JRM JOURNAL OF RESEARCH METHODOLOGY

Journal of Research Methodology
Vol. 38 No. 1 (January–June 2025)
e-ISSN 2697-4835

Tutorial

Thai Text Clustering with K-Means and TF-IDF in Python for Educational Applications

Yotanut Boonyo[1]

Received: May 30, 2025 – Revised: June 21, 2025 – Accepted: June 23, 2025

Abstract

Qualitative text data—such as student feedback, interview transcripts, and online content—are increasingly available from web-based sources and institutional repositories. However, their unstructured nature makes large-scale analysis difficult. Text clustering helps organize such data by grouping documents with similar content. This tutorial presents a step-by-step workflow for clustering Thai-language texts using TF-IDF and the K-means algorithm in Python. It covers preprocessing, vectorization, clustering, and evaluation, with code examples based on Thai-language documents. The tutorial concludes with examples of educational applications, including analyzing open-ended survey responses, exploring curriculum topics, and identifying emerging themes in academic writing.

Keywords: Thai Text Clustering, Qualitative Data, TF-IDF, K-Means, Python, Text Mining

[1] *Corresponding Author,*
Ph.D. student in the Methodology for Innovation Development in Education program, Department of Educational Research and Psychology, Faculty of Education, Chulalongkorn University
yotanut.b@gmail.com

JRM JOURNAL OF RESEARCH METHODOLOGY

วารสารวิธีวิทยาการวิจัย
ปีที่ ๓๘ ฉบับที่ ๑ (มกราคม–มิถุนายน ๒๕๖๘)
e-ISSN 2697-4835

บทความแนะนำวิธีวิทยา

การจัดกลุ่มข้อความภาษาไทยด้วย K-Means และ TF-IDF ในไพทอนสำหรับการประยุกต์ทางการศึกษา

โยธณัฐ บุญโญ[1]

รับต้นฉบับ : 30 พฤษภาคม 2568 – รับแก้ไข : 21 มิถุนายน 2568 – ตอบรับตีพิมพ์ : 23 มิถุนายน 2568

บทคัดย่อ

ข้อมูลประเภทข้อความเชิงคุณภาพ เช่น ข้อเสนอแนะจากผู้เรียน บันทึกการสัมภาษณ์ และเนื้อหาออนไลน์ ในปัจจุบันสามารถเข้าถึงได้ง่ายมากขึ้น ทั้งจากแหล่งข้อมูลบนเว็บไซต์ และคลังข้อมูลของหน่วยงานต่าง ๆ อย่างไรก็ตาม ข้อมูลเหล่านี้เป็นข้อมูลที่ไม่มีโครงสร้าง จึงทำให้การวิเคราะห์ข้อมูลในระดับขนาดใหญ่เป็นไปได้ยาก การจัดกลุ่มข้อความ (text clustering) เป็นวิธีการที่ช่วยจัดการข้อมูลดังกล่าวโดยจัดกลุ่มเอกสารที่มีเนื้อหาคล้ายกันให้อยู่ในกลุ่มเดียวกัน บทช่วยสอน (tutorial) นี้นำเสนอขั้นตอนการจัดกลุ่มข้อความภาษาไทยทีละขั้นตอน โดยใช้เทคนิค TF-IDF ร่วมกับอัลกอริทึม K-means ด้วยภาษาไพทอน ซึ่งครอบคลุมกระบวนการตั้งแต่การเตรียมข้อมูลเบื้องต้น (preprocessing) การแปลงข้อความเป็นเวกเตอร์ (vectorization) การจัดกลุ่ม ไปจนถึงการประเมินผล พร้อมตัวอย่างโค้ดที่อ้างอิงจากเอกสารภาษาไทย และสรุปกรณีศึกษาการประยุกต์ใช้ในบริบททางการศึกษา เช่น การวิเคราะห์คำตอบแบบปลายเปิดจากแบบสำรวจ การสำรวจหัวข้อหลักสูตร และการระบุธีมในการเขียนเชิงวิชาการ

คำสำคัญ : การจัดกลุ่มข้อความภาษาไทย, ข้อมูลเชิงคุณภาพ, TD-IDF, K-means, ไพทอน, การทำเหมืองข้อความ

[1] *ผู้รับผิดชอบบทความหลัก,*
นิสิตระดับดุษฎีบัณฑิต สาขาวิชาวิธีวิทยาการพัฒนานวัตกรรมทางการศึกษา ภาควิชาวิจัยและจิตวิทยาทางการศึกษา คณะครุศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
yotanut.b@gmail.com

ความเป็นมาและความสำคัญ

ข้อมูลประเภทข้อความเป็นข้อมูลรูปแบบหนึ่งที่เพิ่มขึ้นอย่างรวดเร็ว (Mehta et al., 2024) ทั้งจากโซเชียลมีเดีย เช่น เอกซ์ (X) อินสตาแกรม (Instagram) และเฟซบุ๊ก (Facebook) ซึ่งมักจะอยู่ในรูปแบบข้อความสั้น (short text) และเป็นข้อมูลที่ไม่มีโครงสร้าง (unstructured data) (Ahmed et al., 2022) ขณะเดียวกันการจัดการเรียนรู้ในปัจจุบันมีการจัดการเรียนรู้โดยใช้เครื่องมือหรือแพลตฟอร์มออนไลน์ เช่น ระบบการจัดการเรียนรู้ (LMS) ซึ่งทำให้เกิดข้อมูลทั้งที่มีโครงสร้างและไม่มีโครงสร้างจำนวนมหาศาล เช่น ข้อมูลจากงานที่มอบหมายผ่านฟอรัมหรือการสนทนาผ่านแช็ต (Ferreira-Mello et al., 2019) ซึ่งการวิเคราะห์ข้อมูลเหล่านี้เพื่อค้นหาข้อมูลเชิงลึกที่มีความหมาย (meaningful insights) จึงเป็นสิ่งที่ท้าทาย

กระบวนการสกัดข้อมูลหรือความรู้ที่น่าสนใจ และมีคุณค่าจากข้อความที่ไม่มีโครงสร้างหรือการทำเหมืองข้อความ (text mining) (Luque et al., 2019) สามารถทำได้หลากหลายวิธีการ เช่น การจัดกลุ่มข้อความ (text clustering) การจัดประเภทข้อความ (text classification) การสรุปข้อความ (text summarization) โดยหนึ่งในวิธีการที่มีการใช้จำนวนมากคือ การจัดกลุ่มข้อความ เนื่องจากสามารถค้นหาโครงสร้างที่ซ่อนอยู่ในข้อมูลข้อความที่มีขนาดใหญ่และไม่มีโครงสร้าง (Petukhova et al., 2025)

การจัดกลุ่มข้อความเป็นกระบวนการค้นหากลุ่มของข้อความที่มีความคล้ายคลึงกันภายในชุดข้อมูล โดยวัดความคล้ายคลึงกันระหว่างเอกสารด้วยฟังก์ชันความคล้ายคลึง (similarity function) (Allahyari et al., 2017; Aggarwal & Zhai, 2012) โดยข้อความหรือเอกสารที่มีความคล้ายคลึงกันจะถูกจัดอยู่ในกลุ่มเดียวกัน (Bhattacharjee & Mitra, 2021) การจัดกลุ่มเป็นอัลกอริทึมการเรียนรู้แบบไม่มีผู้สอน (unsupervised learning) ซึ่งมุ่งทำความเข้าใจและวิเคราะห์โครงสร้างของข้อมูลโดยไม่มีตัวแปรตอบสนอง (response variable) หรือป้ายกำกับ (label) (Timbers et al., 2024) และมีลักษณะเชิงสำรวจ (exploratory) เพื่อค้นหาโครงสร้างหรือรูปแบบในข้อมูล (Jain, 2010) แต่ในปัจจุบันการจัดกลุ่มไม่ได้จำกัดเพียงการเรียนรู้แบบไม่มีผู้สอน

ก่อนการจัดกลุ่มข้อความจะต้องแปลงข้อความหรือเอกสารให้อยู่ในรูปแบบที่เหมาะสม ไม่สามารถใช้ข้อมูลดิบ (raw text) ได้โดยตรง ซึ่งเป็นกระบวนการที่เรียกว่า การแทนข้อความ (text representation) โดยการแทนข้อความที่นิยมคือ vector space model (Mehta et al., 2024) ในปัจจุบันมีวิธีการอื่น ๆ ที่มีความซับซ้อนมากขึ้น เช่น การใช้เวกเตอร์แทนคำ (word embeddings) ที่เรียนรู้จากข้อมูลขนาดใหญ่ และการนำโมเดล Transformer มาสร้างเวกเตอร์เชิงบริบท (contextualized embeddings) ที่ปรับเปลี่ยนตามประโยคซึ่งช่วยให้การจับความหมายของข้อความแม่นยำขึ้น และผลลัพธ์การจัดกลุ่มมีประสิทธิภาพมากขึ้น

บทช่วยสอน (tutorial) นี้มีจุดมุ่งหมายเพื่ออธิบายความรู้พื้นฐานเกี่ยวกับการจัดกลุ่มข้อความอย่างเป็นลำดับ โดยจะครอบคลุมแนวคิดเบื้องต้นของการจัดกลุ่มข้อความ ลักษณะเฉพาะของข้อมูลประเภทข้อความซึ่งเป็นการแปลงข้อความเป็นเวกเตอร์ อัลกอริทึมการจัดกลุ่มข้อพื้นฐาน เช่น k-means, hierarchical clustering และ DBSCAN เกณฑ์ประเมินผล รวมทั้งแสดงตัวอย่างการวิเคราะห์จัดกลุ่มข้อความภาษาไทยตั้งแต่การเตรียมข้อมูลไปจนถึงการตีความผลลัพธ์โดยใช้ Python เนื่องจากมีไลบรารีที่สามารถประมวลผลภาษาไทย

ได้ดีและได้รับความนิยม นอกจากนี้ยังสรุปตัวอย่างกรณีศึกษาการประยุกต์ใช้ในบริบททางการศึกษา เพื่อให้ผู้ที่สนใจสามารถประยุกต์ใช้เทคนิคนี้กับปัญหาวิจัยทางการศึกษาที่สนใจได้

