

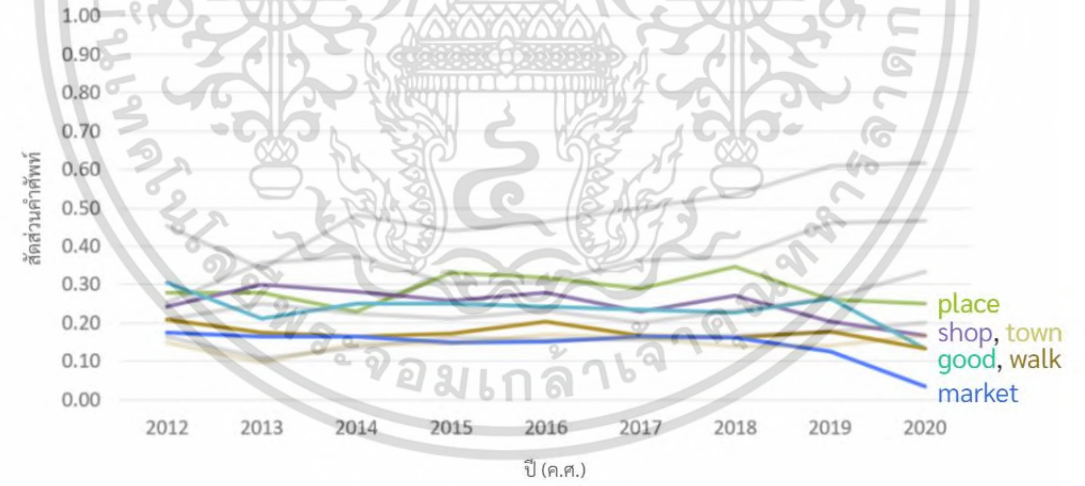
คำศัพท์	ปี (คศ.)								
	2012	2013	2014	2015	2016	2017	2018	2019	2020
food	0.45	0.34	0.48	0.44	0.46	0.50	0.53	0.61	0.62
street	0.25	0.36	0.37	0.30	0.31	0.36	0.37	0.46	0.47
place	0.28	0.28	0.23	0.33	0.32	0.29	0.35	0.26	0.25
shop	0.24	0.30	0.28	0.26	0.28	0.23	0.27	0.20	0.17
good	0.30	0.21	0.25	0.25	0.24	0.24	0.23	0.26	0.13
visit	0.21	0.25	0.22	0.21	0.23	0.20	0.23	0.27	0.33
walk	0.21	0.18	0.17	0.17	0.20	0.17	0.16	0.18	0.13
market	0.17	0.16	0.17	0.15	0.15	0.16	0.16	0.13	0.03
great	0.17	0.11	0.13	0.16	0.16	0.16	0.17	0.18	0.20
town	0.15	0.10	0.15	0.15	0.17	0.16	0.14	0.14	0.17
จำนวนบทวิจารณ์ในแต่ละปี	115	176	260	625	914	916	584	342	60

เมื่อทำการเปรียบเทียบคำศัพท์ในแต่ละปีพบว่าในภาพรวมคำว่า “food” “street” “visit” “great” มีแนวโน้มที่เพิ่มมากขึ้นเมื่อเทียบกับจำนวนบทวิจารณ์ที่มีดังรูปที่ 4.3 ซึ่งการที่นักท่องเที่ยวมีอัตราส่วนการใช้คำศัพท์เหล่านี้มากขึ้น อีกทั้ง “food” “street” เป็นคำที่พบเจอมากกว่าคำอื่นในแทบทุกปีและมีแนวโน้มเพิ่มขึ้น ซึ่งแสดงให้เห็นถึงการที่นักท่องเที่ยวมีความสนใจในสิ่งที่เกี่ยวกับอาหาร อาหารริมทาง และการเยี่ยมชมเยาวราชเพิ่มมากขึ้น



รูปที่ 4.3 กราฟแนวโน้มของคำศัพท์ที่มีความชันเป็นบวก

ตรงกันข้ามกับ “place” “shop” “good” “walk” “market” “town” ที่มีแนวโน้มลดลง ซึ่งแสดงให้เห็นว่านักท่องเที่ยวมีการกล่าวถึงคำพวกนี้ในบทวิจารณ์ลดน้อยลง ทำให้สรุปได้ว่านักท่องเที่ยวมีความสนใจเกี่ยวกับการช้อปปิ้งร้านค้าต่างๆ การเดินเที่ยวชม และตลาด มีแนวโน้มลดน้อยลงในช่วงปีค.ศ. 2018 เป็นต้นไปดังแสดงในรูปที่ 4.4

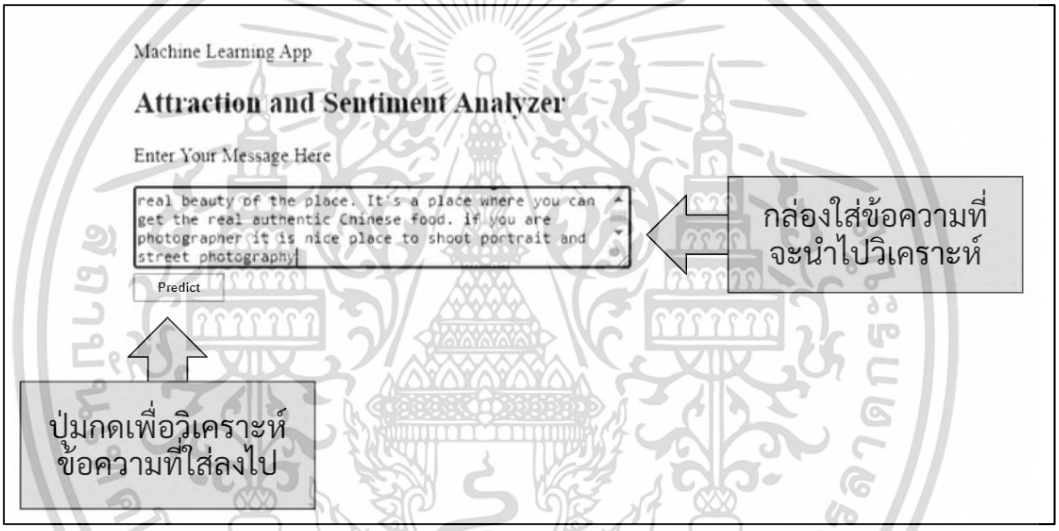


รูปที่ 4.4 กราฟแนวโน้มของคำศัพท์ที่มีความชันเป็นลบ

4.8 การนำโมเดลไปประยุกต์ใช้งาน

ผลจากการวิเคราะห์กลุ่มความสนใจและความรู้สึกจะถูกนำไปพัฒนาเป็นเครื่องมือในรูปแบบเว็ปแอพพลิเคชั่นตัวอย่างสำหรับวิเคราะห์กลุ่มความสนใจและความรู้สึกของนักท่องเที่ยวตามวัตถุประสงค์ข้อที่ 4 โดยเว็ปแอพพลิเคชั่นในส่วนนี้จะเป็นเครื่องมือที่จะช่วยในการวิเคราะห์บทจารณ์ได้อย่างอัตโนมัติ โดยผู้ที่ต้องการใช้งานต้องติดตั้งโปรแกรมด็อกเกอร์

(https://www.docker.com) มาก่อน เพื่อใช้ในการจำลองระบบเว็ปแอพพลิเคชั่นขึ้นมาจากนั้น ผู้ใช้งานสามารถเข้าไปที่เว็ปไซต์ http://127.0.0.1:5000/ เพื่อทำการเริ่มใช้งานแอพพลิเคชั่นแบบออฟไลน์ผ่านเว็ปบราวเซอร์ โดยหน้าเว็ปแอพพลิเคชั่นจะประกอบไปด้วย 2 ส่วนได้แก่กล่องใส่ข้อความ ซึ่งเป็นกล่องที่เอาไว้สำหรับใส่บทวิจารณ์ภาษาอังกฤษที่ต้องการวิเคราะห์ และปุ่ม predict ซึ่งเป็นปุ่มที่ทำการวิเคราะห์ข้อความที่ใส่ลงไปดังรูปที่ 4.5 ซึ่งในที่นี้จะยกตัวอย่างการใช้ประโยค “Visit china town after 6pm in the evening to see the real beauty of the place. It's a place where you can get the real authentic Chinese food. if you are photographer it is nice place to shoot portrait and street photography.” เป็นประโยคตัวอย่างที่จะใช้ในการวิเคราะห์



รูปที่ 4.5 ลักษณะของหน้าเว็ปแอพพลิเคชัน

หลังจากกดปุ่มวิเคราะห์แล้วผลลัพธ์ที่แสดงจะมี 2 ส่วนได้แก่ ผลลัพธ์ของการพยากรณ์กำหนดกลุ่มความสนใจ ซึ่งจากประโยคตัวอย่างข้างต้นพบว่านักท่องเที่ยวที่เขียนบทวิจารณ์ดังกล่าวมีอยู่ในกลุ่มความสนใจด้านอาหาร และผลลัพธ์ความรู้สึกของผู้เขียนบทวิจารณ์เป็นบวกซึ่งแสดงถึงความรู้สึกเชิงบวกต่อเยาวราชดังรูปที่ 4.6

รูปที่ 4.6 ผลลัพธ์ที่แสดงหลังจากทำการวิเคราะห์บทวิจารณ์

โดยแอพพลิเคชั่นตัวนี้สามารถนำไปวิเคราะห์บทวิจารณ์ที่เกี่ยวข้องกับเยาวราชจากแหล่งข้อมูลอื่นได้แม้บทวิจารณ์จะไม่ได้มาจากเว็ป Tripadvisor ก็ตามซึ่งทำให้นักวิเคราะห์ข้อมูลทั่วไปสามารถวิเคราะห์บทวิจารณ์และนำผลไปใช้วิเคราะห์ต่อยอดได้โดยสะดวกโดยไม่ต้องเขียนภาษาไพธอนขึ้นมาเอง

อย่างไรก็ตามเว็ปแอพพลิเคชั่นนี้ผู้ใช้ภายนอกยังไม่สามารถเข้าถึงได้เนื่องจากในงานวิจัยนี้ไม่ได้สร้างระบบการเข้าถึงของผู้ใช้ภายนอกแบบออนไลน์ไว้ทำให้ผู้ที่ใช้งานแอพพลิเคชั่นตัวนี้ต้องใช้งานแบบออฟไลน์ในเครื่องคอมพิวเตอร์ของตนเท่านั้นซึ่งเป็นข้อจำกัดในงานวิจัยชิ้นนี้

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

ในงานวิจัยนี้สามารถสรุป ผลการวิจัย และข้อเสนอแนะต่างๆ ได้ดังนี้

5.1 สรุปผลการวิจัย

จากบทวิจารณ์ทั้งหมด 3,992 บทวิจารณ์ที่รวบรวมมาจากเว็ปไซต์ TripAdvisor เมื่อนำว่าวิเคราะห์หาข้อมูลเชิงลึกแล้วทำให้สามารถสรุปได้ดังต่อนี้

ส่วนที่ 1 การจัดกลุ่มความสนใจของนักท่องเที่ยว เมื่อนำข้อมูลมาทำการจัดกลุ่มข้อมูลแบบเคมีนด้วยการกำหนดกลุ่มด้วยวิธีการหักศอกเพื่อนำไปแบ่งกลุ่มนักท่องเที่ยวด้วยโมเดลการจัดสรรหัวข้อแฝงพบว่านักท่องเที่ยวที่เดินทางมายังเยาวราชสามารถแบ่งออกได้เป็น 4 กลุ่มความสนใจเนื่องจากกราฟของค่าความคลาดเคลื่อนกับจำนวนกลุ่มมีลักษณะเป็นช่วงหักศอกที่แบ่งเป็น 4 กลุ่มโดยจากการพิจารณาคำที่เกิดขึ้นในแต่ละกลุ่มความสนใจประกอบไปด้วยกลุ่มความสนใจแหล่งช้อปปิ้ง (Shopping Place) กลุ่มความสนใจตลาดอาหารริมทางยามค่ำ (Night Street Food Market) กลุ่มความสนใจอาหาร (Food) และกลุ่มความสนใจการเที่ยวชมเมือง (Sightseeing)

ส่วนที่ 2 เป็นส่วนของการวิเคราะห์ความรู้สึกของนักท่องเที่ยว ซึ่งได้มีการใช้เทคนิค SMOTE เพื่อช่วยทำให้ประเภทของข้อมูลมีจำนวนเท่ากันทำให้ บทวิจารณ์ ทำให้บทวิจารณ์ประเภทเชิงบวกและเชิงลบมีจำนวนเท่ากัน จากนั้นจะแบ่งข้อมูลออกเป็น 2 ส่วนได้แก่ส่วนที่ใช้สำหรับฝึกสอน 70% หรือ 4,119 บทวิจารณ์ และสำหรับทดสอบ 30% หรือ 1,765 บทวิจารณ์ โดยใช้ตัวชี้วัด 3 ชนิดในการวัดประสิทธิภาพของโมเดลได้แก่ ค่าความถูกต้อง (Accuracy) ค่าความระลึก (Precision) และค่าความแม่นยำ (Recall) จากผลการศึกษาโมเดลต่างๆ ทั้งหมด 3 โมแดลหลักได้แก่ นาอีฟเบย์ การถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีน ซึ่งจากการทดสอบโมเดลต่างๆ ด้วยชุดข้อมูลทดสอบ 30% สามารถสรุปได้ดังนี้

1. นาอีฟเบย์ มีค่าความถูกต้องเท่ากับ 80.45% ค่าความระลึกเท่ากับ 80.75% ค่าความแม่นยำเท่ากับ 80.29%
2. การถดถอยเชิงโลจิสติค มีค่าความถูกต้องเท่ากับ 83.40% ค่าความระลึกเท่ากับ 83.85% ค่าความแม่นยำเท่ากับ 83.00%

3. ซัพพอร์ตเวกเตอร์แมชชีนที่มีประสิทธิภาพสูงที่สุดคือ ซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF มีค่าความถูกต้องเท่ากับ 83.90% ค่าความระลึกเท่ากับ 84.16% ค่าความแม่นยำเท่ากับ 83.78%

ในส่วนของการเพิ่มประสิทธิภาพของโมเดลต่างๆ ถดถอยเชิงโลจิสติค และซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF จะถูกเพิ่มประสิทธิภาพด้วยการเรกูราไลซ์ สำหรับการถดถอยเชิงโลจิสติคจะมีการปรับเปลี่ยนพารามิเตอร์ความรุนแรงของการเรกูราไลซ์ C พร้อมการใส่พจน์บทลงโทษแบบ Ridge และ Lasso ลงไปเพื่อเพิ่มค่าความถูกต้องสำหรับซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF จะมีการปรับ C และ γ ซึ่งเป็นค่าความสัมพันธ์ของความอคติ และความแปรปรวน โดยหลังจากเพิ่มประสิทธิภาของโมเดลทั้งแล้วพบว่าการถดถอยเชิงโลจิสติคด้วยเรกูราไลซ์แบบ Ridge ให้ค่าความถูกต้องมากที่สุดอยู่ที่ 84.12% ค่าความระลึกเท่ากับ 83.88% ค่าความแม่นยำเท่ากับ 83.56% สำหรับซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF มีค่าความถูกต้องมากที่สุดอยู่ที่ 84.46% ค่าความระลึกเท่ากับ 85.63% ค่าความแม่นยำเท่ากับ 83.56%

เมื่อนำนาอีฟเบย์ การถดถอยเชิงโลจิสติคด้วยเรกูราไลซ์แบบ Ridge และซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นด้วยฟังก์ชันเคอร์เนลแบบ RBF มาทำการตัดสินใจร่วมกันแบบเสียงข้างมากทำให้ผลลัพธ์ค่าความถูกต้องเท่ากับ 88.62% ค่าความระลึกเท่ากับ 87.41% และค่าความแม่นยำเท่ากับ 86.32%

ส่วนที่ 3 ผลการวิเคราะห์เพื่อให้เกิดความเข้าใจของบทวิจารณ์เชิงลึกซึ่งจะเป็นการทดลองตามจุดประสงค์ข้อที่ 3 ซึ่งมีทั้งหมด 4 วิธีได้แก่ การวิเคราะห์ความนิยม การวิเคราะห์ความเด่นและความแพร่หลาย การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก และการวิเคราะห์แนวโน้มของคำ

1. ผลการวิเคราะห์บทวิจารณ์เชิงลึกของเยาวราชด้วยวิธีวิเคราะห์ความนิยมซึ่งเป็นการนำคะแนนของนักท่องเที่ยวมาเฉลี่ยพบว่า คะแนนเฉลี่ยของแต่ละกลุ่มมีความแตกต่างกันไม่มาก จากคะแนนเต็มทั้งหมด 5 คะแนน เมื่อเรียงคะแนนเฉลี่ยจากมากไปน้อยพบว่า กลุ่มอาหาร (Food) มีคะแนนเฉลี่ย 4.02 คะแนน กลุ่มแหล่งช้อปปิ้ง (Shopping Place) มีคะแนนเฉลี่ย 3.98 คะแนน กลุ่มการเที่ยวชมเมือง (Sightseeing) มีคะแนนเฉลี่ย 3.96 และกลุ่มตลาดอาหารริมทางยามค่ำ (Night Street Food Market) 3.95 คะแนน โดยคะแนนเฉลี่ยนรวมของเยาวราชคือ 3.98 คะแนน

2. ผลการวิเคราะห์เพิ่มเติมของบทวิจารณ์เชิงลึกของเยาวราชด้วยวิธีการวิเคราะห์ความเด่นและความแพร่หลาย จะเป็นการหาภาพรวมข้อดีข้อเสียของเยาวราชในรูปของจตุภาค โดยพบว่า ถนนเยาวราชมีมีความโดดเด่น และน่าชื่นชมในเรื่องแหล่งขายอาหารที่สำคัญ โดยเฉพาะอาหารริมทาง และเครื่องประดับ อย่างไรก็ตามเยาวราชยังมีข้อด้อยในเรื่องของถนนทางเท้าของเยาวราชซึ่งมีลักษณะที่เป็นหลุมเป็นบ่อ สินค้าเลียนแบบซึ่งเป็นของที่มีคุณภาพต่ำ และจำนวนของผู้คนที่สัญจรยามค่ำมีความหนาแน่นมากเกินไปทำให้ไม่สะดวกต่อการเดินเที่ยวชม
3. ผลการวิเคราะห์ข้อมูลเชิงลึกของเยาวราชด้วยวิธีการวิเคราะห์ตัวแทนของคำที่แทนความรู้สึกเป็นการนำโมเดลการตัดสินใจร่วมมาใช้ในการจำแนกความรู้สึกของนักท่องเที่ยวจากประโยคเพื่อพิจารณาคำที่อยู่ในเชิงบวก และเชิงลบของแต่ละกลุ่มความสนใจโดยพบว่าในภาพรวมนักท่องเที่ยวมีความรู้สึกเชิงบวกเกี่ยวกับด้านอาหาร ผลไม้ เครื่องประดับ ทัศนียภาพที่เกี่ยวกับคนหาบเร่ขายของ ในทางกลับกันในทางความรู้สึกเชิงลบเป็นในเรื่องของความสกปรกของท้องถนน เช่นสุนัข ท่อน้ำ และกลิ่นอันไม่พึงประสงค์
4. ผลการวิเคราะห์ในส่วนของการวิเคราะห์แนวโน้มของคำจะเป็นการนำคำศัพท์ที่เกิดขึ้นถี่ที่สุด 10 คำแรกมาหาสัดส่วนเทียบกับจำนวนที่เกิดขึ้นทั้งปีพบว่า นักท่องเที่ยวมีแนวโน้มการใช้คำว่า “food” กับ “street” เพิ่มขึ้น ซึ่งแสดงให้เห็นถึงการที่นักท่องเที่ยวมีความสนใจในสิ่งที่เกี่ยวกับ อาหาร อาหารริมทาง เพิ่มมากขึ้นอย่างไรก็ตามนักท่องเที่ยใช้คำศัพท์ “place” “shop” “good” “walk” “market” “town” เหล่านี้ลดลงซึ่งแสดงให้เห็นถึงการช้อปปิ้งร้านค้าต่างๆ การเดินเที่ยวชม และตลาดมีความสนใจลดน้อยลง

ส่วนที่ 4 เป็นการสร้างเว็ปแอพพลิเคชั่นซึ่งเป็นตัวอย่างในการประยุกต์โมเดลในการวิเคราะห์ความสนใจตามจุดประสงค์ข้อที่ 4 ของงานวิจัยนี้โดยแอพพลิเคชั่นตัวนี้สามารถใช้วิเคราะห์บทวิจารณ์ต่างๆ ของนักท่องที่ยวที่มาเยี่ยมชมเยาวราชได้โดยเว็ปแอพพลิเคชั่นตัวนี้พัฒนาขึ้นมาจากด๊อกเกอร์และภาษาไพธอน

จากการศึกษาผลลัพธ์และวิธีการต่างๆ 4 ส่วนที่กล่าวไปข้างต้นพบว่าการศึกษานี้สามารถใช้เป็นส่วนหนึ่งในการวางแผนเพื่อรองรับนักท่องเที่ยวหลังสถานการณ์โควิด-19 ได้ เช่นจากการทดลองพบว่านักท่องเที่ยวไม่พอใจเรื่องของกลิ่นอันไม่พึงประสงค์ หรือมีมาตราการปราบปรามเรื่องสินค้า

ปลอมมากขึ้นเพื่อทำให้นักท่องเที่ยวมีความพึงพอใจมากขึ้น เนื่องจากบทวิจารณ์ที่ใช้ในการทดลองเป็นบทวิจารณ์ของนักท่องเที่ยวผู้เดินทางมายังเยาวราชโดยตรงทำให้สามารถตอบสนองนักท่องเที่ยวที่เข้ามาได้ อีกทั้งในงานวิจัยนี้ยังได้มีการสร้างเว็ปแอพลิเคชั่นตัวอย่างซึ่งสามารถนำไปต่อยอดเพื่อใช้ประโยชน์ในมุมของการนำเทคโนโลยีการเรียนรู้ของเครื่องไปใช้ในชีวิตจริงในรูปของเว็ปแอพพลิเคชั่นซึ่งสามารถนำไปต่อยอดได้

5.2 ข้อเสนอแนะ

ในงานวิจัยฉบับนี้เป็นงานวิจัยที่เกี่ยวกับการทำเหมือนข้อความที่ใช้ความคิดเห็นของชาวต่างชาติซึ่งใช้บทวิจารณ์ภาษาอังกฤษ อย่างไรก็ตามปัจจุบันมีเทคโนโลยีการวิเคราะห์ภาษาไทยซึ่งทำให้สามารถเข้าใจข้อมูลเชิงลึกในรูปแบบของภาษาไทยได้ ทำให้ผู้ที่สนใจสามารถนำงานวิจัยนี้ไปเป็นส่วนหนึ่งในการพัฒนาเป็นการทำเหมืองข้อความสำหรับภาษาไทยที่ใช้ในด้านการวิเคราะห์ความรู้สึกที่มีต่อสถานที่ท่องเที่ยวต่างๆ และนำไปใช้กับสถานที่ท่องเที่ยวอื่นๆ ที่มีรูปแบบ และความโดดเด่นด้านกิจกรรมที่หลากหลาย อีกทั้งยังสามารถนำไปใช้ในการเปรียบเทียบสถานที่ท่องเที่ยวที่มีความคล้ายคลึงกันในประเทศต่างๆ ได้

ในด้านของโมเดลการวิเคราะห์บทวิจารณ์ปัจจุบันได้มีเทคโนโลยีการวิเคราะห์เชิงลึก (Deep learning) ที่ช่วยทำให้คอมพิวเตอร์สามารถเรียนรู้คำพ้องความหมาย (Synonyme) ได้ซึ่งจะช่วยทำให้โมเดลมีความฉลาดในการแยกแยะ และจับกลุ่มคำศัพท์ที่มีความหมายคล้ายคลึงกันให้เป็นคำเดียวกันได้ดียิ่งขึ้น

5.3 ข้อจำกัดของงานวิจัย

ข้อจำกัดที่เกิดขึ้นกับงานวิจัยนี้มีดังนี้

1) การกำหนดชื่อกลุ่มความสนใจเป็นการกำหนดจากความคิดเห็นของผู้เชี่ยวชาญซึ่งอาจมีมุมมองไม่สอดคล้องกับผู้อ่าน
2) คะแนนเกิดที่เกิดจากนักท่องเที่ยวเกิดจากความชอบส่วนบุคคลซึ่งนักท่องเที่ยวมีความยากง่ายในการให้แตกต่างกัน
3) เว็ปแอพพลิเคชั่นยังไม่สามารถใช้งานแบบออนไลน์ได้ต้องใช้ในคอมพิวเตอร์ส่วนบุคคลที่ติดตั้งโปรแกรมด็อกเกอร์แล้วเท่านั้น
4) งานวิจัยนี้ไม่ได้มีโมเดลที่ใช้ในการทำความเข้าใจสำนวน (Idiom) คำพ้องความหมาย (Synonym) และคำที่คู่กันของบทวิจารณ์

เอกสารอ้างอิง

สำนักข่าวเศรษฐกิจ. 2561. ‘การท่องเที่ยวฯ เปิดตัวรายงานเศรษฐกิจด้านการท่องเที่ยวรายไตรมาส ฉบับที่ 1/62’ สืบค้นเมื่อ 28 ธันวาคม 2563, จาก https://www.ryt9.com/s/iq03/305894

ชัยวุฒิ ชัยฤกษ์ และเสรี วงษ์มณฑา. 2561. 'ปัจจัยการตัดสินใจของนักท่องเที่ยวที่เดินทางท่องเที่ยวในเขต เศรษฐกิจพิเศษตาก', *Journal of Thai Hospitality and Tourism*, 14: 16-27.

พิชชานันท์ ช่องรักษ์ และ เจริญชัย เอกมาไพศาล. 2561. ความสัมพันธ์เชิงสาเหตุของภาพลักษณ์อาหารริมทางทัศนคติต่อจุดหมายปลายทางการคล้อยตามกลุ่มอ้างอิงและความตั้งใจกลับมาจุดหมายปลายทางซ้ำของนักท่องเที่ยวกรณีศึกษาย่านเยาวราช. *วารสารเศรษฐศาสตร์และนโยบายสาธารณะ*, 9(17), 1-20.

ธนภัทร์ คุ้มสุภา. 2559. 'การจำแนกประเภทข้อความในภาษาไทยโดยใช้ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร', จุฬาลงกรณ์ มหาวิทยาลัย.

วฤษาย์ ร่มสายหยุด, กชกร ณ นครพนม, พิมผกา ประเสริฐศิลป์ และปิยพร นุรารักษ์. 2561. 'รายงานการวิจัยเรื่องการทำเหมืองข้อความแฝงสำหรับการตรวจพบและป้องกันจากอาชญากรรมไซเบอร์', มหาวิทยาลัยสุโขทัยธรรมาธิราช.

เดชธรรม ศิริ และพยุง มีสัจ. 2554. 'การเรียนรู้แบบรวมกลุ่มด้วยโครงข่ายประสาทเทียมเอดาบูทสำหรับการจำแนกข้อมูล', *Information Technology Journal*, 7: 7-12.

อารยา หลงชวน. 2556. 'การเปรียบเทียบประสิทธิภาพของเทคนิคการลดมิติสำหรับข้อมูลเข้าที่ใช้ในวิธีซัพพอร์ตเวกเตอร์แมชชีน: กรณีศึกษาการแจกแจงแบบเกาส์เซียน', จุฬาลงกรณ์มหาวิทยาลัย.

Aggarwal, Charu C. 2020. *Linear Algebra and Optimization for Machine Learning: A Textbook* (Springer Nature).

Bao, Ho Tu. 2012. "Kernel Methods and Support Vector Machines." In.: Japan Advance Institute of Science and Technology.

Bi, Jian-Wu, Yang Liu, Zhi-Ping Fan, and Jin Zhang. 2019. 'Wisdom of crowds: Conducting importance-performance analysis (IPA) through online reviews', *Tourism Management*, 70: 460-78.

Bishop, Christopher M. 2006. Pattern recognition and machine learning (springer).

Bühlmann, Peter, and Sara Van De Geer. 2011. *Statistics for high-dimensional data: methods, theory and applications* (Springer Science & Business Media).

Chairerk, Chaiwut, and Seri Wongmontha. 2019. 'ปัจจัยการตัดสินใจของนักท่องเที่ยวที่เดินทางท่องเที่ยวในเขตเศรษฐกิจพิเศษตาก', *Journal of Thai Hospitality and Tourism*, 14: 16-27.

Chang Liu, and Haoyuan Ning. 2014. 'Exploring the Differences in Destination Branding Toward International and Domestic Tourists', Dalarna University College.

Chawla, N. V., K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer. 2002. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research 16*: 321-357.

Filieri, Raffaele, Salma Alguezaui, and Fraser McLeay. 2015. 'Why do travelers trust TripAdvisor? Antecedents of trust towards consumer-generated media and its influence on recommendation adoption and word of mouth', *Tourism Management*, 51: 174-85.

Géron, Aurélien. 2019. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems* (O'Reilly Media).

Guo, Yue, Stuart J Barnes, and Qiong Jia. 2017. 'Mining meaning from online ratings and reviews: Tourist satisfaction analysis using latent dirichlet allocation', *Tourism Management*, 59: 467-83.

Hapke, Hannes Max, Hobson Lane, and Cole Howard. 2019. "Natural language processing in action." In.: Manning.

Hastie, Trevor, Saharon Rosset, Robert Tibshirani, and Ji Zhu. 2004. 'The entire regularization path for the support vector machine', *Journal of Machine Learning Research*, 5: 1391-415.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction* (Springer Science & Business Media).

Hsu, Chih-wei, Chih-chung Chang, and Chih-Jen Lin. 2003. 'A Practical Guide to Support Vector Classification Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin'.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning* (Springer).

Jiang, Wu, Shaoxin, Hou, Mengmeng Jin, ‘Social Support and User Roles in a Chinese Online Health Community: A LDA Based Text Mining Study’ International Conference on Smart Health ICSH 2017: 169-176

Juan, Alfons, and Hermann Ney. 2002. "Reversing and Smoothing the Multinomial Naive Bayes Text Classifier." In *PRIS*, 200-12.

Khalid, Haider, and Vincent Wade. 2020. 'Topic Detection from Conversational Dialogue Corpus with Parallel Dirichlet Allocation Model and Elbow Method', arXiv preprint arXiv:2006.03353.

Kimeldorf, George, and Grace Wahba. 1971. 'Some results on Tchebycheffian spline functions', *Journal of mathematical analysis and applications*, 33: 82-95.

Kononenko, Igor, and Matjaz Kukar. 2007. *Machine learning and data mining* (Horwood Publishing).

Korovkinas, Konstantinas, and Gintautas Garšva. 2018. "Selection of Intelligent Algorithms for Sentiment Classification Method Creation." In *Proceedings of the International Conference on Information Technologies*, 152-57.

Kuhamanee, T., N. Talmongkol, K. Chaisuriyakul, W. San-Um, N. Pongpisuttinun, and S. Pongyupinpanich. 2017. "Sentiment analysis of foreign tourists to Bangkok using data mining through online social network." In *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 1068-73.

Liu, Bing, and Lei Zhang. 2012. 'A survey of opinion mining and sentiment analysis.' in, *Mining text data* (Springer).

Luis, Serrano. 2020. ‘Latent Dirichlet Allocation (Part 1 of 2) ‘, Accessed 29 December 2020.https://www.youtube.com/watch?v=T05t-SqKArY&list=PLTfPvi-tTL4ScSEt2woJD_DZpUk1p23UN&ab_channel=LuisSerrano.

Mastercards. 2019. 'Master card global destination 2019', Accessed 5 May 2020. https://newsroom.mastercard.com/wp-content/uploads/2019/09/GDCI-Global-Report-FINAL1.pdf

Moguerza, Javier M, and Alberto Muñoz. 2006. 'Support vector machines with applications', *Statistical Science*, 21: 322-36.

Murphy, Kevin P. 2012. *Machine learning: a probabilistic perspective* (MIT press).

Ng, Andrew. 2018. "Machine learning lecture notes." In.: Standford university.

Ng, Andrew Y. 1997. "Preventing" overfitting" of cross-validation data." In *ICML*, 245-53. Citeseer.

Organization, World Tourism. 2011. *Tourism Towards 2030 / Global Overview - Advance edition presented at UNWTO 19th General Assembly - 10 October 2011*.

Patil, Priyanka, and Pratibha Yalagi. 2016. 'Sentiment Analysis Levels and Techniques: A Survey', *space*, 1: 6.

Phienthrakul, Tanasanee. 2008. 'Kernel functions for support vector machines', Chulalongkorn University.

Porter, Martin F. 1980. 'An algorithm for suffix stripping', *Program*, 14: 130-37.

Rodriguez, Juan D, Aritz Perez, and Jose A Lozano. 2009. 'Sensitivity analysis of k-fold cross validation in prediction error estimation', *IEEE transactions on pattern analysis and machine intelligence*, 32: 569-75.

Rosasco, Lorenzo, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. 2004. 'Are loss functions all the same?', *Neural Computation*, 16: 1063-76.

Russell, Stuart, and Peter Norvig. 2002. 'Artificial intelligence: a modern approach'.

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.

Sarkar, D. 2019. Text analytics with Python: a practitioner's guide to natural language processing, Apress.

Phakhawat Sarakit, Thnaruk Theeramunkong and Choochart Haruechaiyasak. 2015. Improving emotion classification in imbalanced YouTube dataset using SMOTE algorithm. 2015 2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), IEEE.