วัตุประสงค์ของบทความ

1. เพื่อแนะนำแนวคิดพื้นฐาน หลักการ และขั้นตอนการจัดกลุ่มข้อความภาษาไทย (Thai text clustering) อย่างง่าย

2. เพื่อนำเสนอตัวอย่างขั้นตอนการวิเคราะห์และแปลผลการจัดกลุ่มข้อความภาษาไทยโดยใช้ K-Means และ TF-IDF ด้วย Python

แนวคิดการจัดกลุ่มข้อความ

การจัดกลุ่ม (clustering) คือกระบวนการค้นหากลุ่มของวัตถุ (objects) (Aggarwal & Zhai, 2012) หรือโครงสร้างกลุ่ม (Sinaga & Yang, 2020) ที่มีความคล้ายคลึงกันภายในชุดข้อมูล โดยวัดความคล้ายคลึงกันระหว่างเอกสารด้วยฟังก์ชันความคล้ายคลึง (similarity function) (Aggarwal & Zhai, 2012; Allahyari et al., 2017) ในบริบทของข้อมูลข้อความ (text domain) การจัดกลุ่มสามารถนำไปใช้กับข้อมูลในหลายระดับ เช่น เอกสาร (documents) ย่อหน้า (paragraphs) ประโยค (sentences) และคำ (terms) (Aggarwal & Zhai, 2012)

การจัดกลุ่มเป็นอัลกอริทึมที่ใช้แบ่งชุดข้อมูลออกเป็นกลุ่มย่อยที่มีความสัมพันธ์กัน (Timbers et al., 2024) โดยสมาชิกภายในกลุ่มมีความคล้ายคลึงกันสูงสุดและสมาชิกระหว่างกลุ่มมีความต่างกันมากที่สุด (Sinaga & Yang, 2020) ดังนั้นเอกสารที่อยู่ในกลุ่มเดียวกันจะมีความคล้ายคลึงกันสูงสุด ส่วนเอกสารที่อยู่คนละกลุ่มจะมีความแตกต่างกัน (Bhattacharjee & Mitra, 2021)

การจัดกลุ่มโดยทั่วไปจะเป็นการเรียนรู้แบบไม่มีผู้สอน (unsupervised Learning) ซึ่งไม่ต้องใช้ข้อมูลที่มีป้ายกำกับ (labels) ล่วงหน้าในจัดข้อความเข้าสู่กลุ่มต่าง ๆ (Aggarwal & Zhai, 2012; Jain, 2010; Timbers et al., 2024) แตกต่างกับการจำแนกประเภท (classification) ซึ่งเป็นการเรียนรู้แบบมีผู้สอน (supervised Learning) ซึ่งใช้ชุดข้อมูลที่มีป้ายกำกับเพื่อฝึกโมเดลให้สามารถระบุหมวดหมู่ของข้อความใหม่ได้ตามคุณลักษณะที่กำหนดไว้ล่วงหน้า (Aggarwal & Zhai, 2012) แต่ในปัจจุบันทั้งการจัดกลุ่มและการจัดประเภทสามารถเป็นได้ทั้งประเภทการเรียนรู้แบบมีผู้สอนหรือไม่มีผู้สอน รวมทั้งการเรียนรู้แบบกึ่งมีผู้สอน (semi-supervised learning)

การจัดกลุ่มมีวัตถุประสงค์เพื่อ 1) ค้นหาโครงสร้างภายในข้อมูล (underlying structure) โดยช่วยสร้างข้อสมมุติ ตรวจจับความผิดปกติ และระบุคุณลักษณะสำคัญของข้อมูล 2) แบ่งกลุ่มตามธรรมชาติ (natural classification) โดยการจัดประเภทสิ่งมีชีวิตเชิงอนุกรมวิธาน และ 3) ย่อหรือสรุปข้อมูล (compression) โดยใช้ตัวแทนกลุ่ม (prototypes) แทนชุดข้อมูลขนาดใหญ่ (Jain, 2010)

การแทนข้อความ (text representation)

การประมวลผลภาษาด้วยวิธีทางสถิติจะต้องแปลงข้อความให้อยู่ในรูปของตัวเลข โดยแต่ละคำจะมีเวกเตอร์แทนค่า และเมื่อรวมข้อความทั้งหมดเข้าด้วยกันจะได้เป็นเมทริกซ์ที่ใช้แทนชุดข้อมูล ซึ่งช่วยจัดการกับข้อความที่มีความยาวแตกต่างกันให้มีขนาดคงที่และทำให้สามารถนำความรู้ทางพีชคณิตเชิงเส้นมาใช้ในการจัดการเวกเตอร์ คำนวณระยะห่าง และความคล้ายคลึงกันได้ง่ายขึ้น โดยการแทนข้อความ (text representation) เป็นกระบวนการแปลงข้อมูลข้อความดิบ (input text) ให้อยู่ในรูปแบบเชิงตัวเลข เช่น เวกเตอร์หรือเมทริกซ์ (Patil et al., 2024) ซึ่งก่อนการจัดกลุ่มข้อความจะต้องทำการแทนข้อความให้อยู่ในรูปแบบที่เหมาะสม

การแทนข้อความเริ่มจากกระบวนการเตรียมข้อมูลเบื้องต้น (preprocessing) เช่น การลบคำที่ไม่สำคัญ (stopwords) และการปรับให้คำอยู่ในรูปแบบพื้นฐานโดยวิธีการ stemming หรือ lemmatization (Mehta et al., 2024; Patil et al., 2024) สำหรับการประมวลผลธรรมชาติภาษาไทยจะมีไลบรารี PyThaiNLP ซึ่งพัฒนาด้วยภาษา Python และเปิดให้ใช้งานได้ฟรี (open-source) โดยภายในไลบรารีจะประกอบด้วยซอฟต์แวร์ โมเดลที่ได้รับการฝึกไว้ล่วงหน้า และชุดข้อมูลต่าง ๆ ที่รองรับงานด้านภาษาไทยหลากหลายรูปแบบ เช่น การตัดคำออกเป็นหน่วยย่อย (tokenization) PyThaiNLP รองรับอัลกอริทึมการตัดคำหลายรูปแบบ โดยอัลกอริทึมเริ่มต้นคือ newmm ซึ่งใช้วิธี dictionary-based maximum matching ของ Sornlertlamvanich (1993) ร่วมกับ Thai character cluster ของ Theeramunkong et al. (2000) (Phatthiyaphaibun et al., 2023) นอกจากนี้ ยังมีวิธีการตัดคำภาษาไทยอื่น ๆ เช่น AttaCut ซึ่งเป็นการตัดคำโดยใช้โมเดลการเรียนรู้เชิงลึก (Chormai et al., 2019)

ข้อมูลข้อความที่ผ่านการประมวลผลแล้วจะนำมาสร้างคลังคำศัพท์ (vocabulary) ซึ่งประกอบด้วยคำไม่ซ้ำทั้งหมดที่ใช้ในชุดเอกสาร จากนั้นจะแปลงข้อความเป็นเวกเตอร์ โดยสามารถทำได้หลายเทคนิควิธี เช่น one-hot encoding (OHE) และ bag-of-words โดยในบทความนี้จะอธิบายเทคนิคที่นิยมใช้คือ bag-of-words (BoW)

bag-of-words (BoW) คือเทคนิคในการแทนข้อความให้อยู่ในรูปแบบเมทริกซ์ โดยแต่ละแถวของเมทริกซ์แทนประโยคหรือเอกสารในชุดข้อมูล และแต่ละคอลัมน์แทนคำแต่ละคำในคลังคำศัพท์ องค์ประกอบในเมทริกซ์จะแสดงค่าความถี่ของการปรากฏของคำในแต่ละประโยคหรือเอกสาร (non-binary BoW) หรืออาจใช้ค่าทวิภาค (0 หรือ 1) เพื่อระบุการมีอยู่หรือไม่ปรากฏของคำ (binary BoW) แต่ข้อจำกัดสำคัญคือ BoW ไม่สามารถจับบริบทหรือความสัมพันธ์เชิงลำดับของคำได้ จึงมักนำไปใช้กับงานที่เน้นคุณลักษณะเชิงคำเดี่ยว (Patil et al., 2024)

ในการจำแนกหรือจัดกลุ่มเอกสาร การเปรียบเทียบความคล้ายคลึงกันระหว่างเอกสารมักอาศัยการแทนเอกสารเป็นเวกเตอร์ที่บ่งชี้ความสำคัญของแต่ละคำ ซึ่งนิยมใช้ค่าความถี่ของคำในเอกสาร (term frequency: TF) แต่เนื่องจาก TF ให้น้ำหนักเท่ากันกับทุกคำ จึงไม่สามารถสะท้อนความสำคัญเชิงเนื้อหาได้ เช่น คำทั่ว ๆ ไปหรือ stopwords ที่ปรากฏบ่อยแต่ไม่ช่วยบ่งชี้หัวข้อหลัก ดังนั้น จึงจำเป็นต้องใช้วิธีให้คะแนนคำ (term scoring) ที่สามารถปรับน้ำหนักตามความเกี่ยวข้องของแต่ละคำต่อเอกสาร เพื่อให้เวกเตอร์สะท้อนความสำคัญของคำที่เกี่ยวข้องกับเนื้อหาได้อย่างเหมาะสมและแม่นยำยิ่งขึ้น (Mehta et al., 2024)

term frequency-inverse document frequency (TF–IDF) เป็นวิธีการให้น้ำหนักคำในเอกสารเชิงสถิติที่ถูกนำมาใช้เพื่อเพิ่มความแม่นยำในการสืบค้นข้อมูล โดยประกอบด้วยสององค์ประกอบหลักคือ term frequency (TF) ซึ่งวัดความถี่ของคำหนึ่ง ๆ ในเอกสาร และ inverse document frequency (IDF) ซึ่งถ่วงน้ำหนักตามความหายากของคำนั้นในคลังเอกสารทั้งหมด (Salton & Buckley, 1998)

หลังจากที่แปลงข้อความเป็นเวกเตอร์ แต่ละเอกสารถูกแทนด้วยเวกเตอร์ที่มีมิติเท่ากับจำนวนคำทั้งหมดในคลังศัพท์ จะทำให้ได้แบบจำลองปริภูมิเวกเตอร์ (vector space model) ในรูปเมทริกซ์ขนาด M x N โดยที่ M คือจำนวนคำในคลังคำศัพท์ และ N คือจำนวนเอกสารในชุดข้อมูล โดยเมทริกซ์นี้จะเรียกว่า term-document matrix หรือ document-term matrix (Patil et al., 2024)

อัลกอริทึมที่ใช้ในการจัดกลุ่มข้อความ

หลังจากการแปลงข้อความให้อยู่ในรูปของข้อมูลเชิงตัวเลข การจัดกลุ่มสามารถใช้เทคนิคการจัดกลุ่มแบบดั้งเดิมมาประยุกต์ใช้ภายใต้โครงสร้างของ vector space model (Mehta et al., 2024) อัลกอริทึมการจัดกลุ่มข้อความแบ่งออกเป็นหลายประเภทหลัก เช่น

1. การจัดกลุ่มแบบแบ่งส่วน (partitioning algorithms) เป็นวิธีจัดกลุ่มชุดข้อมูล N จุด ให้เป็น K กลุ่ม โดยแต่ละจุดข้อมูลจะอยู่ในกลุ่มเดียวเท่านั้น และให้ผลลัพธ์คือพาร์ติชันเดียว (single partition) ของชุดข้อมูลทั้งหมด (Jain & Dubes, 1988) โดยอัลกอริทึมที่ได้รับความนิยม เช่น k-means และ k-medoids หรือ partitioning around medoids (PAM) ซึ่งการจัดกลุ่มประเภทนี้เหมาะสมกับข้อมูลที่มีขนาดใหญ่ เนื่องจาก ใช้เวลาในการคำนวณค่อนข้างต่ำ และประสิทธิภาพสูงกับข้อมูลที่มีโครงสร้างกลม (spherical clusters) (Xu & Tian, 2015)

2. การจัดกลุ่มตามลำดับชั้น (hierarchical clustering) เป็นการจัดกลุ่มตามข้อมูลแบบลำดับชั้น โดยแต่ละจุดข้อมูลเริ่มต้นถือเป็นกลุ่มย่อย (singleton) แล้วอาศัยการคำนวณระยะหรือตัวชี้วัดความไม่คล้ายคลึงระหว่างกลุ่ม มารวมกลุ่มที่ใกล้กันเข้าด้วยกันทีละขั้น (agglomerative) หรือแยกกลุ่มใหญ่ให้เล็กลงทีละขั้น (divisive) จนได้กลุ่มเดียวสุดท้าย ซึ่งผลลัพธ์ทั้งหมดสามารถแสดงในรูป dendrogram (Murtagh & Contreras, 2012; Xu & Tian, 2015) การจัดกลุ่มประเภทนี้เหมาะสำหรับการสำรวจโครงสร้างของข้อมูลที่เป็นลำดับชั้น (hierarchical relationships) (Xu & Tian, 2015; Xu & Wunsch, 2005)

3. การจัดกลุ่มตามความหนาแน่น (density-based clustering) เป็นการจัดกลุ่มข้อมูลโดยอาศัยความหนาแน่น (density) ของจุดข้อมูลในพื้นที่ ซึ่งข้อมูลที่อยู่ในพื้นที่ความหนาแน่นสูง (high-density regions) จะถูกจัดให้เป็นกลุ่มเดียวกัน โดยไม่ต้องกำหนดรูปร่างของกลุ่มล่วงหน้า (Xu & Tian, 2015) โดยอัลกอริทึมจะค้นหาจุดหลัก (core points) ที่มีจุดรอบข้างเพียงพอแล้วขยายออกเป็นกลุ่มโดยอัตโนมัติ ทำให้ค้นหากลุ่มที่มีรูปร่างอิสระ และจัดการข้อมูลรบกวน (noise) ได้ดี (Kriegel et al, 2011) ดังนั้น การจัดกลุ่มประเภทนี้เหมาะสำหรับข้อมูลที่มีรูปร่างอิสระ (arbitrary shapes) และมีข้อมูลรบกวน (noise) (Xu & Tian, 2015; Xu & Wunsch, 2005)

จากทฤษฎีความเป็นไปไม่ได้ (impossibility theorem) ของ Kleinberg (2002) ไม่มีอัลกอริทึมใดที่จะตอบสนองเงื่อนไขพื้นฐานทั้งสาม ได้แก่ scale-invariance, richness และ consistency ได้พร้อมกัน ดังนั้นในการเลือกใช้งานอัลกอริทึมจัดกลุ่ม ผู้ใช้จึงต้องทดลองหลายวิธีและเลือกวิธีที่เหมาะสมกับข้อมูลและวัตถุประสงค์ (Jain, 2010)

การจัดกลุ่มข้อความในปัจจุบันมีการพัฒนาอัลกอริทึมต่าง ๆ เพื่อให้สามารถจัดกลุ่มข้อความได้มีประสิทธิภาพมากขึ้น เช่น

1. ใช้ตัวแทนเชิงบริบท (contextual representation) จากโมเดลภาษาที่ฝึกฝนไว้ล่วงหน้า (pre-trained language models) เช่น BERT (Devlin et al., 2019) ซึ่งสามารถจับความสัมพันธ์เชิงความหมายในประโยคได้ดีกว่าเวกเตอร์แบบดั้งเดิม ซึ่งช่วยให้การวัดระยะห่างระหว่างเอกสารสะท้อนความใกล้เคียงเชิงความหมายได้แม่นยำขึ้น

2. ใช้อัลกอริทึมการจัดกลุ่มเชิงลึก (deep clustering) เช่น deep embedded clustering (DEC) เป็นการจัดกลุ่มที่เรียนรู้ตัวแทนและจัดกลุ่มร่วมกัน (joint representation and clustering learning) โดย DEC ทำการปรับค่า (optimize) พารามิเตอร์ของโครงข่ายประสาทเทียมร่วมกับตำแหน่งเซนทรอยด์ของกลุ่มใน latent space พร้อมกัน ซึ่งต่างจากวิธีทั่วไปที่มักแยกขั้นตอนเรียนรู้ตัวแทน (feature learning) กับการจัดกลุ่มออกจากกัน (Xie et al., 2016) นอกจากนี้ยังมีอัลกอริทึมอื่น ๆ เช่น variational deep embedding (VaDE) (Jiang et al., 2016) structural deep clustering network (SDCN) (Bo et al., 2020)

3. ใช้อัลกอริทึมผสมผสาน เช่น hierarchical density-based spatial clustering of applications with noise (HDBSCAN) ซึ่งเป็นอัลกอริทึมที่รวมข้อดีของการจัดกลุ่มตามความหนาแน่นร่วมกับการจัดกลุ่มตามลำดับชั้น ทำให้สามารถตรวจจับกลุ่มที่มีความหนาแน่นหลากหลายหรือซับซ้อนได้ดีกว่า DBSCAN (Campello et al., 2013)

การประเมินประสิทธิภาพการจัดกลุ่มข้อความ

ตัวชี้วัดที่ใช้ประเมินประสิทธิภาพของอัลกอริทึมการจัดกลุ่มสามารถแบ่งออกเป็น 2 ประเภทหลัก ได้แก่

1. ตัวชี้วัดภายใน (internal metrics) เป็นตัวชี้วัดที่ไม่จำเป็นต้องใช้ป้ายกำกับจริงในการประเมินผล โดยจะประเมินคุณภาพจากลักษณะของข้อมูลภายในกลุ่ม เช่น ระยะห่างระหว่างกลุ่ม (separation) และความหนาแน่นของข้อมูลภายในแต่ละกลุ่ม (density) ซึ่งตัวชี้วัดประเภทนี้เหมาะสำหรับกรณีที่ไม่มีข้อมูลอ้างอิงหรือป้ายกำกับจริงให้เปรียบเทียบ (Mehta et al., 2024) ตัวอย่างตัวชี้วัด เช่น silhouette coefficient ซึ่งใช้ประเมินความเหมาะสมของการจัดแต่ละตัวอย่างให้อยู่ในกลุ่ม โดยอาศัยระยะห่างระหว่างจุดข้อมูล ซึ่งหาได้จากสูตร $\frac{b-a}{max(a,b)}$ โดย a คือระยะเฉลี่ยภายในกลุ่ม และ b คือระยะเฉลี่ยไปยังกลุ่มที่ใกล้ที่สุด (Shahapure & Nicholas, 2020) โดยค่า s จะมีค่าอยู่ในช่วง [1, -1] หากค่า s มีค่าเข้าใกล้ +1 จะหมายถึงตัวอย่างอยู่ใกล้จุดศูนย์กลาง (centroid) ของกลุ่มตัวเองมากกว่ากลุ่มอื่น ๆ แสดงว่าการจัดกลุ่มถูกต้องและชัดเจน นอกจากนี้ยังมีตัวชี้วัดอื่น ๆ เช่น Davies–Bouldin index (DBI) และ Calinski–Harabasz index (CHI)

2. ตัวชี้วัดภายนอก (external metrics) เป็นตัวชี้วัดที่จำเป็นต้องมีป้ายกำกับจริง (ground truth labels) ของแต่ละจุดข้อมูล โดยใช้ป้ายกำกับเพื่อเปรียบเทียบผลลัพธ์ของการจัดกลุ่มกับคำตอบที่ถูกต้อง ซึ่งตัวชี้วัดประเภทนี้จะใช้ในกรณีที่ทราบหมวดหมู่จริงอยู่แล้ว ตัวอย่างตัวชี้วัด adjusted rand index (ARI) และความบริสุทธิ์ (purity) (Mehta et al., 2024) โดย ARI เป็นตัวชี้วัดความสอดคล้องระหว่างสองพาร์ติชัน (partitions) ที่ปรับค่าความคาดหวังจากการสุ่ม (corrected for chance) (Hubert & Arabie, 1985) และความบริสุทธิ์พื้นฐาน (simple purity) เป็นการประเมินสัดส่วนของตัวอย่างในแต่ละกลุ่มที่มีป้ายกำกับตรงกับคลาสส่วนใหญ่ของกลุ่มนั้น ๆ เมื่อเทียบกับจำนวนตัวอย่างทั้งหมด (Forestier et al., 2010)