Taecharungroj, Viriya, and Boonyanit Mathayomchan. 2019. 'Analysing TripAdvisor reviews of tourist attractions in Phuket, Thailand', *Tourism Management*, 75: 550-68.

Tibshirani, Ryan. 2009. "Data Mining." In.: Carnegie Mellon University.

YuvinaTileng, Marlin, Wiranto Herry Utomo, and Rudy Latuperissa. 2013. 'Analysis of Service Quality using Servqual Method and Importance Performance Analysis (IPA) in Population Department, Tomohon City', *International Journal of Computer Applications*, 70: 23-30.

Zhang, Ziqiong, Qiang Ye, Rob Law, and Yijun Li. 2010. 'The impact of e-word-of-mouth on the online popularity of restaurants: A comparison of consumer reviews and editor reviews', *International Journal of Hospitality Management*, 29: 694-700.

Zisserman, Andrew. 2015. "C19 Machine Learning lectures." In.: University of oxford.

ภาคผนวก ก

ตาราง ก.1 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-15}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-15}	2^{-15}	0.3080	0.4074	0.2530
	2^{-13}	0.3542	0.3174	0.2982
	2^{-11}	0.6168	0.6844	0.6855
	2^{-9}	0.7115	0.6667	0.6513
	2^{-7}	0.7279	0.6424	0.6642
	2^{-5}	0.7138	0.7476	0.7242
	2^{-3}	0.6548	0.7355	0.6845
	2^{-1}	0.6696	0.5546	0.7010
	2^{1}	0.4538	0.5492	0.4672
	2^{3}	0.4345	0.3387	0.3949
	2^{5}	0.3997	0.3694	0.3196
	2^{7}	0.3579	0.3839	0.5150
	2^{9}	0.3829	0.3856	0.3553
	2^{11}	0.4454	0.2917	0.2654
	2^{13}	0.3878	0.4811	0.4689
	2^{15}	0.3903	0.4208	0.3515

ตาราง ก.2 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-13}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-13}	2^{-15}	0.3788	0.3857	0.2945
	2^{-13}	0.3592	0.3102	0.2677
	2^{-11}	0.6196	0.6882	0.6943
	2^{-9}	0.6606	0.6651	0.7033
	2^{-7}	0.7294	0.6509	0.6756
	2^{-5}	0.7363	0.7428	0.6844
	2^{-3}	0.7148	0.6846	0.6271
	2^{-1}	0.6998	0.5795	0.6322
	2^{1}	0.4964	0.4835	0.4165
	2^{3}	0.4397	0.3015	0.4232
	2^{5}	0.3795	0.3194	0.3462
	2^{7}	0.4114	0.4129	0.4712
	2^{9}	0.4132	0.4117	0.3126
	2^{11}	0.4430	0.2992	0.3584
	2^{13}	0.3517	0.4628	0.4673
	2^{15}	0.3650	0.4289	0.3348

ตาราง ก.3 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-11}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-11}	2^{-15}	0.3935	0.4250	0.2730
	2^{-13}	0.4018	0.3333	0.2389
	2^{-11}	0.5844	0.6325	0.6603
	2^{-9}	0.6603	0.6615	0.6184
	2^{-7}	0.6540	0.6311	0.6597
	2^{-5}	0.6601	0.7381	0.6663
	2^{-3}	0.6772	0.7046	0.6442
	2^{-1}	0.6551	0.5625	0.7043
	2^{1}	0.5025	0.5530	0.3939
	2^{3}	0.3536	0.3387	0.4514
	2^{5}	0.3579	0.3347	0.3660
	2^{7}	0.4399	0.3390	0.5066
	2^{9}	0.3596	0.4063	0.3266
	2^{11}	0.3803	0.2968	0.3508
	2^{13}	0.3801	0.5089	0.4248
	2^{15}	0.3892	0.3724	0.2833

ตาราง ก.4 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-9}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-9}	2^{-15}	0.3681	0.4038	0.2413
	2^{-13}	0.3713	0.2933	0.2757
	2^{-11}	0.6575	0.6517	0.6405
	2^{-9}	0.6502	0.6578	0.6086
	2^{-7}	0.6716	0.6568	0.6786
	2^{-5}	0.6918	0.7067	0.7275
	2^{-3}	0.6426	0.7102	0.6375
	2^{-1}	0.6798	0.5599	0.6777
	2^{1}	0.4558	0.4706	0.3856
	2^{3}	0.3945	0.3994	0.4120
	2^{5}	0.3556	0.3343	0.3609
	2^{7}	0.3614	0.3781	0.4861
	2^{9}	0.4242	0.4535	0.2950
	2^{11}	0.4035	0.2961	0.3373
	2^{13}	0.4358	0.5197	0.4601
	2^{15}	0.3682	0.4185	0.2700

ตาราง ก.5 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-7}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-7}	2^{-15}	0.3083	0.4045	0.2472
	2^{-13}	0.3476	0.3772	0.2912
	2^{-11}	0.6322	0.5929	0.6137
	2^{-9}	0.6214	0.7078	0.6387
	2^{-7}	0.6678	0.6840	0.7123
	2^{-5}	0.6486	0.7135	0.6440
	2^{-3}	0.7041	0.6739	0.7134
	2^{-1}	0.6228	0.6008	0.6855
	2^{1}	0.5400	0.5470	0.4057
	2^{3}	0.4365	0.3005	0.4099
	2^{5}	0.4248	0.3611	0.3698
	2^{7}	0.3668	0.3616	0.5158
	2^{9}	0.3568	0.3995	0.2988
	2^{11}	0.4294	0.2880	0.2953
	2^{13}	0.3805	0.4883	0.4644
	2^{15}	0.3652	0.4030	0.3524

ตาราง ก.6 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-5}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-5}	2^{-15}	0.3314	0.3893	0.2719
	2^{-13}	0.3540	0.3302	0.2554
	2^{-11}	0.6133	0.6186	0.6206
	2^{-9}	0.6997	0.6876	0.6241
	2^{-7}	0.7017	0.6874	0.6308
	2^{-5}	0.7430	0.6957	0.6848
	2^{-3}	0.7148	0.7080	0.7228
	2^{-1}	0.6649	0.6145	0.6811
	2^{1}	0.5378	0.4909	0.3912
	2^{3}	0.3871	0.3163	0.4531
	2^{5}	0.3782	0.3221	0.4020
	2^{7}	0.3719	0.3360	0.4620
	2^{9}	0.4468	0.3903	0.3517
	2^{11}	0.4295	0.3264	0.2970
	2^{13}	0.4367	0.4581	0.4787
	2^{15}	0.4100	0.3933	0.3195

ตาราง ก.7 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-3}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-3}	2^{-15}	0.3849	0.3980	0.2913
	2^{-13}	0.3485	0.3102	0.2145
	2^{-11}	0.6463	0.6679	0.6038
	2^{-9}	0.6570	0.7144	0.6243
	2^{-7}	0.6914	0.6993	0.6555
	2^{-5}	0.7375	0.7347	0.6936
	2^{-3}	0.7099	0.6592	0.7213
	2^{-1}	0.6685	0.5657	0.6621
	2^{1}	0.4852	0.4681	0.3695
	2^{3}	0.3686	0.3249	0.4653
	2^{5}	0.3867	0.3281	0.3700
	2^{7}	0.4282	0.4117	0.5072
	2^{9}	0.3732	0.4135	0.3542
	2^{11}	0.4029	0.2792	0.3167
	2^{13}	0.4330	0.4717	0.4286
	2^{15}	0.3692	0.4270	0.3175

ตาราง ก.8 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{-1}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{-1}	2^{-15}	0.3890	0.3771	0.3137
	2^{-13}	0.3238	0.3064	0.2181
	2^{-11}	0.6001	0.5952	0.6273
	2^{-9}	0.6741	0.6768	0.6662
	2^{-7}	0.7209	0.6402	0.7051
	2^{-5}	0.7113	0.6895	0.6851
	2^{-3}	0.6368	0.6780	0.6300
	2^{-1}	0.6790	0.6055	0.6910
	2^{1}	0.4945	0.5122	0.4114
	2^{3}	0.4301	0.3623	0.4194
	2^{5}	0.4229	0.3184	0.3225
	2^{7}	0.3751	0.3417	0.5302
	2^{9}	0.3685	0.3910	0.2908
	2^{11}	0.4392	0.2697	0.2844
	2^{13}	0.3722	0.5032	0.4537
	2^{15}	0.4207	0.3953	0.3448

ตาราง ก.9 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^1

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^1	2^{-15}	0.3504	0.3746	0.2580
	2^{-13}	0.3772	0.3403	0.2712
	2^{-11}	0.6523	0.6319	0.6603
	2^{-9}	0.6873	0.6862	0.6056
	2^{-7}	0.7354	0.7061	0.7109
	2^{-5}	0.6920	0.6926	0.6432
	2^{-3}	0.7006	0.7385	0.7125
	2^{-1}	0.6678	0.6058	0.6250
	2^{1}	0.4786	0.5388	0.4094
	2^{3}	0.3712	0.3184	0.4040
	2^{5}	0.4036	0.3425	0.3477
	2^{7}	0.4277	0.3788	0.4543
	2^{9}	0.4312	0.4150	0.3470
	2^{11}	0.4142	0.2989	0.3391
	2^{13}	0.4343	0.4668	0.4144
	2^{15}	0.4425	0.4211	0.2877

ตาราง ก.10 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^3

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^3	2^{-15}	0.3282	0.4145	0.2815
	2^{-13}	0.3700	0.3190	0.2334
	2^{-11}	0.5902	0.6599	0.6135
	2^{-9}	0.6775	0.6908	0.6668
	2^{-7}	0.6476	0.6743	0.7067
	2^{-5}	0.6834	0.7031	0.6861
	2^{-3}	0.6594	0.7089	0.6741
	2^{-1}	0.6952	0.5486	0.6347
	2^{1}	0.5117	0.5077	0.4148
	2^{3}	0.3552	0.3156	0.4814
	2^{5}	0.3986	0.3217	0.3355
	2^{7}	0.3917	0.3490	0.4746
	2^{9}	0.4071	0.4650	0.3638
	2^{11}	0.4163	0.3516	0.3282
	2^{13}	0.4401	0.4642	0.4781
	2^{15}	0.4087	0.4238	0.2709

ตาราง ก.11 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^5

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^5	2^{-15}	0.3342	0.3969	0.3062
	2^{-13}	0.3544	0.3266	0.2601
	2^{-11}	0.6781	0.6813	0.6464
	2^{-9}	0.6522	0.6923	0.6634
	2^{-7}	0.6606	0.6309	0.6965
	2^{-5}	0.7142	0.7531	0.6434
	2^{-3}	0.7165	0.6746	0.6624
	2^{-1}	0.6581	0.6284	0.6483
	2^{1}	0.4719	0.4794	0.4421
	2^{3}	0.4127	0.3801	0.4442
	2^{5}	0.3528	0.3173	0.3118
	2^{7}	0.4428	0.4091	0.5309
	2^{9}	0.4167	0.4438	0.3429
	2^{11}	0.4096	0.3167	0.2637
	2^{13}	0.4063	0.4461	0.4428
	2^{15}	0.4023	0.3794	0.2826

ตาราง ก.12 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^7

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^7	2^{-15}	0.3619	0.3568	0.3060
	2^{-13}	0.3971	0.3471	0.2835
	2^{-11}	0.5959	0.6483	0.6335
	2^{-9}	0.6741	0.7016	0.6545
	2^{-7}	0.6482	0.7089	0.7022
	2^{-5}	0.6753	0.7005	0.6482
	2^{-3}	0.6697	0.7450	0.6614
	2^{-1}	0.6929	0.5449	0.6837
	2^{1}	0.4728	0.5106	0.3705
	2^{3}	0.4078	0.3261	0.4912
	2^{5}	0.3988	0.3150	0.3158
	2^{7}	0.3592	0.3987	0.4578
	2^{9}	0.3993	0.4438	0.3311
	2^{11}	0.3593	0.2532	0.2706
	2^{13}	0.4472	0.5254	0.4755
	2^{15}	0.4223	0.4661	0.3520

ตาราง ก.13 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^9

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^9	2^{-15}	0.3641	0.4159	0.2921
	2^{-13}	0.3488	0.3343	0.2635
	2^{-11}	0.6253	0.6832	0.6394
	2^{-9}	0.7141	0.6678	0.6505
	2^{-7}	0.6838	0.7125	0.6631
	2^{-5}	0.6720	0.6989	0.7265
	2^{-3}	0.7051	0.7066	0.6975
	2^{-1}	0.6512	0.5788	0.6900
	2^{1}	0.5136	0.4977	0.3849
	2^{3}	0.3779	0.3808	0.4621
	2^{5}	0.4038	0.3464	0.3644
	2^{7}	0.3709	0.3721	0.4829
	2^{9}	0.3948	0.4037	0.3687
	2^{11}	0.3562	0.3025	0.3269
	2^{13}	0.3874	0.5095	0.3867
	2^{15}	0.4347	0.4383	0.3073

ตาราง ก.14 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{11}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{11}	2^{-15}	0.4008	0.3630	0.2588
	2^{-13}	0.3114	0.3086	0.2842
	2^{-11}	0.6553	0.6606	0.6083
	2^{-9}	0.6743	0.6540	0.6762
	2^{-7}	0.7202	0.7010	0.6286
	2^{-5}	0.6766	0.7113	0.7170
	2^{-3}	0.6612	0.7065	0.6487
	2^{-1}	0.6861	0.6318	0.6920
	2^{1}	0.4442	0.5341	0.3690
	2^{3}	0.3928	0.3956	0.4750
	2^{5}	0.3828	0.3233	0.3872
	2^{7}	0.3707	0.3688	0.4450
	2^{9}	0.3973	0.4487	0.3519
	2^{11}	0.4239	0.2724	0.2707
	2^{13}	0.4168	0.4466	0.4664
	2^{15}	0.4018	0.4543	0.2627

ตาราง ก.15 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{13}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{13}	2^{-15}	0.3924	0.3810	0.2428
	2^{-13}	0.3956	0.3118	0.2574
	2^{-11}	0.6700	0.6271	0.6397
	2^{-9}	0.6677	0.6711	0.6873
	2^{-7}	0.7067	0.6377	0.6622
	2^{-5}	0.6913	0.6681	0.7057
	2^{-3}	0.6473	0.7071	0.6590
	2^{-1}	0.6736	0.5628	0.6599
	2^{1}	0.5129	0.5594	0.3806
	2^{3}	0.4388	0.3284	0.4007
	2^{5}	0.3570	0.3406	0.3982
	2^{7}	0.3921	0.4186	0.5340
	2^{9}	0.4078	0.4486	0.3276
	2^{11}	0.3688	0.3262	0.3304
	2^{13}	0.3702	0.5079	0.4288
	2^{15}	0.4054	0.3723	0.3196

ตาราง ก.16 ผลลัพธ์การเพิ่มประสิทธิภาพของซัพพอร์ตเวกเตอร์แมชชีนไม่เชิงเส้นแบบฟังก์ชันเคอร์เนล RBF เมื่อ C เท่ากับ 2^{15}

C	γ	ความถูกต้อง	ความระลึก	ความแม่นยำ
2^{15}	2^{-15}	0.3768	0.3884	0.3276
	2^{-13}	0.3444	0.3739	0.2828
	2^{-11}	0.6299	0.6302	0.6805
	2^{-9}	0.6518	0.7127	0.6495
	2^{-7}	0.6533	0.6980	0.6594
	2^{-5}	0.7056	0.6855	0.6560
	2^{-3}	0.6629	0.6997	0.7189
	2^{-1}	0.6676	0.6001	0.6278
	2^{1}	0.4620	0.5394	0.4287
	2^{3}	0.4060	0.3192	0.4093
	2^{5}	0.3865	0.3575	0.4055
	2^{7}	0.3677	0.3758	0.4553
	2^{9}	0.4365	0.3755	0.2890
	2^{11}	0.4360	0.3306	0.2841
	2^{13}	0.4336	0.4691	0.4188
	2^{15}	0.3761	0.4199	0.3549

ภาคผนวก ข

ตัวอย่างการหาความน่าจะเป็นแบบมีเงื่อนไขสำหรับโมเดลนาอีฟเบย์

ในตัวอย่างนี้เราจะหาความน่าเป็นแบบมีเงื่อนไขของ $P("eat"|POS)$ ซึ่งเป็นโอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีและเนื้อหาในบทวิจารณ์มีคำว่า eat อยู่โดยในตัวอย่างนี้จะกำหนดบทวิจารณ์ตัวอย่างขึ้นมา 3 บทวิจารณ์ดังนี้

1. Night market of yaowarat is the best.
2. Recommended! just try to eat shark fin by yourself.
3. Too crowed. There are a lot of people here.