นอกจากนี้ Hassan et al. (2024) ยังแบ่งกลุ่มตัวชี้วัดที่ใช้ประเมินประสิทธิภาพการจัดกลุ่มเพิ่มเติม โดยแบ่งออกเป็นตัวชี้วัดภายในที่เสนอใหม่ (newly proposed internal) ซึ่งพัฒนาขึ้นหลังปี 2014 เช่น clustering validation based on nearest neighbor (CVNN) และ kernel coverage estimator (KCE)

การประยุกต์ใช้การจัดกลุ่มข้อความในบริบททางการศึกษา

ข้อมูลข้อความทางการศึกษาเป็นข้อมูลที่สามารถเข้าถึงได้จากหลายแหล่ง เช่น แบบสอบถามความคิดเห็น หรือข้อมูลจากเอกสารต่าง ๆ ซึ่งสามารถนำข้อความเหล่านี้มาจัดกลุ่มเพื่อสกัดประเด็นสำคัญและค้นหารูปแบบเชิงลึกได้อย่างรวดเร็ว นอกจากผู้สอนจะนำไปใช้ในชั้นเรียนแล้ว ผู้ที่เกี่ยวข้องกับการศึกษา เช่น ผู้รับผิดชอบหลักสูตร ผู้บริหาร และผู้กำหนดนโยบายทางการศึกษา และนักวิจัยทางการศึกษายังสามารถนำไปประยุกต์ใช้ได้ดังตัวอย่าง

1. การจัดกลุ่มงานเขียน

วิเคราะห์เรียงความหรือบทความของผู้เรียนเพื่อจัดเป็นกลุ่มตามหัวข้อ สไตล์การเขียน หรือจุดบกพร่องที่พบร่วมกัน ซึ่งช่วยให้ผู้สอนสามารถตรวจและให้ข้อเสนอแนะได้รวดเร็วขึ้น ลดอคติ และเพิ่มความเท่าเทียมในการประเมิน

2. การจัดกลุ่มระดับความเข้าใจของผู้เรียน

วิเคราะห์คำตอบที่ผู้เรียนเขียนในข้อสอบอัตนัย เพื่อแบ่งกลุ่มตามระดับความเข้าใจในเนื้อหา เช่น กลุ่มที่มีความเข้าใจสูง ปานกลาง และต่ำ หรือตามความเข้าใจผิดที่ผู้เรียนแสดงออกมา ซึ่งข้อมูลนี้ช่วยให้ผู้สอนสามารถออกแบบกิจกรรมเสริมให้เหมาะสมกับระดับการเรียนรู้ของผู้เรียนแต่ละกลุ่มได้อย่างตรงจุด

3. การจัดกลุ่มความสนใจและรูปแบบการเรียนรู้

จัดกลุ่มโพสต์ในบอร์ดหรือผลสำรวจความสนใจของผู้เรียน เพื่อระบุกลุ่มที่มีความชอบหรือสไตล์การเรียนรู้ใกล้เคียงกัน จากนั้นนำข้อมูลไปสร้างระบบแนะนำกิจกรรมหรือเส้นทางการเรียนรู้ที่เหมาะสมกับแต่ละกลุ่มได้ ซึ่งผู้เรียนจะได้เรียนรู้ที่ตรงกับความสามารถของตนเองมากขึ้น

4. การจัดกลุ่มหัวข้ออภิปรายในหลักสูตรออนไลน์

จัดกลุ่มกระทู้หรือโพสต์ในชั้นเรียนออนไลน์ เช่น MOOCs หรือ LMS Forum เพื่อให้ผู้สอนสามารถจัดการกับกระทู้คำถามหรือหัวข้ออภิปรายได้ง่ายขึ้น ซึ่งอาจนำไปพัฒนาระบบตอบคำถามอัตโนมัติ

5. การจัดกลุ่มข้อความแสดงความรู้สึกในการเรียน

วิเคราะห์ข้อความที่ผู้เรียนเขียนสะท้อนเกี่ยวกับความรู้สึกในการเรียน เพื่อจัดกลุ่มปัญหาด้านอารมณ์ ความเครียด หรือแรงจูงใจในการเรียน และนำข้อมูลไปวางแผนกิจกรรมสนับสนุนทางจิตวิทยาหรือกิจกรรมสร้างความสัมพันธ์ในชั้นเรียน

6. การจัดกลุ่มความคิดเห็นเกี่ยวกับการจัดการเรียนการสอน

รวบรวมข้อเสนอแนะจากแบบสอบถามประเมินการสอน โพสต์ในบอร์ด และคอมเมนต์ในชั้นเรียนออนไลน์ ซึ่งช่วยให้ผู้สอนเห็นภาพรวมความพึงพอใจ จุดเด่น หรือจุดปรับปรุงสำคัญได้อย่างรวดเร็ว นำไปสู่การปรับเนื้อหา กิจกรรม และวิธีการสอนให้ตอบโจทย์ผู้เรียนได้ตรงจุด โดยธีมที่มักปรากฏจากการจัดกลุ่มความคิดเห็นเกี่ยวกับการจัดการเรียนการสอน เช่น รูปแบบกิจกรรมการเรียนรู้ สื่อและเอกสารประกอบการสอน วิธีการวัดผลและประเมินผล ซึ่งผู้สอนสามารถนำไปออกแบบหรือปรับหลักสูตรได้ ตัวอย่าง ในกลุ่มของรูปแบบกิจกรรมการเรียนรู้ หากในกลุ่มมีคำสำคัญที่เสนอแนะว่า “อยากให้มีกิจกรรม” และ “น่าเบื่อ” ผู้สอนสามารถนำไปปรับรูปแบบการเรียนรู้ของรายวิชาให้มีกิจกรรมมากขึ้น

7. การวิเคราะห์และพัฒนาหลักสูตร

ผู้ที่รับผิดชอบหลักสูตรสามารถจัดกลุ่มข้อความหรือเอกสารกับหลักสูตรเพื่อหาความเชื่อมโยง ช่องว่าง หรือความซ้ำซ้อน โดยนำเอกสารต่างๆ เช่น คำอธิบายรายวิชา หรือผลการเรียนรู้ที่คาดหวังจากหลาย ๆ วิชา มาจัดกลุ่ม ซึ่งอาจพบว่ามีกลุ่มรายวิชาที่สอนเนื้อหาใกล้เคียงกันมาก ซึ่งอาจเป็นความซ้ำซ้อน หรืออาจพบว่าไม่มีรายวิชาใดครอบคลุมผลการเรียนรู้ที่สำคัญบางประเด็น ซึ่งเป็นช่องว่างของหลักสูตร

8. การจัดกลุ่มเนื้อหา

การจัดกลุ่มเนื้อหาที่สอนเพื่อค้นหาความเชื่อมโยงระหว่างหัวข้อ แนวคิด หรือทักษะที่เกี่ยวข้องกัน สามารถช่วยให้ผู้สอนเห็นโครงสร้างของความรู้ทั้งในระดับบทเรียนและรายหลักสูตรได้ชัดเจนยิ่งขึ้น เช่น การนำคำอธิบายรายวิชาหรือแผนการสอนมาวิเคราะห์ด้วยการจัดกลุ่มข้อความ เพื่อระบุว่าหัวข้อใดมีความซ้ำซ้อน เชื่อมโยง นอกจากนี้ยังสามารถใช้เพื่อค้นหาความสอดคล้องของเนื้อหาในแต่ละระดับชั้นหรือวิชา เช่น จัดกลุ่มเนื้อหาในวิชาวิทยาศาสตร์และสังคมศึกษามีหัวข้อที่เชื่อมโยงกัน ซึ่งช่วยในการออกแบบกิจกรรมบูรณาการ

9. การจัดกลุ่มงานวิจัยทางการศึกษา

นักวิจัยสามารถจัดกลุ่มงานวิจัยโดยอาจใช้ชื่อเรื่องหรือบทคัดย่อจำแนกกลุ่มของงานวิจัยในประเด็นที่สนใจ เพื่อให้เห็นภาพรวมแนวโน้มของงานวิจัย รวมทั้งเพื่อให้เห็นช่องว่างของงานวิจัย

10. การกำหนดนโยบายทางการศึกษา

ผู้กำหนดนโยบายสามารถรวบรวมและวิเคราะห์ข้อความจากแบบสำรวจออนไลน์ การรับฟังความคิดเห็นประชาชน และโซเชียลมีเดีย เพื่อจัดกลุ่มธีมหลัก เช่น งบประมาณการศึกษา การพัฒนาผู้สอน หรือการใช้เทคโนโลยี จากนั้นจัดลำดับความสำคัญของนโยบาย พร้อมทั้งสังเคราะห์ความคิดเห็นจากผู้มีส่วนได้ส่วนเสีย (stakeholders) โดยจัดกลุ่มประเด็นร่วมและข้อเสนอเฉพาะกลุ่ม เพื่อออกแบบนโยบายที่ครอบคลุมทุกฝ่าย