จากประโยคทั้ง 3 ข้างต้นให้ 2 บทวิจารณ์แรกเป็นบทวิจารณ์ที่มีความรู้สึกเชิงบวก (POS) และบทวิจารณ์สุดท้ายเป็นบทวิจารณ์ที่มีความรู้สึกเชิงลบ ทำให้เราสามารถคำนวณ $P("eat"|POS)$ ได้บทวิจารณ์ตัวอย่างทั้งหมดพบว่าเรามีบทวิจารณ์ที่เป็นความรู้สึกเชิงบวก (POS) อยู่ 2 บทวิจารณ์แต่มีเพียงแค่บทวิจารณ์เดียวเท่านั้นที่มีคำว่า eat ดังนั้น

$$P("eat"|POS)=\frac{Number\ of\ \ positive\ reviews\ with\ "eat"}{Number\ of\ \ positive\ reviews}=\frac{1}{2}=0.5$$

ตัวอย่างการวิเคราะห์ความรู้สึกของนักท่องเที่ยว

ในตัวอย่างนี้จะยกตัวอย่างการวิเคราะห์ความรู้สึกด้วยนาอีฟเบย์ซึ่งเป็นวิธีหนึ่งที่ได้ใช้ในงานวิจัย โดยกำหนดให้ “I love Yaowarat” เป็นประโยคตัวอย่างที่จะใช้ในการทดสอบทำให้เมื่อนำไปวิเคราะห์ด้วยโมเดลนาอีฟเบย์โดยกำหนดให้

n คือ ตัวแปรจำนวนคำศัพท์ มีค่าเท่ากับ 3 คือ $"I"$ $"love"$ และ $"yaowarat"$

i คือ ตัวแปรประเภทของคำตอบมีค่าเท่ากับ 2 คือ POS และ NEG

$P(POS)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีมีค่าเท่ากับ 0.6

$P(NEG)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีมีค่าเท่ากับ 0.4

$P("I"|POS)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีและเนื้อหาในบทวิจารณ์มีคำว่า I มีค่าเท่ากับ 0.08

$P("love"|POS)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีและเนื้อหาในบทวิจารณ์มีคำว่า $"love"$ มีค่าเท่ากับ 0.1

$P("yaowarat"|POS)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกดีและเนื้อหาในบทวิจารณ์มีคำว่า $"yaowarat"$ มีค่าเท่ากับ 0.04

$P("I"|NEG)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกไม่ดีและเนื้อหาในบทวิจารณ์มี คำว่า $"I"$ มีค่าเท่ากับ 0.002

$P("love"|NEG)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกไม่ดีและเนื้อหาในบทวิจารณ์มีคำว่า $"love"$ มีค่าเท่ากับ 0.001

$P("yaowarat"|NEG)$ คือ โอกาสที่จะเจอบทวิจารณ์ที่มีความรู้สึกไม่ดีและเนื้อหาในบทวิจารณ์มีคำว่า $"yaowarat"$ มีค่าเท่ากับ 0.01

จากสมการนาอีฟเบย์ (2.6) กับประโยคตัวอย่างที่กำหนดเราสามารถแจกแจงได้ดังนี้

$$P(c|"I\ love\ yaowarat") = \arg\max\{$$
$$P(POS)\big(P(I|POS)+P(love|POS)+P(yaowarat|POS)\big),$$
$$P(NEG)\big(P(I|NEG)+P(love|NEG)+P(yaowarat|NEG)\big)\}$$

เมื่อแทนค่าสมการด้านบนพบว่า

$$P(c|"I\ love\ yaowarat") = \arg\max\{$$
$$0.6(0.08+0.1+0.04),\ 0.4(0.002+0.001+0.01)\}$$
$$= \arg\max\{(0.132),(0.0052)\}$$

หรือเขียนได้เป็นสองคำตอบคือ

$$P(POS|"I\ love\ yaowarat") = 0.132$$

และ

$$P(NEG|"I\ love\ yaowarat") = 0.0052$$

จากผลลัพธ์ทำให้สามารถสรุปได้ว่า “I love yaowarat” นี้เป็นประโยคที่มีความรู้สึกเชิงบวก เนื่องจากมีโอกาสที่จะเกิดเป็นความรู้สึกเชิงบวกเท่ากับ 0.132 ซึ่งมีความน่าจะเป็นมากกว่าความรู้สึกเชิงลบที่มีค่าเท่ากับ 0.0052

ภาคผนวก ค

โปรแกรมไพธอนที่ใช้ในการทดลอง

ลำดับของโปรแกรมไพธอนจะอ้างอิงจากโครงสร้างของงานวิจัยในรูปที่ 3.1 ซึ่งจะมีทั้งหมด 9 ขั้นตอนดังนี้

1. การเก็บข้อมูลจากเว็ปไซต์
2. การเตรียมข้อมูล
3. การแบ่งกลุ่มความสนใจของนักท่องเที่ยว
4. การวิเคราะห์ความรู้สึกของนักท่องเที่ยว
5. การวิเคราะห์ความนิยม
6. การวิเคราะห์ความเด่นและความแพร่หลาย
7. การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก
8. เว็ปแอพพลิเคชั่นสำหรับการวิเคราะห์กลุ่มความสนใจและความรู้สึก
9. การวิเคราะห์แนวโน้มของคำที่นักท่องเที่ยวใช้ในบทวิจารณ์

โดยโปรแกรมไพธอนทั้ง 9 ขั้นตอน สามารถด้านล่างสามารถสั่งใช้งานแบบบรรทัดต่อบรรทัดได้เลยเนื่องจาก ผู้วิจัยใช้ตัวเขียนโปรแกรมการเขียนโค้ดแบบบรรทัดต่อบรรทัด

1. การดึงข้อมูลจากเว็ปไซต์ Tripadvisor

```
# import packages
from datetime import datetime
import time
import pandas as pd
import pickle as pk
from bs4 import BeautifulSoup
import requests
import os
import re
# os.chdir(r"D:\GitHub_Personal\2019-01-Web-Scraping-using-selenium-and-bs4")
# open the output text file
# with open('condo_links_all.txt') as f:
#     condo_links_all = f.read().splitlines()
# print(len(condo_links_all))
# link = "https://www.hipflat.co.th/en/projects/srithong-condo-acugqa"
link = "https://www.tripadvisor.com/Attraction_Review-g293916-d447272-Reviews-or5-
Chinatown_Bangkok-Bangkok.html#REVIEWS"
page = requests.get(link)
print(link)
soup = BeautifulSoup(page.content, 'html.parser')
soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--1Kusx")[0].findAll("span")
# [0] implies the first review in the page
# wrote date
post_date = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[0].findAll("span")[3].get_text().split(" ")[-2:]
post_month = post_date[0]
```

```
post_year = post_date[1]
# address
# soup.findAll(class_= "social-member-MemberHeaderStats__event_info--
30wFs")[0].findAll("span")[0].get_text()
address = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[0].findAll("span")[5].get_text()
# contribution
# soup.findAll(class_= "social-member-MemberHeaderStats__event_info--
30wFs")[0].findAll("span")[5].get_text()
contribution = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[0].findAll("span")[9].get_text()
# help_vote # not working
# help_vote = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[0].findAll("span")[12].get_text()
# star
star = re.sub('\D', '', str(soup.findAll(class_= "location-review-review-list-parts-
RatingLine__bubbles--GcJvM")[0].find("span")))
# in_short
in_short = soup.findAll(class_= "location-review-review-list-parts-
ReviewTitle__reviewTitle--2GO9Z")[0].get_text()
# content
content = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[0].findAll("span")[0].get_text()
# exp date
# exp_date = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[0].findAll("span")[3].get_text().split(" ")[-
2:] can't be used due to there are some anormally pattern in the structure
exp_date = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[0].findAll("span")[-11].get_text().split(" ")[-
2:]
```

```
exp_month = exp_date[0]
exp_year = exp_date[1]
print(post_month, "|", post_year)
print(address + contribution + help_vote)
print(star)
print(in_short)
print(content)
print(exp_month, "|", exp_year)
for i in range(5):
    # don't use post_date we wil use post_month and year instead
    post_date = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[3].get_text().split(" ")[-2:]
    post_month = post_date[0]
    post_year = post_date[1]
    address = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[5].get_text()
    contribution = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[9].get_text()
    # help_vote = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[12].get_text()
    star = re.sub('\D', '', str(soup.findAll(class_= "location-review-review-list-parts-
RatingLine__bubbles--GcJvM")[i].find("span")))
    in_short = soup.findAll(class_= "location-review-review-list-parts-
ReviewTitle__reviewTitle--2GO9Z")[i].get_text()
    content = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[i].findAll("span")[0].get_text()

    # don't use exp_date
```

```
    exp_date = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[i].findAll("span")[-11].get_text().split(" ")[-
2:]
    exp_month = exp_date[0]
    exp_year = exp_date[1]
    print(exp_month, exp_year, i)
# previous work
# address = soup.find(class_= "social-member-MemberHeaderStats__event_info--
30wFs").findAll("span")[0].get_text()
# contribution = soup.find(class_= "social-member-MemberHeaderStats__event_info--
30wFs").findAll("span")[5].get_text()
# help_vote = soup.find(class_= "social-member-MemberHeaderStats__event_info--
30wFs").findAll("span")[8].get_text()
# exp_date = soup.find(class_= "location-review-review-list-parts-
EventDate__event_date--1epHa").get_text().split(" ")[-2:]
# exp_date = soup.find(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE").findAll("span")[3].get_text().split(" ")[-2:]
df = pd.DataFrame(columns=["header_name", "address", "job"])
len(links)
# import packages
from datetime import datetime
import time
import pandas as pd
import pickle as pk
from bs4 import BeautifulSoup
import requests
import os
import re
def retrieve(link):
    # page = requests.get(link)
    # print(link)
    # soup = BeautifulSoup(page.content, 'html.parser')
    # don't use post_date we wil use post_month and year instead
```

```
    # post_date = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[3].get_text().split(" ")[-2:]
    # post_month = post_date[0]
    # post_year = post_date[1]
    # address = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[5].get_text()
    # contribution = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[9].get_text()
    # help_vote = soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--
1Kusx")[i].findAll("span")[12].get_text()
    star = re.sub('\D', '', str(soup.findAll(class_= "location-review-review-list-parts-
RatingLine__bubbles--GcJvM")[i].find("span")))
    # in_short = soup.findAll(class_= "location-review-review-list-parts-
ReviewTitle__reviewTitle--2GO9Z")[i].get_text()
    content = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[i].findAll("span")[0].get_text()

    # don't use exp_date
    # exp_date = soup.findAll(class_= "location-review-review-list-parts-
ExpandableReview__containerStyles--1G0AE")[i].findAll("span")[-11].get_text().split(" ")[-
2:]
    # exp_month = exp_date[0]
    # exp_year = exp_date[1]
    # return ([star, in_short, content, exp_month, exp_year, post_month, post_year,
address, contribution])
    return ([star, content])
link = 'https://www.tripadvisor.com/Attraction_Review-g293916-d447272-Reviews-
or55-Chinatown_Bangkok-Bangkok.html'
page = requests.get(link)
```

```
print(link)
soup = BeautifulSoup(page.content, 'html.parser')
soup.findAll(class_= "social-member-event-
MemberEventOnObjectBlock__member_event_block--1Kusx")[3].findAll("span")
import pandas as pd
# df = pd.DataFrame(columns=["star", "in_short", "content", "exp_month", "exp_year",
"post_month", "post_year", "address", "contribution"])
start_time = datetime.now()
tripad_list=[]
r=0
for link in links:
    page = requests.get(link)
    soup = BeautifulSoup(page.content, 'html.parser')

    # tripAd increase each webpage index with 5. So,
    for i in range(5):
        try:
            tripad_list.append(retrieve(link))
        except Exception:
            pass
    # try:
    #     data = retrieve(link)
    #     df.loc[i] = pd.Series(data, index = df.columns)
    # except Exception: # Let the codes go if there is any error.
    #     pass
    print(r, link)
    # time_elapsed = datetime.now() - start_time
    # print('Time elapsed (hh:mm:ss.ms) {}'.format(time_elapsed))
    # ### Give the 'sleep' time = 5 seconds. Space out each request so the server
isn't overwhelmed.
    time.sleep(1)
    r=r+1
df_complete = pd.DataFrame(tripad_list)
```

```
#     # This is the preventive step...
#     # You can even clear the list and name a new file to save processing memory.
#     # Dump the data periodically every 5 iterations.
#     if (i%5==0):
#         # Delete 'None' elements from the list.
#         tripad_list = [c for c in tripad_list if c is not None]
#         df = pd.DataFrame(tripad_list)
#         with open('df.pkl', 'wb') as f:
#             pk.dump(df, f)
#         # Print out i,len(condo_list), so we can trace back if error occur.
#         # i is the index of 'condo_links_all'
#         print('------------------------ dump @ i = ',i,len(tripad_list))
# print("completed")
# # Once complete, dump to pickle and save as 'df_completed.pkl'.
# tripad_list = [c for c in tripad_list if c is not None]
# df_completed = pd.DataFrame(tripad_list)
# with open('df_completed.pkl', 'wb') as f:
#     pk.dump(df_completed, f)
df_complete.to_excel("tripadvisor_dataset_china_town.xlsx")
from google.colab import files
files.download('tripadvisor_dataset_china_town.xlsx')
pd.DataFrame(tripad_list)
# Once complete, dump to pickle and save as 'df_completed.pkl'.
tripad_list = [c for c in tripad_list if c is not None]
df_completed = pd.DataFrame(tripad_list)
with open('df_completed.pkl', 'wb') as f:
    pk.dump(df_completed, f)
df_completed.pkl
test = [ ["name1", "add1", "job1"], ["name2", "add2", "job2"] ]
pd.DataFrame(test)
link = "https://www.hipflat.co.th/en/listings/bangkok-condo-jrexisav"
test = scrapping(link)
start_time = datetime.now()
```

```
condo_list=[]
i=0z
for link in condo_links_all:
    try:
        condo_list.append(retrieve(link))
    except Exception: # Let the codes go if there is any error.
        pass
    print(i)
    time_elapsed = datetime.now() - start_time
    print('Time elapsed (hh:mm:ss.ms) {}'.format(time_elapsed))


    ### Give the 'sleep' time = 5 seconds. Space out each request so the server isn't
overwhelmed.
    time.sleep(5)
    i=i+1
    # This is the preventive step...
    # You can even clear the list and name a new file to save processing memory.
    # Dump the data periodically every 5 iterations.
    if (i%5==0):
        # Delete 'None' elements from the list.
        condo_list = [c for c in condo_list if c is not None]
        df = pd.DataFrame(condo_list)
        # Print out i,len(condo_list), so we can trace back if error occur.
        # i is the index of 'condo_links_all'
        print('---------------------- dump @ i = ',i,len(condo_list))
print("completed")
import pandas as pd
df = pd.DataFrame(columns=["header_name", "address", "latitude", "longitude",\
            "year_built", "proj_area", "num_buildings", "num_floors", \
            "for_sale", "for_rent", "num_bed", "num_bath", "in_area", \
            "promoter", "description", "amenities", "transportation", \
            "price_sqm", "change_last_q", "change_last_y", "rental_yield",
"change_last_y_rental_price"])
```

```
df.loc[0] = pd.Series(test, index = df.columns)
#####################################################################
# Run the loop to retrieve data and store data as DataFrame, save as pickle.
start_time = datetime.now()
condo_list=[]
i=0
for link in condo_links_all:
    try:
        condo_list.append(retrieve(link))
    except Exception: # Let the codes go if there is any error.
        pass
    print(i)
    time_elapsed = datetime.now() - start_time
    print('Time elapsed (hh:mm:ss.ms) {}'.format(time_elapsed))

    ### Give the 'sleep' time = 5 seconds. Space out each request so the server isn't
overwhelmed.
    time.sleep(5)
    i=i+1
    # This is the preventive step...
    # You can even clear the list and name a new file to save processing memory.
    # Dump the data periodically every 5 iterations.
    if (i%5==0):
        # Delete 'None' elements from the list.
        condo_list = [c for c in condo_list if c is not None]
        df = pd.DataFrame(condo_list)
        with open('df.pkl', 'wb') as f:
            pk.dump(df, f)
        # Print out i,len(condo_list), so we can trace back if error occur.
        # i is the index of 'condo_links_all'
        print('---------------------- dump @ i = ',i,len(condo_list))
print("completed")
# Once complete, dump to pickle and save as 'df_completed.pkl'.
```

```
condo_list = [c for c in condo_list if c is not None]
df_completed = pd.DataFrame(condo_list)
with open('df_completed.pkl', 'wb') as f:
    pk.dump(df_completed, f)

# export to csv
col_names=
['name','district','latitude','longitude','year_built','proj_area','nbr_buildings','nbr_floors','u
nits', 'shops','schools','restaurants','hospital','amenities','transportation',
'price_sqm','change_last_q','change_last_y','rental_yield','change_last_y_rental_price','p
rice_hist']
df_completed.to_csv("df_completed.csv" ,header=col_names,index=False,encoding='
utf-8-sig')
#load csv
df_dirty= pd.read_csv("df_completed.csv", sep=',',encoding='utf-8-sig')
```

2. การเตรียมข้อมูล

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import re
# % matplotlib inline
plt.style.use('ggplot')
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
import unicodedata
import contractions # signature
from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
# import model_evaluation_utils as meu # signature
np.set_printoptions(precision=2, linewidth=80)
full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
full_df.head(5)
# display dateset size
print(full_df.shape[0], "rows ,", full_df.shape[1], "columns")
# add sentiment column
full_df["sentiment"] = full_df["star"].apply(lambda star: 1 if star >= 40 else 0)
full_df["sentiment"].value_counts()
# # in case of negative sentiment is insufficient
# df["count"] = df["in_short"].apply(lambda sentence: (sentence.count(" ")+1))
# df.loc[(df['count'] >= 3) & (df['sentiment'] == 1)]
```

```
# df["exp_month"] = df["exp_month"].astype(str).apply(lambda month:
month.replace("January","Jan").replace("January","Jan").replace("January","Jan"))
df = full_df[["content","sentiment"]]
df.shape
from sklearn.model_selection import train_test_split
train, test = train_test_split(df, test_size=0.70, random_state=42)
train_reviews = train['content']
train_sentiments = train["sentiment"]
test_reviews = test["content"]
test_sentiments = test["sentiment"]
# normalize datasets
stop_words = nltk.corpus.stopwords.words('english')
stop_words.remove('no')
stop_words.remove('but')
stop_words.remove('not')
# # Full stack cleaning
# norm_train_reviews = tn.normalize_corpus(train_reviews, stopwords=stop_words)
# norm_test_reviews = tn.normalize_corpus(test_reviews, stopwords=stop_words)
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.cluster import KMeans
norm_reviews = tn.normalize_corpus(df["content"], stopwords=stop_words)
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0)
reviews_features = cv.fit_transform(norm_reviews)
# tv = TfidfVectorizer(use_idf=True, min_df=0.0, max_df=1.0, sublinear_tf=True) #
ngram_range=(1,2)
# reviews_features_tf = tv.fit_transform(norm_reviews )
# plt.savefig('foo.png')
```

3. การแบ่งกลุ่มความสนใจของนักท่องเที่ยว

```
sse = []
# for k in range(4, 5):
kmeans = KMeans(n_clusters=4, max_iter=6, random_state=None, n_jobs=-
1).fit(reviews_features)
    # df["clusters"] = kmeans.labels_
    #print(data["clusters"])
#     sse.append(kmeans.inertia_)
# n_cluster = range(1, 11)
# import matplotlib
# font = {'size': 13}
# matplotlib.rc('font', **font)
# plt.figure(figsize=(8,10))
# plt.title("Error VS Number of clusters")
# plt.ylabel("sum square error")
# plt.xlabel("number of cluster")
# axes = plt.gca() # must be below the "plt" method
# plt.plot(n_cluster, sse, "ro")
# plt.show()
df["predict"]= pd.DataFrame(kmeans.predict(reviews_features))
df_for_km = pd.read_excel("Copy of norm_df_for_VIZ.xlsx")
df_for_km = df_for_km[["content", "sentiment"]]
df_for_km["predict"]= pd.DataFrame(kmeans.predict(reviews_features))
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
Klabels = kmeans.labels_
metrics.silhouette_score(reviews_features, Klabels, metric='euclidean')
```

```
sse = []
sil = []
for k in range(2, 10):
    kmeans = KMeans(n_clusters=k, max_iter=5, random_state=5, n_jobs=-
1).fit(reviews_features)
    df["clusters_"+str(k)] = kmeans.labels_
    Klabels = kmeans.labels_
    sse.append(kmeans.inertia_)
    sil.append(metrics.silhouette_score(reviews_features, Klabels, metric='euclidean'))
kmeans = KMeans(n_clusters=4, max_iter=100, random_state=5, n_jobs=-
1).fit(reviews_features)
# df["clusters_"+str(k)] = kmeans.labels_
# Klabels = kmeans.labels_
sse.append(kmeans.inertia_)
# sil.append(metrics.silhouette_score(reviews_features, Klabels, metric='euclidean'))
df_for_km["predict"]= pd.DataFrame(kmeans.predict(reviews_features))
# s_score = sil
# x = [2,3,4,5,6,7,8,9]
# plt.plot(x, s_score, "o")
n_cluster = range(2, 10)
import matplotlib
font = {'size': 10}
matplotlib.rc('font', **font)
plt.figure(figsize=(8,8))
# plt.title("Er VS Number of clusters")
plt.ylabel("Error")
plt.xlabel("Number of Clusters")
axes = plt.gca() # must be below the "plt" method
plt.plot(n_cluster, sse_score, "ro")
plt.show()
kmeans = KMeans(n_clusters=4, max_iter=500, random_state= 10, n_jobs=-
1).fit(reviews_features)
# df["clusters_"+str(k)] = kmeans.labels_
```

```
# Klabels = kmeans.labels_
# sse.append(kmeans.inertia_)
# # sil.append(metrics.silhouette_score(reviews_features, Klabels, metric='euclidean'))
# df_for_km["predict"]= pd.DataFrame(kmeans.predict(reviews_features))
# df_for_km = pd.read_excel("Copy of norm_df_for_VIZ.xlsx")
# df_for_km = df_for_km[["content", "sentiment"]][0:3992]
# df_for_km["predict"]= pd.DataFrame(kmeans.predict(reviews_features))
# df_for_km.to_excel("k_means_result_for_paper.xlsx", index=False,
columns=["content","sentiment","predict"])
main_K = pd.read_excel("k_means_result_for_paper.xlsx")
main_K
review_num_1 = main_K[main_K["predict"] == 0].shape[0]
review_num_2 = main_K[main_K["predict"] == 1].shape[0]
review_num_3 = main_K[main_K["predict"] == 2].shape[0]
review_num_4 = main_K[main_K["predict"] == 3].shape[0]
print(review_num_1 + review_num_2 + review_num_3 + review_num_4)
review_num_1, review_num_2 , review_num_3, review_num_4
all1 = main_K[main_K["predict"] == 0]
all2 = main_K[main_K["predict"] == 1]
all3 = main_K[main_K["predict"] == 2]
all4 = main_K[main_K["predict"] == 3]
NEG1 = all1[all1["sentiment"]==0].shape[0]
POS1 = all1[all1["sentiment"]==1].shape[0]
NEG2 = all2[all2["sentiment"]==0].shape[0]
POS2 = all2[all2["sentiment"]==1].shape[0]
NEG3 = all3[all3["sentiment"]==0].shape[0]
POS3 = all3[all3["sentiment"]==1].shape[0]
NEG4 = all4[all4["sentiment"]==0].shape[0]
POS4 = all4[all4["sentiment"]==1].shape[0]
print(POS1 ,NEG1)
print(POS2,NEG2)
print(POS3,NEG3)
print(POS4,NEG4)
```

```
c0 = df_for_km[df_for_km["predict"] == 0]["content"].tolist()
lst = []
for i in c0:
    lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    count0 = len([w for a_lst in lst for w in a_lst])
pd.DataFrame(pd.Series([w for a_lst in lst for w in a_lst]).value_counts()[0:10])
c1 = df_for_km[df_for_km["predict"] == 1]["content"].tolist()
lst = []
for i in c1:
    lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    count1 = len([w for a_lst in lst for w in a_lst])
pd.DataFrame(pd.Series([w for a_lst in lst for w in a_lst]).value_counts()[0:10])
c2 = df_for_km[df_for_km["predict"] == 2]["content"].tolist()
lst = []
for i in c2:
    lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    count2 = len([w for a_lst in lst for w in a_lst])
pd.DataFrame(pd.Series([w for a_lst in lst for w in a_lst]).value_counts()[0:10])
c3 = df_for_km[df_for_km["predict"] == 3]["content"].tolist()
lst = []
for i in c3:
    lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    count3 = len([w for a_lst in lst for w in a_lst])
pd.DataFrame(pd.Series([w for a_lst in lst for w in a_lst]).value_counts()[0:10])
pd.DataFrame([count0 ,count1 ,count2 ,count3])
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
# build BOW features on train reviews
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0) # ngram_range=(1,2)
cv_train_features = cv.fit_transform(norm_train_reviews)
# build TFIDF features on train reviews
tv = TfidfVectorizer(use_idf=True, min_df=0.0, max_df=1.0, sublinear_tf=True) #
ngram_range=(1,2)
tv_train_features = tv.fit_transform(norm_train_reviews)
```

```
# transform test reviews into features
# CROSS VALIDATION; https://towardsdatascience.com/why-and-how-to-cross-validate-a-model-d6424b45261f
cv_test_features = cv.transform(norm_test_reviews)
tv_test_features = tv.transform(norm_test_reviews)
print('BOW model: Train features shape:', cv_train_features.shape[0], ' Test features shape:', cv_test_features.shape[0])
# print('TFIDF model: Train features shape:', tv_train_features.shape[0], ' Test features shape:', tv_test_features.shape[0])
"""## Modeling ### dependencies"""
```

4. การวิเคราะห์ความรู้สึกของนักท่องเที่ยว

```
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import LinearSVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.svm import SVC
# from sklearn.ensemble import RandomForestClassifier
# template paper, more papers refer to ch.2 ch.5, COVID, strategies after COVID...
# proposal, short introduction
lr = LogisticRegression(penalty='l2', max_iter=1000, C=1) # parameters, why L2
norm???
svm = LinearSVC(penalty='l2', C=1, random_state=42) # parameters, C????,
optimization, F1???
mnb = MultinomialNB(alpha=1) # parameters
# bayesian network
#
# rfc = RandomForestClassifier(n_estimators=10, random_state=42)
gbc = GradientBoostingClassifier(n_estimators=10, random_state=42)
from sklearn import metrics
def train_predict_model(classifier,
                        train_features, train_labels,
                        test_features, test_labels):
    # build model
    classifier.fit(train_features, train_labels)
    # predict using model
    predictions = classifier.predict(test_features)
```

```
    return predictions
# Model Performance metrics:
def performance_metrics(true_labels, predicted_labels):
    print('Accuracy:', np.round(metrics.accuracy_score(true_labels, predicted_labels),4))
    print('Precision:', np.round(metrics.precision_score(true_labels,
predicted_labels,average='weighted'),4))
    print('Recall:', np.round(metrics.recall_score(true_labels,
predicted_labels,average='weighted'),4))
    print('F1 Score:', np.round(metrics.f1_score(true_labels,
predicted_labels,average='weighted'),4))
# full result
def classification_report(true_labels, predicted_labels, classes=[1,0]):
    report = metrics.classification_report(y_true=true_labels, y_pred=predicted_labels,
labels=classes)
    print(report)
# confusion_matrix(
def confusion_matrix(true_labels, predicted_labels, classes=[1,0]):
    total_classes = len(classes)
    level_labels = [total_classes*[0], list(range(total_classes))]
    cm = metrics.confusion_matrix(y_true=true_labels, y_pred=predicted_labels,
labels=classes)
    cm_frame = pd.DataFrame(data=cm,
                            columns=pd.MultiIndex(levels=[['Predicted:'], classes],
codes=level_labels),
                            index=pd.MultiIndex(levels=[['Actual:'], classes],
codes=level_labels))
    print(cm_frame)
"""### Sentiment analyzer
#### Naive bayes
"""
# mnb_bow_predictions = meu.train_predict_model(classifier = mnb,
#                                               train_features = cv_train_features,
#                                               train_labels = train_sentiments,
```

```
#                                       test_features = cv_test_features,
#                                       test_labels = test_sentiments)
# print('Model Performance metrics:')
# print('-'*30)
# performance_metrics(test_sentiments, mnb_bow_predictions)
# print('\nModel Classification report:')
# print('-'*30)
# classification_report(test_sentiments, mnb_bow_predictions)
# print('\nPrediction Confusion Matrix:')
# print('-'*30)
# confusion_matrix(test_sentiments, mnb_bow_predictions)
from sklearn.naive_bayes import MultinomialNB as NaiveB
from sklearn.model_selection import cross_val_score
nb = NaiveB()
scores = cross_val_score(nb, cv_train_features, train_sentiments, cv=10)
# nb.fit(cv_train_features, train_sentiments)
import statistics as stats
stats.mean(scores)
from sklearn.naive_bayes import MultinomialNB as NaiveB
from sklearn.model_selection import cross_val_score
nb = NaiveB()
nb.fit(cv_train_features, train_sentiments)
from sklearn.model_selection import GridSearchCV
result_nb_all = []
result_nb_best = []
a_lst = np.arange(1, 100, 0.5)
# [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7, 2**9,
2**11, 2**13, 2**15]
for i in a_lst:
    gg = [i]
    parameters = {'alpha': gg}
    nb = NaiveB()
    cnb = GridSearchCV(nb, parameters, cv=10)
```