ตัวอย่างการจัดกลุ่มข้อความโดยใช้ k-means

บทความนี้นำเสนอตัวอย่างการจัดกลุ่มโดยใช้ k-means เนื่องจากเป็นอัลกอริทึมพื้นฐานที่มีความยืดหยุ่น (flexibility) เนื่องจากสามารถประยุกต์ใช้ได้กับข้อมูลหลายรูปแบบ เป็นอัลกอริทึมที่มีประสิทธิภาพ (efficiency) เนื่องจากมีความเรียบง่าย (simplicity) และมีความซับซ้อนทางคำนวณต่ำ (low computational complexity) โดยใช้การคำนวณระยะทางระหว่างจุดข้อมูลกับเซนทรอยด์ และปรับตำแหน่งเซนทรอยด์ซ้ำ ๆ จึงใช้เวลาประมวลผลต่ำเมื่อเทียบกับเทคนิคอื่น นอกจากนี้ยังง่ายต่อการนำไปใช้ (ease of implementation) เนื่องจากมีไลบรารีที่สามารถนำไปใช้งานได้ (Ikotun et al., 2023)

k-means เป็นวิธีการจัดกลุ่มแบบแบ่งส่วน (partitioning algorithms) หรือการแบ่งกลุ่มตามจุดศูนย์กลาง (centroid based-clustering) ซึ่งแบ่งข้อมูลออกกลุ่ม n ตัวอย่างออกเป็น k กลุ่ม โดยสุ่มจุดศูนย์กลางเริ่มต้น (centroids) k จุด แล้วทำงานสลับระหว่างการจัดกลุ่มและการอัปเดตตำแหน่งเซนทรอยด์ใหม่ จนกว่าเซนทรอยด์จะคงที่ เพื่อให้ within-cluster variance ต่ำสุด

วิธีการเลือกจำนวนกลุ่มที่นิยมใช้ เช่น elbow method ซึ่งวัดประสิทธิภาพด้วย within-cluster sum-of-squares (WCSS) โดยคำนวณผลรวมของกำลังสองของระยะระหว่างจุดข้อมูลแต่ละจุดกับเซนทรอยด์ของกลุ่มที่ข้อมูลนั้นอยู่ เมื่อค่า k เพิ่มขึ้น ค่า WCSS จะลดลง แต่ในช่วงหนึ่งของการลดลง ค่า WCSS จะเริ่มชะลอตัวจนเกิดเป็นข้อศอก (elbow) บนกราฟ ซึ่งค่า k ที่ตำแหน่งข้อศอก จึงมักถือเป็นจำนวนกลุ่มที่เหมาะสมที่สุด (Cui, 2020) นอกจากนี้ยังสามารถพิจารณาค่าอื่น ๆ ประกอบ เช่น silhouette coefficient อีกทั้งพิจารณาเลือกจำนวนกลุ่มจากความหมายและการตั้งชื่อกลุ่มร่วมด้วย

ขั้นตอนการทำงานของ k-means

1. กำหนดจำนวนกลุ่ม (k) และสุ่มเลือกจุดศูนย์กลางเริ่มต้น (centroids) จากชุดข้อมูลต้นฉบับ

2. สำหรับแต่ละจุดข้อมูลในชุดข้อมูล คำนวณระยะทางไปยังเซนทรอยด์แต่ละจุด แล้วจัดจุดข้อมูลนั้นเข้าไปในกลุ่มของเซนทรอยด์ที่ใกล้ที่สุด

3. เมื่อทุกจุดข้อมูลถูกจัดกลุ่มแล้ว คำนวณเซนทรอยด์ใหม่ของแต่ละกลุ่ม โดยใช้ค่าเฉลี่ยของจุดในกลุ่มนั้น

4. ทำซ้ำขั้นตอนที่ 2 และ 3 จนกว่าตำแหน่งเซนทรอยด์จะไม่เปลี่ยนแปลงที่หรือค่าการเปลี่ยนแปลงต่ำกว่าเกณฑ์ที่กำหนด (Aggarwal & Zhai, 2012; Kodinariya & Makwana, 2013)

การตั้งชื่อกลุ่มจะกำหนดภายหลังจากการวิเคราะห์และใช้วิจารณญาณของผู้วิเคราะห์ในการตั้งชื่อกลุ่ม โดยสามารถกำหนดจากคำสำคัญที่มีค่าน้ำหนักสูงสุดในเซนทรอยด์ของแต่ละกลุ่มเพื่อสรุปธีมร่วม และอาจศึกษาตัวอย่างเอกสารในกลุ่มเพื่อยืนยันว่าความสอดคล้องของคำสำคัญกับเนื้อหา รวมทั้งอาจใช้ความรู้เฉพาะด้านเกี่ยวกับเรื่องที่วิเคราะห์มาตั้งชื่อให้สอดคล้องกับบริบท

ตัวอย่างโค้ดที่จะใช้เป็นตัวอย่างการจัดกลุ่มข้อความด้วย k-means

#ติดตั้งและนำเข้าไลบรารี

```
!pip install pythainlp scikit-learn matplotlib
import re
import random
import pandas as pd
from itertools import chain
from collections import Counter
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.decomposition import TruncatedSVD
import matplotlib.pyplot as plt
```

#กำหนดคลาสสำหรับการจัดกลุ่มข้อความภาษาไทย

```
class ThaiTextClustering:
    def __init__(
        self,
        ngram_range=(1,1),
        max_df=0.25,
        min_df=10,
        sublinear_tf=True,
        stopwords_extra=None,
        random_state=42 ):
```

ตั้งค่าพารามิเตอร์ TF–IDF, stopwords และตัวแปรเริ่มต้น

```
        self.vec_params = dict(
            tokenizer=str.split,
            token_pattern=None,
            ngram_range=ngram_range,
            max_df=max_df,
            min_df=min_df,
            sublinear_tf=sublinear_tf)
        self.random_state = random_state
        self.stop = set(thai_stopwords())
        if stopwords_extra:
            self.stop.update(stopwords_extra)
        self.vectorizer = None
        self.model = None
        self.X = None
        self.labels = None
```

กำหนดฟังก์ชันทำความสะอาดข้อความและตัดคำ

```
    def clean_and_tokenize(self, text: str) -> List[str]:
        txt = normalize(text)
        txt = re.sub(r"http\S+|@\w+|#\w+|\d+|[^\u0E00-\u0E7F\s]", " ", txt)
        toks = word_tokenize(txt, engine="newmm")
        return [t for t in toks if len(t) > 1 and t not in self.stop]
```

กำหนดฟังก์ชันสร้าง TF-IDF matrix

```
    def fit_transform(self, docs: List[str]):
        tokenized = [self.clean_and_tokenize(d) for d in docs]
        docs_str = [" ".join(t) for t in tokenized]
        self.vectorizer = TfidfVectorizer(**self.vec_params)
        self.X = self.vectorizer.fit_transform(docs_str)
        return self.X, tokenized
```

กำหนดฟังก์ชันประเมินหาช่วงค่า k ที่เหมาะสม

```
    def find_best_k(self, ks=range(2, 11)):
        inertias, sils = [], []
        for k in ks:
          km = KMeans(n_clusters=k, random_state=self.random_state).fit(self.X)
            inertias.append(km.inertia_)
            sils.append(silhouette_score(self.X, km.labels_))
        return list(ks), inertias, sils
```

กำหนดฟังก์ชันสร้างโมเดล k-means

```
    def fit_kmeans(self, k: int):
        self.model = KMeans(n_clusters=k, random_state=self.random_state).fit(self.X)
```

```
56          self.labels = self.model.labels_
57          return self.labels
    # กำหนดฟังก์ชันสรุปจำนวนเอกสารและเปอร์เซ็นต์ในแต่ละกลุ่ม
58      def summarize_clusters(self, docs: List[str]) -> pd.DataFrame:
59          df = pd.DataFrame({'text': docs, 'cluster': self.labels})
60          counts = df['cluster'].value_counts().sort_index()
61         percents = df['cluster'].value_counts(normalize=True).sort_index() * 100
62         return pd.DataFrame({'count': counts, 'percent': percents.round(2)})
    # กำหนดฟังก์ชันดึงคำสำคัญของแต่ละกลุ่ม
63      def top_terms_per_cluster(self, top_n=20) -> Dict[int, List[str]]:
64          terms = self.vectorizer.get_feature_names_out()
65          centers = self.model.cluster_centers_
66          result = {}
67          for i, center in enumerate(centers):
68              idx = center.argsort()[::-1][:top_n]
69              result[i] = [terms[j] for j in idx]
70          return result
    # กำหนดฟังก์ชันแสดงตัวอย่างเอกสารจากแต่ละกลุ่ม
71      def sample_docs(self, docs: List[str], n_samples: int = 5):
72          df = pd.DataFrame({'text': docs, 'cluster': self.labels})
73          for c in sorted(df['cluster'].unique()):
74              print(f"Cluster {c}")
75              docs_c = df[df['cluster'] == c]['text'].tolist()
76             for doc in random.sample(docs_c, min(n_samples, len(docs_c))):
77                  print(doc)
                print()
    # กำหนดฟังก์ชันวาดกราฟ elbow และ silhouette
78      def plot_elbow_silhouette(self, ks, inertias, sils):
79          plt.figure(figsize=(10, 4))
80          plt.subplot(1, 2, 1)
81          plt.plot(ks, inertias, marker='o')
82          plt.title("Elbow Method")
83          plt.xlabel("k"); plt.ylabel("Inertia")
84          plt.subplot(1, 2, 2)
85          plt.plot(ks, sils, marker='o')
86          plt.title("Silhouette Score")
87          plt.xlabel("k"); plt.ylabel("Score")
88          plt.tight_layout()
89          plt.show()
    # กำหนดฟังก์ชันสร้างและแสดง word cloud ของคำสำคัญแต่ละกลุ่ม
90      def plot_wordclouds(self, tokenized_docs: List[List[str]], font_path: str, top_n=12):
91          df_idx = pd.DataFrame({'tokens': tokenized_docs, 'cluster': self.labels})
92          for c in sorted(df_idx['cluster'].unique()):
93              tokens = list(chain.from_iterable(df_idx[df_idx['cluster'] == c]['tokens']))
94              freq = dict(Counter(tokens).most_common(top_n))
95               wc = WordCloud(font_path=font_path, width=800, height=400,
    background_color='white').generate_from_frequencies(freq)
96              plt.figure(figsize=(10, 5))
97              plt.imshow(wc, interpolation='bilinear')
98              plt.axis('off')
99              plt.title(f'Cluster {c} Top {top_n} Words')
100             plt.show()
    # การนำไปใช้งาน
    # 1. อัปโหลดและอ่านไฟล์
101 uploaded = files.upload()
102 df = pd.read_csv('data.csv')
103 docs = df['news'].fillna('').astype(str).tolist()
    # 2. สร้าง pipeline สำหรับจัดกลุ่มข้อความ
104 pipeline = ThaiTextClustering(stopwords_extra={"ไทย","บาท","เพจ","ปี"})
    # 3. แปลงข้อความเป็น TF–IDF
105 X, tokenized = pipeline.fit_transform(docs)
```

```
    # 4. หา k ที่เหมาะสม
106 ks, inertias, sils = pipeline.find_best_k(range(2,11))
107 pipeline.plot_elbow_silhouette(ks, inertias, sils)
    # 5. ฝึกโมเดล k-means ด้วย k ที่เลือก
108 pipeline.fit_kmeans(5)
    # 6. สรุปจำนวนเอกสารและเปอร์เซ็นต์ในแต่ละกลุ่ม
109 print(pipeline.summarize_clusters(docs))
    # 7. แสดงคำสำคัญของแต่กลุ่ม
110 for c, terms in pipeline.top_terms_per_cluster(10).items():
111     print(f"Cluster {c}:", ", ".join(terms))
    # 8. แสดงตัวอย่างเอกสารสุ่มในแต่ละกลุ่ม
112 pipeline.sample_docs(docs, n_samples=5)
    # 9. Visualization
113 uploaded = files.upload()
114 pipeline.plot_2d_scatter()
115 pipeline.plot_wordclouds(tokenized, font_path='/content/ChulaCharasNewReg.ttf')
```