```
    cnb.fit(cv_train_features, train_sentiments)
    result_nb_all.append(cnb.cv_results_)
    result_nb_best.append(cnb.best_score_)
# compared to real sentiment from star
# 1 as positive and 0 as negative
NB_df = test[["content", "sentiment"]].reset_index(drop=True)
NB_df ["predict_NB"] = pd.DataFrame(list(mnb_bow_predictions))
NB_df.head()
"""#### LR"""
# >>> from sklearn import svm, datasets
# >>> from sklearn.model_selection import GridSearchCV
# >>> iris = datasets.load_iris()
# >>> parameters = {'kernel':('linear', 'rbf'), 'C':[1, 10]}
# >>> svc = svm.SVC()
# >>> clf = GridSearchCV(svc, parameters)
# >>> clf.fit(iris.data, iris.target)
# GridSearchCV(estimator=SVC(),
#              param_grid={'C': [1, 10], 'kernel': ('linear', 'rbf')})
# >>> sorted(clf.cv_results_.keys())
# ['mean_fit_time', 'mean_score_time', 'mean_test_score',...
#  'param_C', 'param_kernel', 'params',...
#  'rank_test_score', 'split0_test_score',...
#  'split2_test_score', ...
#  'std_fit_time', 'std_score_time', 'std_test_score']
# print(__doc__)
# # Loading the Digits dataset
# digits = datasets.load_digits()
# # To apply an classifier on this data, we need to flatten the image, to
# # turn the data in a (samples, feature) matrix:
# n_samples = len(digits.images)
# X = digits.images.reshape((n_samples, -1))
# y = digits.target
from sklearn.linear_model import LogisticRegression, SGDClassifier
```

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import LinearSVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.svm import SVC
# lr = LogisticRegression(penalty='l2', max_iter=1000, random_state = 42) #
parameters, why L2 norm???
# svm = LinearSVC(penalty='l2', C=1, random_state=42)
# mnb = MultinomialNB(alpha=1) # parameters
[0.001, 0.01, 0.1, 1, 10, 100, 1000]
"""solver : {‘newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’, ‘saga’}, default=’lbfgs’
‘newton-cg’, ‘lbfgs’, ‘sag’ and ‘saga’ handle L2 or no penalty
‘liblinear’ and ‘saga’ also handle L1 penalty
"""
# df_result = pd.DataFrame(["1**-15", "1**-13", "1**-11", "1**-9", "1**-7", "1**-5", "1**-3",
"1**-1", "1**1", "1**3", "1**5", "1**7", "1**9", "1**11", "1**13", "1**15"], columns=["C"])
# [1**-15, 1**-13, 1**-11, 1**-9, 1**-7, 1**-5, 1**-3, 1**-1, 1**1, 1**3, 1**5, 1**7, 1**9,
1**11, 1**13, 1**15]
from sklearn.model_selection import cross_val_score
lr = LogisticRegression(penalty='none', max_iter=1000, random_state = 42)
scores = cross_val_score(lr, cv_train_features, train_sentiments, cv=10)
import statistics as stats
stats.mean(scores)
result_lr_all = []
result_lr_best = []
result_lr_l1_all = []
result_lr_l1_best = []
result_lr_l2_all = []
result_lr_l2_best = []
df_result = pd.DataFrame(["2**-15", "2**-13", "2**-11", "2**-9", "2**-7", "2**-5", "2**-3",
"2**-1", "2**1", "2**3", "2**5", "2**7", "2**9", "2**11", "2**13", "2**15"], columns=["C"])
```

```
c_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
# penalty='none' is no "C"
# for i in c_lst:
#     gg = [i]
#     parameters = {'C': gg}
#     lr = LogisticRegression(penalty='none', max_iter=1000, random_state = 42)
#     clr = GridSearchCV(lr, parameters, cv=10)
#     clr.fit(cv_train_features, train_sentiments)
#     result_lr_all.append(clr.cv_results_)
#     result_lr_best.append(clr.best_score_)
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    lr = LogisticRegression(penalty='l1', max_iter=1000, random_state = 42,
solver='liblinear')
    clr = GridSearchCV(lr, parameters, cv=10)
    clr.fit(cv_train_features, train_sentiments)
    result_lr_l1_all.append(clr.cv_results_)
    result_lr_l1_best.append(clr.best_score_)
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    lr = LogisticRegression(penalty='l2', max_iter=1000, random_state = 42,
solver='liblinear')
    clr = GridSearchCV(lr, parameters, cv=10)
    clr.fit(cv_train_features, train_sentiments)
    result_lr_l2_all.append(clr.cv_results_)
    result_lr_l2_best.append(clr.best_score_)
# df_result["lr"] = pd.Series(result_lr_best)
df_result["lr_l1"] = pd.Series(result_lr_l1_best)
df_result["lr_12"] = pd.Series(result_lr_l2_best)
# GridSearchCV(estimator = clr, param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] })
```

```
df_result
# clr.cv_results_["params"][0]["C"]
df_result.to_excel("logist_regression_result.xlsx", encoding="utf-8",
columns=df_result.columns)
"""# SVM
`[linear', 'poly', 'rbf']`
```
l2, l1
```
### LinearSVC
"""
df_result_SVM = pd.DataFrame(["2**-15", "2**-13", "2**-11", "2**-9", "2**-7", "2**-5", "2**-
3", "2**-1", "2**1", "2**3", "2**5", "2**7", "2**9", "2**11", "2**13", "2**15"], columns=["C"])
result_svm_all = []
result_svm_best = []
result_svm_l1_all = []
result_svm_l1_best = []
result_svm_l2_all = []
result_svm_l2_best = []
c_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    svm = LinearSVC(penalty='l1', random_state=42, dual=False)
    csvm = GridSearchCV(svm, parameters, cv=10)
    csvm.fit(cv_train_features, train_sentiments)
    result_svm_l1_all.append(csvm.cv_results_)
    result_svm_l1_best.append(csvm.best_score_)
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    svm = LinearSVC(penalty='l2', random_state=42, dual=False)
```

```
    csvm = GridSearchCV(svm, parameters, cv=10)
    csvm.fit(cv_train_features, train_sentiments)
    result_svm_l2_all.append(csvm.cv_results_)
    result_svm_l2_best.append(csvm.best_score_)
df_result_SVM["svm_l1"] = pd.Series(result_svm_l1_best)
df_result_SVM["svm_12"] = pd.Series(result_svm_l2_best)
# GridSearchCV(estimator = clr, param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] })
df_result_SVM
df_result_SVM.to_excel("svm_regular_2.xlsx", encoding="utf-8",
columns=df_result_SVM.columns)
"""### SVC (Non linear)
#### RBF
"""
from sklearn.svm import SVC
result_Ksvm_rbf_best = []
name = ["-15", "-13", "-11", "-9", "-7", "-5", "-3", "-1", "1", "3", "5", "7", "9", "11", "13", "15"]
c_lst = [2**7, 2**9, 2**11, 2**13, 2**15]
g_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
df_result_KSVM_rbf = pd.DataFrame(list(range(1, len(g_lst)+1)), columns=["index"])
round_c = 6
for c in c_lst:
    cc = [c]*len(c_lst)
    result_Ksvm_rbf_best = []
    for g in g_lst:
        gg = [g]
        parameters = {'C': cc, "gamma":gg}
        Ksvm = SVC(kernel="rbf", random_state=42)
        cKsvm = GridSearchCV(Ksvm, parameters, cv=10, n_jobs=-1)
        cKsvm.fit(cv_train_features, train_sentiments)
        result_Ksvm_rbf_best.append(cKsvm.best_score_)

    print(result_Ksvm_rbf_best)
```

```
    df_result_KSVM_rbf["C"+ name[round_c]] =
pd.DataFrame([name[round_c]]*len(g_lst))
    df_result_KSVM_rbf["G"+ name[round_c]] = pd.DataFrame(name)
    df_result_KSVM_rbf["acc"+ name[round_c]] = pd.DataFrame(result_Ksvm_rbf_best)
    round_c += 1
# df_result_KSVM_rbf.to_excel("7_to_15.xlsx", encoding="utf-8",
columns=df_result_KSVM_rbf.columns)
from sklearn.svm import SVC
df_result_SVM_li = pd.DataFrame(["2**-15", "2**-13", "2**-11", "2**-9", "2**-7", "2**-5",
"2**-3", "2**-1", "2**1", "2**3", "2**5", "2**7", "2**9", "2**11", "2**13", "2**15"],
columns=["C"])
result_Ksvm_li_all = []
result_Ksvm_li_best = []
c_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    Ksvm = SVC(kernel="linear", random_state=42)
    csvm = GridSearchCV(Ksvm, parameters, cv=10)
    csvm.fit(cv_train_features, train_sentiments)
    result_Ksvm_li_all.append(csvm.cv_results_)
    result_Ksvm_li_best.append(csvm.best_score_)
df_result_SVM_li["Ksvm_li"] = pd.Series(result_Ksvm_li_best)
# GridSearchCV(estimator = clr, param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] })
df_result_SVM_li.to_excel("Ksvm_li.xlsx", encoding="utf-8",
columns=df_result_SVM_li.columns)
from sklearn.svm import SVC
# result_Ksvm_poly_best = []
name = ["-15", "-13", "-11", "-9", "-7", "-5", "-3", "-1", "1", "3", "5", "7", "9", "11", "13", "15"]
c_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
```

```
g_lst = [2**-15, 2**-13, 2**-11, 2**-9, 2**-7, 2**-5, 2**-3, 2**-1, 2**1, 2**3, 2**5, 2**7,
2**9, 2**11, 2**13, 2**15]
df_result_KSVM_poly = pd.DataFrame(list(range(1, len(g_lst)+1)), columns=["index"])
round_c = 0
for c in c_lst:
    cc = [c]*len(c_lst)
    result_Ksvm_poly_best = []
    for g in g_lst:
        gg = [g]
        parameters = {'C': cc, "gamma":gg}
        Ksvm = SVC(kernel="poly", random_state=42, degree=7, cache_size=1000)
        cKsvm = GridSearchCV(Ksvm, parameters, cv=5, n_jobs=-1)
        cKsvm.fit(cv_train_features, train_sentiments)
        result_Ksvm_poly_best.append(cKsvm.best_score_)

    print(result_Ksvm_rbf_best)
    df_result_KSVM_poly["C"+ name[round_c]] =
pd.DataFrame([name[round_c]]*len(g_lst))
    df_result_KSVM_poly["G"+ name[round_c]] = pd.DataFrame(name)
    df_result_KSVM_poly["acc"+ name[round_c]] =
pd.DataFrame(result_Ksvm_poly_best)
    round_c += 1
    df_result_KSVM_poly.to_excel("-15_to_15_POLY_7.xlsx", encoding="utf-8",
columns=df_result_KSVM_poly.columns)
df_result_KSVM_poly.to_excel("-15_to_15_POLY.xlsx", encoding="utf-8",
columns=df_result_KSVM_poly.columns)
result_svm_l2_all = []
result_svm_l2_best = []
for i in c_lst:
    gg = [i]
    parameters = {'C': gg}
    svm = LinearSVC(penalty='l2', random_state=42, loss="hinge")
    clr = GridSearchCV(lr, parameters, cv=10)
```

```
    clr.fit(cv_train_features, train_sentiments)
    result_svm_l2_all.append(clr.cv_results_)
    result_svm_l2_best.append(clr.best_score_)

df_result_SVM["svm_12"] = pd.Series(result_svm_l2_best)
# lr.score(cv_train_features, train_sentiments)
aa[0]["param_C"].data
# Commented out IPython magic to ensure Python compatibility.
print("Best parameters set found on development set:"); print()
print(clr.best_params_); print()
print("Grid scores on development set:"); print()
means = clr.cv_results_['mean_test_score']
stds = clr.cv_results_['std_test_score']
for mean, std, params in zip(means, stds, clr.cv_results_['params']):
    print("%0.3f (+/-%0.03f) for %r"
#           % (mean, std * 2, params))
print()
print("Detailed classification report:"); print()
print("The model is trained on the full development set.")
print("The scores are computed on the full evaluation set."); print()
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression, SGDClassifier
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] }
clf = GridSearchCV(LogisticRegression(penalty='l2'), param_grid)
GridSearchCV(cv=None, estimator = LogisticRegression(C=1.0,
                                    intercept_scaling=1,
                                    dual=False,
                                    fit_intercept=True,
                                    penalty='l2',
                                    tol=0.0001),
                                    param_grid={'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]})
# Logistic Regression model on BOW features
lr_bow_predictions = train_predict_model(classifier = lr,
```

```
                                    train_features = cv_train_features,
                                    train_labels = train_sentiments,
                                    test_features = cv_test_features,
                                    test_labels = test_sentiments)
print('Model Performance metrics:')
print('-'*30)
performance_metrics(test_sentiments, lr_bow_predictions)
print('\nModel Classification report:')
print('-'*30)
classification_report(test_sentiments, lr_bow_predictions)
print('\nPrediction Confusion Matrix:')
print('-'*30)
confusion_matrix(test_sentiments, lr_bow_predictions)
LR_df = test[["content", "sentiment"]].reset_index(drop=True)
LR_df["predict_LR"] = pd.DataFrame(list(lr_bow_predictions))
LR_df.head()
"""#### SVM"""
svm_bow_predictions = train_predict_model(classifier = svm,
                                          train_features = cv_train_features,
                                          train_labels = train_sentiments,
                                          test_features = cv_test_features,
                                          test_labels = test_sentiments)
# optimization
print('Model Performance metrics:')
print('-'*30)
performance_metrics(test_sentiments, svm_bow_predictions)
print('\nModel Classification report:')
print('-'*30)
classification_report(test_sentiments, svm_bow_predictions)
print('\nPrediction Confusion Matrix:')
print('-'*30)
confusion_matrix(test_sentiments, svm_bow_predictions)
SVM_df = test[["content", "sentiment"]].reset_index(drop=True)
```

```
SVM_df["predict_SVM"] = pd.DataFrame(list(svm_bow_predictions))
SVM_df.head()
"""#### summary"""
test_df = test[["content", "sentiment"]] #.reset_index(drop=True)
test_df["predict_NB"] = pd.DataFrame(list(mnb_bow_predictions))
test_df["predict_LR"] = pd.DataFrame(list(lr_bow_predictions))
test_df["predict_SVM"] = pd.DataFrame(list(svm_bow_predictions))
test_df["sum"] = test_df.sum(axis=1)
test_df["all"] = test_df["sum"].apply(lambda sum: 1 if sum > 1 else 0)
test_df.drop(columns=["sum"], axis=1, inplace=True)
test_df.head()
test_df.to_excel("LSVA_dataset.xlsx")
test_df["compare"] = test_df["all"] == test_df["sentiment"]
test_df["compare"].value_counts()
test_df_part = test_df[["content","sentiment", "all"]]
test_df_part = test_df_part.astype(str)
test_df_part["concat"] = test_df_part["sentiment"] + test_df_part["all"]
test_df_part["concat"].value_counts()
### LDA ####
# Commented out IPython magic to ensure Python compatibility.
# %%capture
# !pip install gensim
# https://gist.github.com/alvations/a4a6e0cc24d2fd9aff86
from math import sqrt
from collections import defaultdict
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
def get_V(y, n):
    ngram_vectorizer = CountVectorizer(analyzer='char', ngram_range=(n-1, n),
min_df=1)
    V = ngram_vectorizer.fit_transform(y)
    return V,  ngram_vectorizer
def get_V_by_size(V):
```

```
    V_by_size = defaultdict(list)
    for i, y_vec in enumerate(V):
        size_y = np.sum(y_vec.toarray())
        V_by_size[size_y].append(i)
    return V_by_size
def min_max_y(size_X, alpha):
    return int(alpha * alpha * size_X), int(size_X / (alpha * alpha))
def overlapjoin(vec_x, tau, sub_V, l):
    for i, _y in sub_V:
        num_overlaps = sum([1 for x_fi, y_fi in zip(vec_x, _y)
                            if x_fi&y_fi > 0 and x_fi == y_fi])
        if num_overlaps > tau:
            yield i


def approx_dict_matching(x, y, V=None, vectorizer=None, V_by_size=None, n=3,
alpha=0.7):
    if V == vectorizer == V_by_size == None:
        V, vectorizer = get_V(y, n)
        V_by_size = get_V_by_size(V)


    vec_x = vectorizer.transform([x]).toarray()[0]


    size_X = sum(vec_x)
    min_y, max_y = min_max_y(size_X, alpha)
    output = set()
    for l in range(min_y, max_y):
        tau = alpha * sqrt(size_X * l)
        sub_V_indices = V_by_size.get(l)
        if sub_V_indices:
            sub_V = [(i, V[i].toarray()[0]) for i in sub_V_indices]
            R = (list(overlapjoin(vec_x, tau, sub_V, l)))
            output.update(R)
```

```
    return set([y[i] for i in output])


import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from nltk.stem.porter import *
import pandas as pd
import numpy as np
np.random.seed(2018)
import nltk
nltk.download('wordnet')
# Commented out IPython magic to ensure Python compatibility.
import matplotlib.pyplot as plt
import seaborn as sns
import re
# % matplotlib inline
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
import unicodedata
import contractions # signature
from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
import model_evaluation_utils as meu # signature
np.set_printoptions(precision=2, linewidth=80)
full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
full_df.head(2)
# add sentiment column
full_df["sentiment"] = full_df["star"].apply(lambda star: "positive" if star >= 40 else
"negative")
```

```
full_df["sentiment"].value_counts()
# df["exp_month"] = df["exp_month"].astype(str).apply(lambda month:
month.replace("January","Jan").replace("January","Jan").replace("January","Jan"))
from nltk.stem.porter import *
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
stemmer = PorterStemmer()
def preprocess(text):
    result = []
    full_dict = []
    for token in gensim.utils.simple_preprocess(text):
        # if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3
and \
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3\
        and token not in ["chinatown", "bangkok", "china", "chinese", "place", "go", "dont",
"morning", "possible", "visit"]:
        # token != "chinatown" and token != "bangkok" and token != "china" and \
        # token != "chinese":
            result.append(lemmatize_stemming(token))
    return result
def all_dict(text):
    full_dict = []
    for token in gensim.utils.simple_preprocess(text):
        # if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3
and \
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3\
        and token not in ["chinatown", "bangkok", "china", "chinese", "place", "go",
"good"]:
        # token != "chinatown" and token != "bangkok" and token != "china" and \
        # token != "chinese":
            full_dict.append(token)
    return full_dict
data_text = full_df["content"]
```

```
processed_docs = data_text.map(preprocess)
full_dict = data_text.map(all_dict)
full_stack_dic = list(set([a_word for a_list in list(full_dict) for a_word in a_list]))
dictionary = gensim.corpora.Dictionary(processed_docs)
count = 0
for k, v in dictionary.iteritems():
    print(k, v)
    count += 1
    if count > 10:
        break
# the result suggests that the words has already lemmatized
print(data_text[4043])
print(processed_docs[4043])
dictionary = gensim.corpora.Dictionary(processed_docs)
# filter si very impar to acc"
dictionary = gensim.corpora.Dictionary(processed_docs)
dictionary.filter_extremes(no_above= 0.5)
bow_corpus = [dictionary.doc2bow(doc) for doc in processed_docs]
bow_lda = gensim.models.Lda(bow_corpus, num_topics=4,
                            id2word=dictionary, passes=2,
                            workers=31, iterations=5000,
                            random_state=42, alpha="auto", eta="auto")
for idx, topic in bow_lda.print_topics(-1):
    print('Topic: {} \nWords: {}'.format(idx, topic))
# LDA_list = list(bow_lda.print_topics(-1))
# list(LDA_list[0])[0]
# list(LDA_list[0])[1]
# LDA_list_split = []
# for i in LDA_list:
#     lst = list(i)[1]
#     LDA_list_split.append(lst.replace('"', "").split(" + "))
# # [a_word for a_row in LDA_list_split for a_word in a_row]
# LDA_out = []
```

```
# for lst in LDA_list_split:
#     for word in lst:
#         LDA_out.append(word.split("*"))
```

5. การวิเคราะห์ความนิยม

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import re
# % matplotlib inline
plt.style.use('ggplot')
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
import unicodedata
import contractions # signature
from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
# import model_evaluation_utils as meu # signature
np.set_printoptions(precision=2, linewidth=80)
import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from nltk.stem.porter import *
import pandas as pd
import numpy as np
np.random.seed(2018)
import nltk
```

```
nltk.download('wordnet')
"""## Create Norm df"""
word_df = pd.read_excel("LDA result.xlsx", header= 1)
word_list = word_df[["topic1_word", "topic2_word", "topic3_word",
"topic4_word"]].values.tolist()
word_list
words = [a_word for a_sentence in word_list for a_word in a_sentence]
words = list(set(words))
words
full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
full_df["sentiment"] = full_df["star"].apply(lambda star: 1 if star >= 40 else 0)
df = full_df[["content","sentiment"]]
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
train, test = train_test_split(df, test_size=0.30, random_state=42)
train_reviews = train['content']
train_sentiments = train["sentiment"]
test_reviews = test["content"]
test_sentiments = test["sentiment"]
# normalize datasets
stop_words = nltk.corpus.stopwords.words('english')
stop_words.remove('no')
stop_words.remove('but')
stop_words.remove('not')
norm_reviews = tn.normalize_corpus(df["content"], stopwords=stop_words)
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0)
from nltk.stem.porter import *
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
stemmer = PorterStemmer()
def preprocess(text):
    result = []
    full_dict = []
```

```
    for token in gensim.utils.simple_preprocess(text):
        # if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3
and \
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
        # \ and token not in ["chinatown", "bangkok", "china", "chinese", "place", "go",
"dont", "morning", "possible", "visit"]:
        # token != "chinatown" and token != "bangkok" and token != "china" and \
        # token != "chinese":
            result.append(lemmatize_stemming(token))
    return result
def all_dict(text):
    full_dict = []
    for token in gensim.utils.simple_preprocess(text):
        # if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3
and \
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
        #: and token not in ["chinatown", "bangkok", "china", "chinese", "place", "go",
"good"]:
        # token != "chinatown" and token != "bangkok" and token != "china" and \
        # token != "chinese":
            full_dict.append(token)
    return full_dict
data_text = full_df["content"]
processed_docs = data_text.map(preprocess)
# the result suggests that the words has already lemmatized
norm_df = pd.concat([pd.DataFrame(processed_docs), df["sentiment"]], axis=1)
norm_df.head()
norm_df["content"][7].count("go")
norm_word_lst = []
for i in words:
    norm = preprocess(i)
    norm_word_lst.append(norm)
norm_word_lst = [a_word for a_sen in norm_word_lst for a_word in a_sen]
```

```
# two words (go, lot) are disappear why??
norm_word_lst = norm_word_lst + ["go", "lot"]
for w in norm_word_lst:
    lst = []
    for row in norm_df["content"]:
        count = row.count(w)
        lst.append(count)
    norm_df[w] = pd.Series(lst)
# norm_df.to_excel("norm_df_for_VIZ.xlsx")
"""## filter Norm df preparation"""
word_df = pd.read_excel("LDA result.xlsx", header= 1)
full_norm_df = pd.read_excel("norm_df_for_VIZ.xlsx")
full_norm_df.columns
# # for K-means paper (No needd to do I was confused)
# word_df = pd.read_excel("K-means interpretation for paper.xlsx", header= 1)
# full_norm_df = pd.read_excel("dataset for K-means inferenece.xlsx")
# # for K-means paper (No needd to do I was confused)
# topic1 = [i.lower() for i in ["Food","Street","Place","Chines","Chinatown","Good", "Visit",
"Bangkok", "Great", "Shop"]]
# topic2 = [i.lower() for i in ["Chinatown", "Bangkok", "Shop", "Visit", "Food", "Street",
"Walk", "Place", "Market", "Chines"]]
# topic3 = [i.lower() for i in ["Food", "Place", "Shop", "Street", "Chinatown", "Chines",
"Visit", "Go", "Bangkok", "China"]]
# topic4 = [i.lower() for i in ["Place", "Shop", "China",
"Food","Town","Good","Visit","Chines","Market","Bangkok"]]
# all = topic1 + topic2 + topic3 + topic4
# all = list(set(all))
# all = ['Bangkok','Street','Place','Chinatown','Good','Market",
Chines','Go','Market','Great','Shop','Town','China','Chines','Food','Visit','Walk']
# norm_word_lst = []
# for i in all:
#     norm_word_lst.append(i.lower())
full_topic1 = full_norm_df[["content", "sentiment"] + topic1]
```

```
full_topic2 = full_norm_df[["content", "sentiment"] + topic2]
full_topic3 = full_norm_df[["content", "sentiment"] + topic3]
full_topic4 = full_norm_df[["content", "sentiment"] + topic4]
# topic1 = ['shop','street','cheap','area','good','walk','gold','food','time','visit']
# topic2 = ['street','walk','food','market','shop','go','town','peopl','time','night']
# topic3 = ['food','street','shop','great','good','town','market','lot','restaur','stall']
# topic4 = ['shop','food','good','street','great','market','road','restaur','stall','taxi']
# full_topic1 = full_norm_df[["content", "sentiment"] + topic1]
# full_topic2 = full_norm_df[["content", "sentiment"] + topic2]
# full_topic3 = full_norm_df[["content", "sentiment"] + topic3]
# full_topic4 = full_norm_df[["content", "sentiment"] + topic4]
def weight_columns(weight, table):
    weight_col = pd.DataFrame([weight]*table.shape[0])
    return weight_col
weight_lst = word_df['topic1_weight']
index = 0
for i in weight_lst:
    mod_i = np.round(i,4)
    name = topic1[index] + "_w"
    full_topic1[name] = weight_columns(mod_i, full_topic1)
    index += 1
weight_lst = word_df['topic2_weight']
index = 0
for i in weight_lst:
    mod_i = np.round(i,4)
    name = topic2[index] + "_w"
    full_topic2[name] = weight_columns(mod_i, full_topic2)
    index += 1
weight_lst = word_df['topic3_weight']
index = 0
for i in weight_lst:
    mod_i = np.round(i,4)
    name = topic3[index] + "_w"
```

```
    full_topic3[name] = weight_columns(mod_i, full_topic3)
    index += 1
weight_lst = word_df['topic4_weight']
index = 0
for i in weight_lst:
    mod_i = np.round(i,4)
    name = topic4[index] + "_w"
    full_topic4[name] = weight_columns(mod_i, full_topic4)
    index += 1
full_topic1.to_excel("Topic1_LDA_.xlsx", index=False)
full_topic2.to_excel("Topic2_LDA_.xlsx", index=False)
full_topic3.to_excel("Topic3_LDA_.xlsx", index=False)
full_topic4.to_excel("Topic4_LDA_.xlsx", index=False)
# for K-means paper (No needd to do I was confused)
# full_topic1.to_excel("Topic1_K_means_.xlsx", index=False)
# full_topic2.to_excel("Topic2_K_means_.xlsx", index=False)
# full_topic3.to_excel("Topic3_K_means_.xlsx", index=False)
# full_topic4.to_excel("Topic4_K_means_.xlsx", index=False)
```

6. การวิเคราะห์ความเด่นและความแพร่หลาย

```
word_df = pd.read_excel("LDA result.xlsx", header= 1)
full_norm_df = pd.read_excel("norm_df_for_VIZ.xlsx")
full_topic1 = pd.read_excel("Topic1_LDA_.xlsx")
full_topic2 = pd.read_excel("Topic2_LDA_.xlsx")
full_topic3 = pd.read_excel("Topic3_LDA_.xlsx")
full_topic4 = pd.read_excel("Topic4_LDA_.xlsx")
# # for K-means paper (No needd to do I was confused)
# word_df = pd.read_excel("K-means interpretation for paper.xlsx", header= 1)
# full_norm_df = pd.read_excel("dataset for K-means inferenece.xlsx")
# full_topic1 = pd.read_excel("Topic1_K_means_.xlsx")
# full_topic2 = pd.read_excel("Topic2_K_means_.xlsx")
# full_topic3 = pd.read_excel("Topic3_K_means_.xlsx")
# full_topic4 = pd.read_excel("Topic4_K_means_.xlsx")
topic1 = ['shop','street','cheap','area','good','walk','gold','food','time','visit']
topic2 = ['street','walk','food','market','shop','go','town','peopl','time','night']
topic3 = ['food','street','shop','great','good','town','market','lot','restaur','stall']
topic4 = ['shop','food','good','street','great','market','road','restaur','stall','taxi']
# for K-means paper (No needd to do I was confused)
# topic1 = [i.lower() for i in ["Food","Street","Place","Chines","Chinatown","Good", "Visit",
"Bangkok", "Great", "Shop"]]
# topic2 = [i.lower() for i in ["Chinatown", "Bangkok", "Shop", "Visit", "Food", "Street",
"Walk", "Place", "Market", "Chines"]]
# topic3 = [i.lower() for i in ["Food", "Place", "Shop", "Street", "Chinatown", "Chines",
"Visit", "Go", "Bangkok", "China"]]
# topic4 = [i.lower() for i in ["Place", "Shop", "China",
"Food","Town","Good","Visit","Chines","Market","Bangkok"]]
```

```
def multiply_weight(topic, table):
    temp = pd.DataFrame()
    for a_word in topic:
        weight_col = a_word + "_w"
        multiply = "mul_" + a_word
        temp[multiply] = table[a_word] * table[weight_col]
        temp["SUM_mul"] = temp.sum(axis=1)
    return temp
full_topic1["similarity_w*word"] = multiply_weight(topic1, full_topic1)["SUM_mul"]
full_topic2["similarity_w*word"] = multiply_weight(topic2, full_topic2)["SUM_mul"]
full_topic3["similarity_w*word"] = multiply_weight(topic3, full_topic3)["SUM_mul"]
full_topic4["similarity_w*word"] = multiply_weight(topic4, full_topic4)["SUM_mul"]
full_topic1["similarity_w*word"]
"""### Cosine similarity & Jackkard"""
# def filter_DSVA(topic_col, full_norm_df):

#     df_top = full_norm_df[["content", "sentiment"] + topic_col]
#     df_top["SUM_words_axis1"] = full_norm_df[topic_col].sum(axis=1)
#     # df_top = df_top[df_top["SUM_words_axis1"] != 0].reset_index(drop=True)

#     return df_top
# print("Gold shop", df_top1.shape)
# print("Night market", df_top2.shape)
# print("Steet food", df_top3.shape)
# print("Streetscape", df_top4.shape)
# %%capture
# !pip install textdistance
# import re
# from textdistance import *
# import numpy as np
# import pandas as pd
# import warnings
# warnings.filterwarnings('ignore')
```

```
# def topic_to_doc(topic):
#     doc = re.sub('[^a-zA-Z]+', ' ', str(topic))
#     return doc
# full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
# full_df["sentiment"] = full_df["star"].apply(lambda star: 1 if star >= 40 else 0)
# df = full_df[["content","sentiment"]]
# word_df = pd.read_excel("LDA result.xlsx", header= 1)
# ori_topic1 = list(word_df["topic1_word"])
# ori_topic2 = list(word_df["topic2_word"])
# ori_topic3 = list(word_df["topic3_word"])
# ori_topic4 = list(word_df["topic4_word"])
# doc1 = topic_to_doc(ori_topic1)
# doc2 = topic_to_doc(ori_topic2)
# doc3 = topic_to_doc(ori_topic3)
# doc4 = topic_to_doc(ori_topic4)
df = full_df[["content","sentiment"]]
docs = [doc1] + [doc2 ]+ [doc3] + [doc4]
docs
# df = full_df[["content","sentiment"]]
# for i, a_doc in enumerate(docs):
#     topic_index = "topic_" + str(i+1) + "_cosine"
#     df[topic_index] = df["content"].astype(str).apply(lambda a_comment:
cosine.normalized_similarity(a_comment, a_doc))
# for i, a_doc in enumerate(docs):
#     topic_index = "topic_" + str(i+1) + "_jaccard"
#     df[topic_index] = df["content"].astype(str).apply(lambda a_comment:
jaccard.normalized_similarity(a_comment, a_doc))
"""### combine 3 similarity"""
full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
full_df["sentiment"] = full_df["star"].apply(lambda star: 1 if star >= 40 else 0)
df = full_df[["content","sentiment"]]
lda_prob = pd.DataFrame()
lda_prob["topic1_lda_prob"] = full_topic1["similarity_w*word"]
```

```
lda_prob["topic2_lda_prob"] = full_topic2["similarity_w*word"]
lda_prob["topic3_lda_prob"] = full_topic3["similarity_w*word"]
lda_prob["topic4_lda_prob"] = full_topic4["similarity_w*word"]
# all_stat = pd.concat([df, lda_prob], axis=1)
lda_prob = pd.concat([df,lda_prob], axis=1)
lda_prob["max"] = lda_prob[["topic1_lda_prob", "topic2_lda_prob", "topic3_lda_prob",
"topic4_lda_prob"]].max(axis=1)
lda_prob = lda_prob[lda_prob["max"] != 0]
lda_prob.shape
def rep_of_each_model(table, MAX_col, topic1234_method):
    temp = pd.DataFrame()
    for i in topic1234_method:
        temp[i] = table[MAX_col] == table[i]
    return temp
lda_prob = lda_prob[['topic1_lda_prob', 'topic2_lda_prob',
'topic3_lda_prob','topic4_lda_prob']]
lda_prob["max"] = lda_prob.max(axis=1)
lda_prob_top = ['topic1_lda_prob','topic2_lda_prob', 'topic3_lda_prob',
'topic4_lda_prob']
all_lda_prob = rep_of_each_model(lda_prob, "max", lda_prob_top)
top1_lda_prob = all_lda_prob.iloc[:, [0]]
top2_lda_prob = all_lda_prob.iloc[:, [1]]
top3_lda_prob = all_lda_prob.iloc[:, [2]]
top4_lda_prob = all_lda_prob.iloc[:, [3]]
top1_lda_prob = top1_lda_prob[top1_lda_prob["topic1_lda_prob"]]
top2_lda_prob = top2_lda_prob[top2_lda_prob["topic2_lda_prob"]]
top3_lda_prob = top3_lda_prob[top3_lda_prob["topic3_lda_prob"]]
top4_lda_prob = top4_lda_prob[top4_lda_prob["topic4_lda_prob"]]
cond1 = {True: "Topic1"}
cond2 = {True: "Topic2"}
cond3 = {True: "Topic3"}
cond4 = {True: "Topic4"}
top1_lda_prob = pd.DataFrame(top1_lda_prob["topic1_lda_prob"].map(cond1))
```

```
top2_lda_prob = pd.DataFrame(top2_lda_prob["topic2_lda_prob"].map(cond2))
top3_lda_prob = pd.DataFrame(top3_lda_prob["topic3_lda_prob"].map(cond3))
top4_lda_prob = pd.DataFrame(top4_lda_prob["topic4_lda_prob"].map(cond4))
top1_lda_prob.rename(columns={"topic1_lda_prob": "final_lda_prob"}, inplace=True)
top2_lda_prob.rename(columns={"topic2_lda_prob": "final_lda_prob"}, inplace=True)
top3_lda_prob.rename(columns={"topic3_lda_prob": "final_lda_prob"}, inplace=True)
top4_lda_prob.rename(columns={"topic4_lda_prob": "final_lda_prob"}, inplace=True)
frames = [top1_lda_prob, top2_lda_prob, top3_lda_prob, top4_lda_prob]
final_lda_prob = pd.concat(frames)
lda_prob.shape
full_lda_prob = pd.merge(lda_prob, final_lda_prob, left_index=True,
right_index=True, how="outer")
full_lda_prob[full_lda_prob.index == 345]
count_dup = pd.DataFrame(pd.Series(list(final_lda_prob.index)).value_counts(),
columns=["count"])
count_dup[count_dup["count"] > 1]
count_dup[count_dup["count"] > 1].index
# full_lda_prob.to_excel("final_lda_prob_for_DSVA.xlsx")
# -*- coding: utf-8 -*-
```

7. การวิเคราะห์ตัวแทนของคำที่แทนความรู้สึก

```
# Commented out IPython magic to ensure Python compatibility.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
# % matplotlib inline
plt.style.use('ggplot')
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
import unicodedata
import contractions # signature
from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
# import model_evaluation_utils as meu # signature
np.set_printoptions(precision=2, linewidth=80)
from sklearn.ensemble import VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB as NaiveB
from sklearn.svm import LinearSVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score
full_df = pd.read_excel("tripadvisor_dataset_china_town.xlsx")
```

```
full_df["sentiment"] = full_df["star"].apply(lambda star: 1 if star >= 40 else 0)
df = full_df[["content","sentiment"]]
from sklearn.model_selection import train_test_split
train, test = train_test_split(df, test_size=0.30, random_state=42)
train_reviews = train['content']
train_sentiments = train["sentiment"]
test_reviews = test["content"]
test_sentiments = test["sentiment"]
# normalize datasets
stop_words = nltk.corpus.stopwords.words('english')
stop_words.remove('no')
stop_words.remove('but')
stop_words.remove('not')
# Full stack cleaning
norm_train_reviews = tn.normalize_corpus(train_reviews, stopwords=stop_words)
norm_test_reviews = tn.normalize_corpus(test_reviews, stopwords=stop_words)
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
# build BOW features on train reviews
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0) # ngram_range=(1,2)
cv_train_features = cv.fit_transform(norm_train_reviews)
cv_test_features = cv.transform(norm_test_reviews)
nb = NaiveB()
lr = LogisticRegression(penalty='l2', solver='liblinear')
RBF_svm = SVC(kernel="rbf", C = 2**1, gamma = 2**-5)
ensemble = VotingClassifier(estimators=[('Naive bayes', nb), ('Logistic regression', lr),
('RBF_SVM', RBF_svm)], voting='hard')
ensemble = ensemble.fit(cv_train_features, train_sentiments)
print(ensemble.predict(cv_train_features))
ensemble.score(cv_test_features, test_sentiments)
"""### Sentence preparation (separated by topic and sentiment)"""
final_df = pd.read_excel("final_dataset.xlsx")
top1P = final_df[(final_df["final_lda_prob"] == "Topic1") & (final_df["sentiment"] == 1)]
top1N = final_df[(final_df["final_lda_prob"] == "Topic1") & (final_df["sentiment"] == 0)]
```