คำอธิบายโค้ด

Lines 1-11 ติดตั้งและนำเข้าไลบรารีที่ใช้สำหรับการวิเคราะห์ข้อความภาษาไทย

Lines 12-20 กำหนดคลาสสำหรับการจัดกลุ่มข้อความภาษาไทย

Line 15 ngram_range=(1,1) กำหนดช่วงของ n-gram ที่จะใช้ใน TF–IDF

Line 16 max_df=0.25 ตัดคำที่ปรากฏในเอกสารมากเกิน 25% ของทั้งหมด เพื่อกรองคำทั่วไป

Line 17 min_df=10 ตัดคำที่ปรากฏในเอกสารน้อยกว่า 10 ชิ้น เพื่อกรองคำที่หายากเกินไป

Line 18 sublinear_tf=True เปิดใช้งานการปรับ TF แบบ log-scaling (1 + log(tf)) แทนค่า TF แบบเดิม

Line 19 stopwords_extra=None รับชุด stopwords เพิ่มเติม สำหรับกรณีมีคำศัพท์ที่เฉพาะที่กำหนดขึ้น

Line 20 random_state=42 กำหนด seed สำหรับการสุ่มซ้ำ

Lines 21-35 การตั้งค่าพารามิเตอร์ TF–IDF, stopwords และตัวแปรเริ่มต้น

Lines 21-27 ตั้งค่าการสร้างเวกเตอร์ TF–IDF โดยกำหนดพารามิเตอร์ต่างๆ

Lines 29-31 โหลด stopwords จาก PyThaiNLP พร้อมเพิ่มคำเฉพาะ เพื่อกรองคำที่ไม่ต้องการออก

Lines 32-35 ประกาศ attribute เริ่มต้น (vectorizer, model, X และ labels) ให้เป็น none เพื่อวางโครงสร้างคลาสให้ชัดเจน และหลีกเลี่ยงการเรียกใช้ตัวแปรก่อนพร้อมใช้งาน

Lines 36-40 ฟังก์ชันทำความสะอาดข้อความและตัดคำ

Line 37 ปรับ normalization (ปรับสระและตัวอักษรใน PyThaiNLP ให้อยู่ในรูปมาตรฐาน)

Line 38 ลบ URL, mentions, hashtags, ตัวเลข และอักขระที่ไม่ใช่ภาษาไทยด้วย regex

Line 39 ตัดคำด้วย word_tokenize โดยกำหนด engine="newmm"

Line 40 กรอง token ที่มีความยาว 1 (ตัวอักษรเดี่ยว) หรือตรงกับ stopwords

Lines 41-46 กำหนดฟังก์ชันสร้าง TF-IDF matrix

Lines 41-43 แปลงเอกสารเป็นโทเคน (tokenize) แล้วรวมโทเคนแต่ละรายการเป็นสตริงเดียว (join) เพื่อใช้เป็นอินพุตให้ TF–IDเพื่อป้อนให้ TF–IDF

Lines 44–45 สร้าง TF–IDF matrix ด้วย TfidfVectorizer และเก็บผลลัพธ์ไว้ใน self.X

Lines 47-53 กำหนดฟังก์ชันประเมินหาช่วงค่า k ที่เหมาะสม

Line 48 เตรียมลิสต์ว่างสำหรับเก็บผล

Lines 49–52 วนลูปสร้างโมเดล คำนวณ inertia และ silhouette_score

Lines 54-57 กำหนดฟังก์ชันสร้างโมเดล k-means

Lines 58-62 กำหนดฟังก์ชันสรุปจำนวนเอกสารและเปอร์เซ็นต์ในแต่ละกลุ่ม

Line 59 สร้าง DataFrame ชั่วคราว df จับคู่คอลัมน์ 'text' (เอกสาร) กับ 'cluster' (labels)

Line 60 คำนวณจำนวนเอกสารต่อกลุ่ม

Line 61 คำนวณเปอร์เซ็นต์ของเอกสารต่อกลุ่ม

Lines 63-70 กำหนดฟังก์ชันดึงคำสำคัญของแต่ละกลุ่ม

Lines 64-65 ดึงชื่อฟีเจอร์จาก vectorizer และค่าเซนทรอยด์จาก model

Lines 66-69 หา index ของ top-n terms สำหรับแต่ละกลุ่ม แล้วเชื่อมโยงกลับเป็นคำ

Lines 71-77 กำหนดฟังก์ชันแสดงตัวอย่างเอกสารจากแต่ละกลุ่ม

Lines 71–73 เตรียม DataFrame และวนลูปแต่ละกลุ่ม

Lines 74–77 พิมพ์หัวข้อและสุ่มตัวอย่างเอกสารแต่ละกลุ่ม

Lines 78-89 กำหนดฟังก์ชันวาดกราฟ elbow และ silhouette

Lines 79-83 วาดกราฟ elbow method

Lines 84-87 วาดกราฟ silhouette score

Lines 88-89 จัด layout และแสดงผล

Lines 90-100 กำหนดฟังก์ชันสร้างและแสดง word cloud ของคำสำคัญแต่ละกลุ่ม

Lines 91-93 ทำ DataFrame รวม tokens กับ cluster

Lines 94-95 คำนวณความถี่และสร้าง word cloud

Lines 96-100 วาดและปรับแต่ง word cloud ทีละกลุ่ม

Lines 101-103 อัปโหลดและอ่านไฟล์

Line 101 เรียก files.upload() ให้เลือกอัปโหลดไฟล์จากเครื่อง

Line 102 อ่านไฟล์ชื่อ data.csv ด้วย pd.read_csv() เก็บลงเป็น DataFrame ชื่อ df

Line 103 ดึงคอลัมน์ 'news' จาก df แทนค่าว่างด้วย '' แล้วแปลงเป็น Python list ชื่อ docs

Lines 104 สร้าง pipeline โดยสร้างอ็อบเจกต์ ThaiTextClustering พร้อมกำหนด stopwords เพิ่มเติม ได้แก่ ({"ไทย","บาท","เพจ","ปี"})

Line 105 แปลงข้อความเป็น TF–IDF

Lines 106-107 หา k ที่เหมาะสม

Line 106 เรียก find_best_k(range(2,11)) เพื่อประเมินค่า inertia และ silhouette score ของแต่ละ k

Line 107 วาดกราฟ elbow method และ silhouette score

Line 108 ฝึกโมเดล k-means ด้วย k ที่เลือก

Line 109 สรุปจำนวนเอกสารและเปอร์เซ็นต์ในแต่ละกลุ่ม
Lines 110-111 แสดงคำสำคัญของแต่กลุ่ม
Line 112 แสดงตัวอย่างเอกสารสุ่มในแต่ละกลุ่ม
Lines 113-115 Visualization

Line 113 เรียก files.upload() เพื่ออัปโหลดฟอนต์ที่จะใช้

Line 114 สร้าง scatter plot 2D ของข้อมูลที่ลดมิติมา

Line 115 สร้างและแสดง word cloud ของแต่ละกลุ่ม โดยกำหนดที่อยู่ของฟอนต์คือ '/content/ChulaCharasNewReg.ttf'

ตัวอย่างการรายงานผลการวิเคราะห์

การเตรียมข้อมูล (preprocessing)

ข้อมูลที่ใช้วิเคราะห์ในครั้งนี้เป็นหัวข้อข่าวปลอมภาษาไทยจำนวน 1,728 เรื่อง จากเว็บไซต์ศูนย์ต่อต้านข่าวปลอมประเทศไทย โดยเก็บข้อมูลตั้งแต่วันที่ 1 มีนาคม 2567 ถึง 31 มีนาคม 2568 ซึ่งมีการเตรียมข้อมูล ดังนี้

1) ทำความสะอาดข้อความ ด้วยการ normalize และลบ URL, mentions, hashtags, ตัวเลขหรือตัวอักษรที่ไม่ใช่ภาษาไทย

2) ตัดคำ (tokenization) โดยใช้ PyThaiNLP (newmm) แล้วกรอง stopwords และคำสั้น (1 ตัวอักษร)

3) สกัดคุณลักษณะ ด้วย TF–IDF (max_df=0.25, min_df=10, ngram_range=(1,1), sublinear_tf=True) ซึ่งผลลัพธ์เป็นเมทริกซ์ TF–IDF ขนาด 1,728×496