```
top2P = final_df[(final_df["final_lda_prob"] == "Topic2") & (final_df["sentiment"] == 1)]
top2N = final_df[(final_df["final_lda_prob"] == "Topic2") & (final_df["sentiment"] == 0)]
top3P = final_df[(final_df["final_lda_prob"] == "Topic3") & (final_df["sentiment"] == 1)]
top3N = final_df[(final_df["final_lda_prob"] == "Topic3") & (final_df["sentiment"] == 0)]
top4P = final_df[(final_df["final_lda_prob"] == "Topic4") & (final_df["sentiment"] == 1)]
top4N = final_df[(final_df["final_lda_prob"] == "Topic4") & (final_df["sentiment"] == 0)]
print("top1P:", top1P.shape[0])
print("top1N:", top1N.shape[0])
print("top2P:", top2P.shape[0])
print("top2N:", top2N.shape[0])
print("top3P:", top3P.shape[0])
print("top3N:", top3N.shape[0])
print("top4P:", top4P.shape[0])
print("top4N:", top4N.shape[0])
top1P_lst = []; top1N_lst = []
top2P_lst = []; top2N_lst = []
top3P_lst = []; top3N_lst = []
top4P_lst = []; top4N_lst = []
all_topics = [top1P, top1N,
              top2P, top2N,
              top3P, top3N,
              top4P, top4N]
all_lst = [top1P_lst, top1N_lst,
           top2P_lst, top2N_lst,
           top3P_lst, top3N_lst,
           top4P_lst, top4N_lst]
for top, top_lst in zip(all_topics, all_lst):
    for i in top["content"]:
        sentence_lst = i.split(".")
        for sen in sentence_lst:
            top_lst.append(sen)
def clean_sen(top_lst):
    return [sen for sen in top_lst if sen]
```

```
def topic_clean(top, top_lst):
    top_lst = []
    for i in top["content"]:
        # sentence_num = len(i.split("."))
        sentence_lst = i.split(".")
        for sen in sentence_lst:
            top_lst.append(sen)
    return [sen for sen in top_lst if sen]
topic1_POS_df = pd.DataFrame(topic_clean(top1P, top1P_lst)); topic1_POS_df["topic"]
= "topic1"; topic1_POS_df["sentiment"] = 1
topic2_POS_df = pd.DataFrame(topic_clean(top2P, top2P_lst)); topic2_POS_df["topic"]
= "topic2"; topic2_POS_df["sentiment"] = 1
topic3_POS_df = pd.DataFrame(topic_clean(top3P, top3P_lst)); topic3_POS_df["topic"]
= "topic3"; topic3_POS_df["sentiment"] = 1
topic4_POS_df = pd.DataFrame(topic_clean(top4P, top4P_lst)); topic4_POS_df["topic"]
= "topic4"; topic4_POS_df["sentiment"] = 1
topic1_NEG_df = pd.DataFrame(topic_clean(top1N, top1N_lst)); topic1_NEG_df["topic"]
= "topic1"; topic1_NEG_df["sentiment"] = 0
topic2_NEG_df = pd.DataFrame(topic_clean(top2N, top2N_lst)); topic2_NEG_df["topic"]
= "topic2"; topic2_NEG_df["sentiment"] = 0
topic3_NEG_df = pd.DataFrame(topic_clean(top3N, top3N_lst)); topic3_NEG_df["topic"]
= "topic3"; topic3_NEG_df["sentiment"] = 0
topic4_NEG_df = pd.DataFrame(topic_clean(top4N, top4N_lst)); topic4_NEG_df["topic"]
= "topic4"; topic4_NEG_df["sentiment"] = 0
topic1_POS_df = pd.DataFrame(topic1_POS); topic1_POS_df["topic"] = "topic1";
topic1_POS_df["sentiment"] = 1
topic2_POS_df = pd.DataFrame(topic2_POS); topic2_POS_df["topic"] = "topic2";
topic2_POS_df["sentiment"] = 1
topic3_POS_df = pd.DataFrame(topic3_POS); topic3_POS_df["topic"] = "topic3";
topic3_POS_df["sentiment"] = 1
topic4_POS_df = pd.DataFrame(topic4_POS); topic4_POS_df["topic"] = "topic4";
topic4_POS_df["sentiment"] = 1
```

```
topic1_NEG_df = pd.DataFrame(topic1_NEG); topic1_NEG_df["topic"] = "topic1";
topic1_NEG_df["sentiment"] = 0
topic2_NEG_df = pd.DataFrame(topic2_NEG); topic2_NEG_df["topic"] = "topic2";
topic2_NEG_df["sentiment"] = 0
topic3_NEG_df = pd.DataFrame(topic3_NEG); topic3_NEG_df["topic"] = "topic3";
topic3_NEG_df["sentiment"] = 0
topic4_NEG_df = pd.DataFrame(topic4_NEG); topic4_NEG_df["topic"] = "topic4";
topic4_NEG_df["sentiment"] = 0
all_sentence = pd.concat([topic1_POS_df, topic2_POS_df, topic3_POS_df,
topic4_POS_df, topic1_NEG_df, topic2_NEG_df, topic3_NEG_df,
topic4_NEG_df]).rename(columns={0:"sentence"})
all_sentence = all_sentence[~all_sentence["sentence"].isin([" "])]
all_sentence.shape
all_sentence.to_excel("all_sentence.xlsx", index=False)
```

8. เว็ปแอพพลิเคชั่นสำหรับการวิเคราะห์กลุ่มความสนใจและความรู้สึก

```
import unicodedata
# import contractions # signature
# from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
# import model_evaluation_utils as meu # signature
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB as NaiveB
from sklearn.svm import LinearSVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import VotingClassifier
import gensim
from nltk.stem import WordNetLemmatizer
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem.porter import *
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
stemmer = PorterStemmer()
def preprocess(text):
    result = []
    full_dict = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
```

```
            result.append(lemmatize_stemming(token))
    return result
def similarity(text_input, topic, topic_w):
    sim = []
    for num, i in enumerate(topic):
        sim.append(text_input.count(i)*topic_w[num])
        sum_sim = sum(sim)
    return sum_sim
app = Flask(__name__)
@app.route('/')
def home():
        return render_template('home.html')
@app.route('/predict',methods=['POST'])
def predict():

############################## load pre-proessor ######################
# for pickle the BOW voolcab
# https://medium.com/velotio-perspectives/real-time-text-classification-using-kafka-and-scikit-learn-c2875ad80b3c
# for web app
# https://towardsdatascience.com/develop-a-nlp-model-in-python-deploy-it-with-flask-step-by-step-744f3bdd7776

    vocabulary_to_load = pickle.load(open("BOW_vocab.pickle", 'rb'))
    cv = CountVectorizer(vocabulary=vocabulary_to_load)

    stop_words = nltk.corpus.stopwords.words('english')
    stop_words.remove('no')
    stop_words.remove('but')
    stop_words.remove('not')

    sim1 = []
    topic1 = ['shop','street','cheap','area','good','walk','gold','food','time','visit']
```

```
topic1_w = [0.037, 0.023, 0.020, 0.019, 0.019, 0.017, 0.015, 0.014, 0.012, 0.011]

sim2 = []
topic2 = ['street','walk','food','market','shop','go','town','peopl','time','night']
topic2_w = [0.023, 0.020, 0.020, 0.019, 0.019, 0.015, 0.015, 0.013, 0.013, 0.013]

sim3 = []
topic3 = ['food','street','shop','great','good','town','market','lot','restaur','stall']
topic3_w = [0.065, 0.042, 0.024, 0.016, 0.016, 0.014, 0.013, 0.013, 0.012, 0.012]

sim4 = []
topic4 = ['shop','food','good','street','great','market','road','restaur','stall','taxi']
topic4_w = [0.024, 0.023, 0.020, 0.017, 0.015, 0.014, 0.013, 0.013, 0.012, 0.011]
# text_input = "test Yaowarat good great awesome"
# comment = [text_input]
# comm_df = pd.DataFrame(comment, columns=["input"])
# norm_comment = tn.normalize_corpus(comm_df, stopwords=stop_words)
# cv_comment = cv.transform(norm_comment)

ensemble_load = pickle.load(open("ensemble_save.sav", 'rb'))
if request.method == 'POST':
    message = request.form['message']
    data = [message]
    # comm_df = pd.DataFrame(data, columns=["input"])
    norm_comment = tn.normalize_corpus(data, stopwords=stop_words)
    cv_comment = cv.transform(norm_comment)
    # vect = cv.transform(data).toarray()
    prediction = int(ensemble_load.predict(cv_comment))
    if prediction == 1:
        prediction = 'Positive Sentiment and this comment belong to '
    else:
        prediction = 'Negative Sentiment and this comment belong to '
    ""
```

```
        processed_docs = pd.Series(str(data)).map(preprocess)
        text_input = processed_docs.tolist()
        text_input = [i for lst in text_input for i in lst]
        sim1 = similarity(text_input, topic1, topic1_w)
        sim2 = similarity(text_input, topic2, topic2_w)
        sim3 = similarity(text_input, topic3, topic3_w)
        sim4 = similarity(text_input, topic4, topic4_w)
        map1 = {sim1:"Grocery store", sim2:"Night street food market", sim3:"Food",
sim4:"Ambience"}
        result = "" + prediction + "" + "" + map1.get(max(map1)) + ' Dimension"'
        return result
    # return render_template('result.html',prediction = my_prediction)
if __name__ == '__main__':
        app.run(host="0.0.0.0", port=5000)

#### For Docker file ####
FROM python:3
RUN mkdir -p /usr/sec/app
WORKDIR /usr/src/app
COPY requirements.txt /usr/src/app
RUN pip install --no-cache-dir -r requirements.txt
RUN pip install -U spacy
RUN python -m spacy download en
COPY . /usr/src/app
#EXPOSE 5000
CMD ["python", "./app.py"]
#### requirement file.txt ####
flask
nltk
sklearn
gensim
spacy
pandas
```

```
### CSS file (static folder) ###
body{
        font:15px/1.5 Arial, Helvetica,sans-serif;
        padding: 0px;
        background-color:#f4f3f3;
}
.container{
        width:100%;
        margin: auto;
        overflow: hidden;
}
header{
        background:#03A9F4;#35434a;
        border-bottom:#448AFF 3px solid;
        height:120px;
        width:100%;
        padding-top:30px;
}
.main-header{
                text-align:center;
                background-color: blue;
                height:100px;
                width:100%;
                margin:0px;
        }
#brandname{
        float:left;
        font-size:30px;
        color: #fff;
        margin: 10px;
}
header h2{
        text-align:center;
```

```
        color:#fff;
}
.btn-info {background-color: #2196F3;
        height:40px;
        width:100px;} /* Blue */
.btn-info:hover {background: #0b7dda;}
.resultss{
        border-radius: 15px 50px;
    background: #345fe4;
    padding: 20px;
    width: 200px;
    height: 150px;
}
###### template folder ######

<!DOCTYPE html>
<html>
<head>
        <title>Home</title>
        <!-- <link rel="stylesheet" type="text/css" href="../static/css/styles.css"> -->
        <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='css/styles.css') }}">
</head>
<body>
        <header>
                <div class="container">
                <div id="brandname">
                        Machine Learning App
                </div>
                <h2>Attraction and Sentiment Analyzer</h2>
        </div>
        </header>
        <div class="ml-container">
```

```
                <form action="{{ url_for('predict')}}" method="POST">
                <p>Enter Your Message Here</p>
                <!-- <input type="text" name="comment"/> -->
                <textarea name="message" rows="4" cols="50"></textarea>
                <br/>
                <input type="submit" class="btn-info" value="predict">
        </form>
        </div>
</body>
</html>
```

9. การวิเคราะห์แนวโน้มของคำที่นักท่องเที่ยวใช้ในบทวิจารณ์

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
# % matplotlib inline
plt.style.use('ggplot')
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
import unicodedata
import contractions # signature
from contractions import CONTRACTION_MAP
import text_normalizer as tn # signature
# import model_evaluation_utils as meu # signature
np.set_printoptions(precision=2, linewidth=80)
from nltk.stem.porter import *
def word_occurrence(df, col):
    lst = []
    for i in df[col]:
```

```
        lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    return pd.DataFrame(pd.Series([w for a_lst in lst for w in a_lst]).value_counts())
def word_count(df, col):
    lst = []
    for i in df[col]:
        lst.append(i.replace('[', "").replace(']', "").replace("'", "").split(", "))
    return len(lst)
def differnce(pos, neg):
    return [i for i in pos if i not in neg]
df = pd.read_excel("all sentence analysis.xlsx")
topic1_pos = df["topic1_pos"].dropna().tolist()
topic2_pos = df["topic2_pos"].dropna().tolist()
topic3_pos = df["topic3_pos"].dropna().tolist()
topic4_pos = df["topic4_pos"].dropna().tolist()
topic1_neg = df["topic1_neg"].dropna().tolist()
topic2_neg = df["topic2_neg"].dropna().tolist()
topic3_neg = df["topic3_neg"].dropna().tolist()
topic4_neg = df["topic4_neg"].dropna().tolist()
differnce(topic1_pos + topic1_neg, topic2_pos + topic2_neg)
### Web application ###
import numpy as np
import pandas as pd
from flask import Flask,render_template,url_for,request
import nltk
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')

top1_lda_prob = all_lda_prob.iloc[:, [0]]
top2_lda_prob = all_lda_prob.iloc[:, [1]]
top3_lda_prob = all_lda_prob.iloc[:, [2]]
top4_lda_prob = all_lda_prob.iloc[:, [3]]
top1_lda_prob = top1_lda_prob[top1_lda_prob["topic1_lda_prob"]]
```

```
top2_lda_prob = top2_lda_prob[top2_lda_prob["topic2_lda_prob"]]
top3_lda_prob = top3_lda_prob[top3_lda_prob["topic3_lda_prob"]]
top4_lda_prob = top4_lda_prob[top4_lda_prob["topic4_lda_prob"]]
cond1 = {True: "Topic1"}
cond2 = {True: "Topic2"}
cond3 = {True: "Topic3"}
cond4 = {True: "Topic4"}
top1_lda_prob = pd.DataFrame(top1_lda_prob["topic1_lda_prob"].map(cond1))
top2_lda_prob = pd.DataFrame(top2_lda_prob["topic2_lda_prob"].map(cond2))
top3_lda_prob = pd.DataFrame(top3_lda_prob["topic3_lda_prob"].map(cond3))
top4_lda_prob = pd.DataFrame(top4_lda_prob["topic4_lda_prob"].map(cond4))
top1_lda_prob.rename(columns={"topic1_lda_prob": "final_lda_prob"}, inplace=True)
top2_lda_prob.rename(columns={"topic2_lda_prob": "final_lda_prob"}, inplace=True)
top3_lda_prob.rename(columns={"topic3_lda_prob": "final_lda_prob"}, inplace=True)
top4_lda_prob.rename(columns={"topic4_lda_prob": "final_lda_prob"}, inplace=True)
frames = [top1_lda_prob, top2_lda_prob, top3_lda_prob, top4_lda_prob]
topic1 = ['shop','street','cheap','area','good','walk','gold','food','time','visit']
topic2 = ['street','walk','food','market','shop','go','town','peopl','time','night']
topic3 = ['food','street','shop','great','good','town','market','lot','restaur','stall']
topic4 = ['shop','food','good','street','great','market','road','restaur','stall','taxi']
```

ประวัติผู้เขียน

ชื่อ	นายนิธิกร เลิศชาญวุฒิ
วัน เดือน ปีเกิด	20 พฤศจิกายน 2536
ที่อยู่ปัจจุบัน	23/28 ถนนข้าวหลาม แขวงตลาดน้อย เขตสัมพันธวงศ์ กรุงเทพมหานคร 10110
ประวัติการศึกษา	(2559) วิทยาศาสตรบัณฑิต สาขาวิชาฟิสิกส์ (สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง)
ผลงานวิชาการ	การวิเคราะห์ความสนใจของนักท่องเที่ยวด้วยวิธีการแบ่งกลุ่มข้อมูลแบบเคมีน กรณีศึกษา ถนนเยาวราช ประเทศไทย การประชุมวิชาการระดับชาติด้านธุรกิจ สารสนเทศและการจัดการ ครั้งที่ 3 วันที่ 16-18 กันยายน 2563