การเลือกจำนวนกลุ่ม

เมื่อพิจารณากราฟ elbow และ silhouette score ร่วมกัน และคำนึงถึงการตีความหัวข้อในแต่ละกลุ่ม พบว่า k = 5 เป็นจำนวนกลุ่มที่เหมาะสมที่สุด เนื่องจากค่า inertia จาก elbow method ลดลงอย่างชัดเจนเมื่อเปลี่ยนจาก k = 4 เป็น k = 5 และ silhouette score มีค่าสูงขึ้นในช่วงดังกล่าว โดยเมื่อกำหนด k = 5 silhouette score จะมีค่าเท่ากับ 0.0436 นอกจากนี้การแบ่งออกเป็น 5 กลุ่มยังสามารถตีความเนื้อหาในแต่ละกลุ่มได้อย่างเหมาะสม ดังนั้น การวิเคราะห์ครั้งนี้จึงเลือกใช้ k = 5 เป็นจำนวนกลุ่มหลัก เพื่อให้ได้ทั้งความชัดเจนเชิงสถิติและความสมเหตุสมผลเชิงเนื้อหา

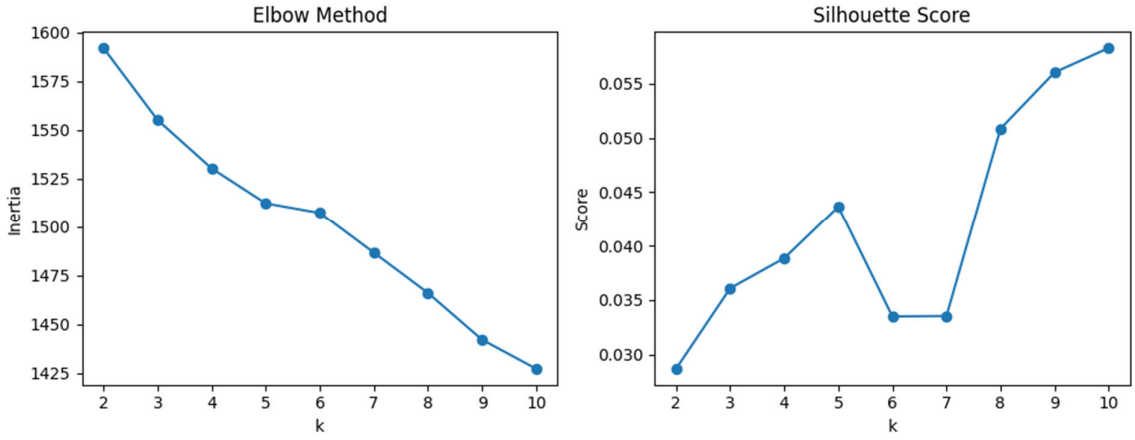
การวิเคราะห์นี้ได้ค่า silhouette score ที่ต่ำ (0.0436) บ่งชี้ว่าการจัดกลุ่มยังไม่ชัดเจนและระยะห่างระหว่างกลุ่มมีค่าน้อย ทำให้การสรุปธีมหรือลักษณะร่วมของแต่ละกลุ่มอาจคลาดเคลื่อน เนื่องจากเวกเตอร์ TF–IDF เป็นข้อมูลที่มีมิติสูง (high-dimensional) และมีลักษณะความหนาแน่นต่ำ (sparse) การวัดระยะทางด้วย Euclidean จึงวัดได้เพียงความใกล้เคียงเชิงตัวเลขเท่านั้น แต่ไม่สะท้อนความใกล้เคียงทางความหมายหรือบริบทของเอกสารจริง

นอกจากนี้การจัดกลุ่มโดยใช้ k-means มีข้อจำกัด เช่น 1) ต้องกำหนดจำนวนกลุ่ม k ล่วงหน้า ซึ่งอาศัยหลักเกณฑ์เชิงคณิตศาสตร์แบบ heuristic หรือการทดลองวิเคราะห์หลายค่าแล้วเลือกค่าที่เหมาะสมที่สุด 2) ผลลัพธ์ขึ้นกับการเริ่มต้น (initialization) และอาจติดที่ local

minimum 3) สมมุติรูปทรงกลุ่มเป็นทรงกลม (spherical clusters) ไม่เหมาะกับโครงสร้างข้อมูลที่เป็นกลุ่มรูปร่างอิสระหรือซับซ้อน และ 4) ไวต่อข้อมูลรบกวน (noise) และข้อมูลผิดปกติ (outliers) โดยจะดึงเซนทรอยด์ให้เคลื่อนห่างจากกลุ่มหลัก (Jain, 2010) ดังนั้น ในการวิเคราะห์จัดกลุ่มจึงควรวิเคราะห์ลักษณะเบื้องต้นของข้อมูล และอาจเปรียบเทียบผลลัพธ์กับวิธีจัดกลุ่มด้วยอัลกอริทึมอื่น ๆ รวมทั้งใช้ตัวชี้วัดประสิทธิภาพการจัดกลุ่มด้วยตัวชี้วัดอื่น ๆ ร่วมด้วย

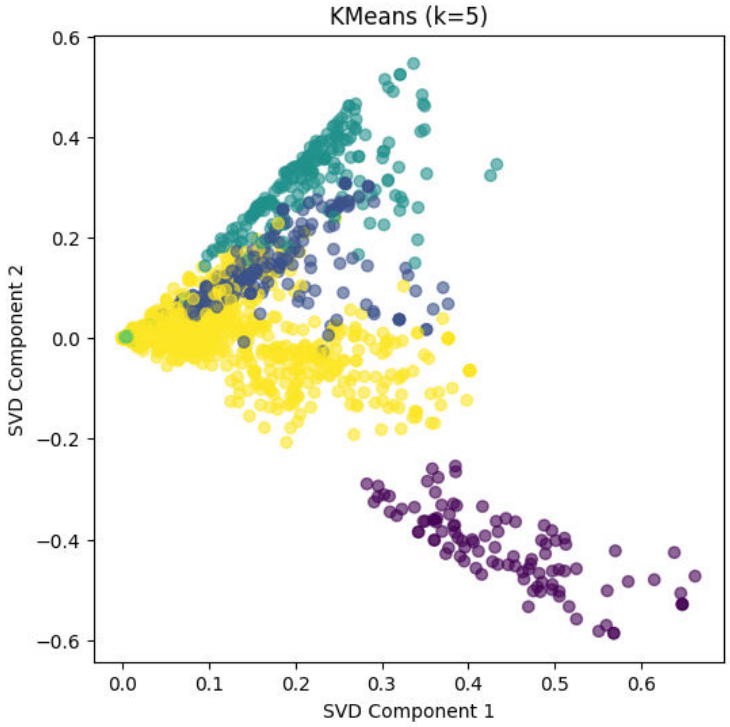
ภาพ 1

กราฟเปรียบเทียบ inertia (elbow method) และ silhouette score (k ตั้งแต่ 2 ถึง 10)



ภาพ 2

การจัดกลุ่มข้อความด้วย k-means (k = 5) บนพื้นที่สองมิติด้วย Truncated SVD



ลักษณะของแต่ละกลุ่ม

จากการวิเคราะห์จัดกลุ่มโดยกำหนดจำนวนกลุ่ม 5 กลุ่ม จะสามารถจัดได้ 5 กลุ่ม ดังนี้ **1) การสมัครงาน** เป็นกระบวนการรับสมัครงานผ่านช่องทางออนไลน์ ซึ่งมีสัดส่วนสูงสุด เท่ากับ 66.90% **2) การลงทุน** รวบรวมหัวข้อเกี่ยวกับตลาดหลักทรัพย์ หุ้น และบริการชักชวนลงทุน ซึ่งมีสัดส่วน 13.02% **3) การเงิน** ครอบคลุมการปล่อยสินเชื่อ ธนาคาร และดอกเบี้ยซึ่งมีสัดส่วน 11.40% **4) การขนส่ง** สะท้อนข่าวสารเรื่องต่ออายุใบขับขี่และบริการกรมการขนส่ง ซึ่งมีสัดส่วน 5.84% และ**5) สุขภาพ** เกี่ยวกับความเสี่ยง โรคหัวใจ และอาการต่างๆ มีสัดส่วนน้อยที่สุดคือ 2.84%

ตาราง 1

คำสำคัญและจำนวนเอกสารของแต่ละกลุ่ม

กลุ่ม	ตัวอย่างคำสำคัญ	จำนวนเอกสาร	ร้อยละ
การขนส่ง	ใบขับขี่, ทำ, ออนไลน์, กรมการขนส่งทางบก, ขนส่ง, ต่ออายุ, ไม่ต้อง, สอบ, ทุกชนิด, ถูกกฎหมาย, กรม, เฟซบุ๊ก, บัตร, ประเภท, ลงทะเบียน	101	5.84
การลงทุน	ลงทุน, ตลาดหลักทรัพย์, หุ้น, ไลน์, บัญชี, ชักชวน, สำนักงาน, เริ่มต้น, การลงทุน, ประเทศ, ปันผล, เฟซบุ๊ก, กองทุน, พอร์ต, เว็บไซต์	225	13.02
การเงิน	สินเชื่อ, ปล่อย, ออมสิน, กรุงไทย, กู้, ธนาคารออมสิน, วงเงิน, ผ่อน, บัญชี, เงินกู้, ธนาคาร, เดือน, ดอกเบี้ย, ลงทะเบียน, ส่วนบุคคล	197	11.40
สุขภาพ	เสี่ยง, สมอง, หลอดเลือด, เลือด, เส้นเลือด, เป็นโรค, หัวใจ, ดื่ม, ร่างกาย, น้ำ, โรค, สัญญาณ, อาการ, อักเสบ, คน	49	2.84
การสมัครงาน	เฟซบุ๊ก, รับสมัคร, ออนไลน์, แจ้ง, ลงทะเบียน, คืน, ชื่อ, ติดต่อ, ผลิตภัณฑ์, เงิน, รับเงิน, บิน, กรมการจัดหางาน, สมัครงาน, รายได้	1,156	66.90

ภาพ 3

คำสำคัญของกลุ่มการขนส่ง

ภาพ 4

คำสำคัญของกลุ่มการลงทุน

ภาพ 5

คำสำคัญของกลุ่มการเงิน

ภาพ 6

คำสำคัญของกลุ่มการลงทุน

ภาพ 7

คำสำคัญของกลุ่มการขนส่ง

หมายเหตุเพิ่มเติมจากตัวอย่าง

ค่า silhouette score ที่ได้จากการวิเคราะห์ในตัวอย่างข้างต้นนี้อยู่ในระดับต่ำ (0.0436) แสดงถึงความไม่ชัดเจนในการแบ่งกลุ่ม โดยเฉพาะในกรณีข้อมูลข้อความที่มีมิติสูง (high-dimensional) และความหนาแน่นต่ำแบบ sparse อย่างเวกเตอร์ TF–IDF ซึ่งไม่สามารถสะท้อนความหมายเชิงบริบทของข้อความได้ดีนัก ข้อจำกัดนี้ชี้ให้เห็นว่าการจัดกลุ่มโดยใช้ k-means กับข้อมูลข้อความอาจต้องพิจารณาร่วมกับเทคนิคอื่น เช่น การแทนข้อความแบบ contextual embeddings (เช่น BERT หรือ fastText) การลดมิติข้อมูลด้วย PCA หรือ UMAP การใช้อัลกอริทึม clustering ที่ไม่อิงโครงสร้างกลุ่มกลม เช่น DBSCAN หรือ HDBSCAN ค่า silhouette score ควรใช้ร่วมกับการตรวจสอบกลุ่มในเชิงเนื้อหา เพื่อประเมินว่ากลุ่มที่ได้มีความหมายและใช้ประโยชน์ได้จริงหรือไม่

สรุปและข้อเสนอแนะเพิ่มเติม

บทความนี้แสดงให้เห็นถึงขั้นตอนการจัดกลุ่มข้อความภาษาไทย (Thai text clustering) แบบไม่ซับซ้อน โดยใช้ TF–IDF และ k-means พร้อมโค้ด Python และตัวอย่างการวิเคราะห์จริงในบริบทข่าวปลอมภาษาไทย จุดเด่นคือการแสดงลำดับการทำงานแบบครบถ้วน ตั้งแต่การเตรียมข้อมูลไปจนถึงการตีความผลลัพธ์ แม้ว่าผลลัพธ์ของการจัดกลุ่มจะไม่ชัดเจนในเชิงสถิติ แต่การสรุปธีมจากกลุ่มยังสามารถนำไปใช้วิเคราะห์ข้อมูลเชิงเนื้อหาในเชิงสำรวจได้ โดยเฉพาะในงานที่เน้นการคัดแยกเบื้องต้นหรือจัดระเบียบข้อมูลปริมาณมาก

ข้อเสนอแนะเพิ่มเติมสำหรับผู้ที่สนใจพัฒนาแนวทางนี้ต่อ อาทิ การทดลองใช้ representation ที่ซับซ้อนขึ้น เช่น contextual embeddings จาก BERT หรือ fastText การเปรียบเทียบผลกับอัลกอริทึมอื่น เช่น hierarchical clustering หรือ HDBSCAN หรือการประยุกต์ใช้กับข้อมูลจากการเรียนการสอนจริง เช่น คำตอบผู้เรียน ความคิดเห็น หรือคำสะท้อนหลังเรียน ดังที่ได้นำเสนอตัวอย่างการนำไปประยุกต์ใช้ไว้ในบทความนี้ เทคนิคการจัดกลุ่มข้อความเป็นเครื่องมือที่ทันสมัย ยืดหยุ่น ช่วยให้นักวิจัยสามารถวิเคราะห์ข้อความในปริมาณมากได้อย่างมีประสิทธิภาพและสามารถต่อยอดได้ทั้งในเชิงวิจัยและการพัฒนานวัตกรรมทางการศึกษาได้อีกมาก

เอกสารอ้างอิง

Aggarwal, C. C., & Zhai, C. (2012). A survey of text clustering algorithms. *Mining text data*, 77-128. https://doi.org/10.1007/978-1-4614-3223-4_4

Ahmed, M. H., Tiun, S., Omar, N., & Sani, N. S. (2022). Short text clustering algorithms, application and challenges: a survey. *Applied Sciences*, *13*(1), 342. https://doi.org/10.3390/app13010342

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*. https://doi.org/10.48550/arXiv.1707.02919

Bhattacharjee, P., & Mitra, P. (2021). A survey of density based clustering algorithms. *Frontiers of Computer Science*, *15*, 1-27. https://doi.org/10.1007/s11704-019-9059-3

Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E., & Cui, P. (2020). Structural deep clustering network. In *Proceedings of the web conference 2020* (pp. 1400-1410). https://doi.org/10.1145/3366423.3380214

Campello, R. J., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 160-172). Springer. https://doi.org/10.1007/978-3-642-37456-2_14

Chormai, P., Prasertsom, P., & Rutherford, A. (2019). Attacut: A fast and accurate neural thai word segmenter. *arXiv*. https://doi.org/10.48550/arXiv.1911.07056

Cui, M. (2020). Introduction to the k-means clustering algorithm based on the elbow method. *Accounting, Auditing and Finance*, *1*(1), 5-8. https://doi.org/10.23977/accaf.2020.0101024

Ferreira-Mello, R., André, M., Pinheiro, A., Costa, E., & Romero, C. (2019). Text mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(6), e1332. https://doi.org/10.1002/widm.1332

Forestier, G., Wemmert, C., & Gançarski, P. (2010). Background knowledge integration in clustering using purity indexes. In *Knowledge Science, Engineering and Management: 4th International Conference, KSEM 2010, Belfast, Northern Ireland, UK, September 1-3, 2010. Proceedings 4* (pp. 28-38). Springer. https://doi.org/10.1007/978-3-642-15280-1_6

Hassan, B. A., Tayfor, N. B., Hassan, A. A., Ahmed, A. M., Rashid, T. A., & Abdalla, N. N. (2024). From A-to-Z review of clustering validation indices. *Neurocomputing*, *601*, 128198. https://doi.org/10.1016/j.neucom.2024.128198

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*, 193-218. https://doi.org/10.1007/BF01908075

Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2023). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, *622*, 178-210. https://doi.org/10.1016/j.ins.2022.11.139

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition letters*, *31*(8), 651-666. https://doi.org/10.1016/j.patrec.2009.09.011

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall.

Jiang, Z., Zheng, Y., Tan, H., Tang, B., & Zhou, H. (2016). Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv*. https://doi.org/10.48550/arXiv.1611.05148

Kleinberg, J. (2002). An impossibility theorem for clustering. *Advances in neural information processing systems*, *15*. https://proceedings.neurips.cc/paper/2002/hash/43e4e6a6f341e00671e123714de019a8-Abstract.html

Kodinariya, T. M., & Makwana, P. R. (2013). Review on determining number of Cluster in K-Means Clustering. *International Journal*, *1*(6), 90-95.

Kriegel, H. P., Kröger, P., Sander, J., & Zimek, A. (2011). Density-based clustering. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, *1*(3), 231-240. https://doi.org/10.1002/widm.30

Luque, C., Luna, J. M., Luque, M., & Ventura, S. (2019). An advanced review on text mining in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(3), e1302. https://doi.org/10.1002/widm.1302

Mehta, V., Agarwal, M., & Kaliyar, R. K. (2024). A comprehensive and analytical review of text clustering techniques. *International Journal of Data Science and Analytics*, *18*(3), 239-258. https://doi.org/10.1007/s41060-024-00540-x

Murtagh, F., & Contreras, P. (2012). Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *2*(1), 86-97. https://doi.org/10.1002/widm.53

Patil, R., Boit, S., Gudivada, V., & Nandigam, J. (2023). A survey of text representation and embedding techniques in NLP. *IEEE Access*, *11*, 36120-36146. https://doi.org/10.1109/ACCESS.2023.3266377

Petukhova, A., Matos-Carvalho, J. P., & Fachada, N. (2025). Text clustering with large language model embeddings. *International Journal of Cognitive Computing in Engineering*, *6*, 100-108. https://doi.org/10.1016/j.ijcce.2024.11.004

Phatthiyaphaibun, W., Chaovavanich, K., Polpanumas, C., Suriyawongkul, A., Lowphansirikul, L., Chormai, P., ... & Udomcharoenchaikit, C. (2023). PyThaiNLP: Thai natural language processing in Python. *arXiv*. https://doi.org/10.48550/arXiv.2312.04649

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513-523. https://doi.org/10.1016/0306-4573(88)90021-0

Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., ... & Lin, C. T. (2017). A review of clustering techniques and developments. *Neurocomputing, 267*, 664-681. https://doi.org/10.1016/j.neucom.2017.06.053

Shahapure, K. R., & Nicholas, C. (2020). Cluster quality analysis using silhouette score. In *2020 IEEE 7th international conference on data science and advanced analytics (DSAA)* (pp. 747-748). IEEE. https://doi.org/10.1109/DSAA49011.2020.00096.

Shutaywi, M., & Kachouie, N. N. (2021). Silhouette analysis for performance evaluation in machine learning with applications to clustering. *Entropy*, *23*(6), 759. https://doi.org/10.3390/e23060759

Sinaga, K. P., & Yang, M. S. (2020). Unsupervised k-means clustering algorithm. *IEEE Access, 8,* 80716–80727. https://doi.org/10.1109/ACCESS.2020.2988796

Timbers, T., Campbell, T., Lee, M., Ostblom, J., & Heagy, L. (2024). *Data Science: A First Introduction with Python*. CRC Press.

Xie, J., Girshick, R., & Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. In *International conference on machine learning* (pp. 478-487). PMLR. https://proceedings.mlr.press/v48/xieb16.html

Xu, D., & Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Annals of data science*, *2*(2), 165-193. https://doi.org/10.1007/s40745-015-0040-1

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on neural networks*, *16*(3), 645-678. https://doi.org/10.1109/TNN.2005.845141