Clustering and Thematic Analysis of TikTok Content For Insights into Social Media Trends

Suphakorn Homnan

School of Management

University of Bath

Supervisor: Professor David Ellis

Student Number: 239148629

Submitted as part of the requirement for completing an MSc in Business Analytics 2023-2024 at the University of Bath

Word count: 13,373

Abstract

Social media analysis is one of the most popular sections for understanding the primary theme and trending content to enhance the user experience and produce the best service to their users. Many researchers studied different online social media communities, such as Facebook, Twitter, and TikTok, using text mining and unsupervised and supervised machine learning to solve their questions for developing online businesses and increasing the efficiency of content in their area of social media. In this paper, we focus on the famous online video TikTok platform to understand the main theme of this social media and which categories of popular content are used by finding the proper tool that can categorize video type by considering video description, which we propose three clustering algorithms: K-means, Hierarchical, and Latent Dirichlet Allocation. We decided to select the LDA method as the suitable model that can provide the coherence score = 0.632 and can provide better relevant keywords for each topic than other models to produce the video category for presenting insights by doing thematic analysis and sentiment analysis to address our questions that show the entertainment, social media, and comedy are most of the contents and social media content revealed the most famous content in the TikTok dataset. These findings support entrepreneurs, content creators, and marketers in understanding their target audiences and improving marketing strategies to develop their decision-making and produce their products or content efficiently. We advise that future studies or improving this study should explore opinion analysis to understand the audience's reactions to the contents for enhancing user satisfaction to stay with the community for a long time.

Keywords: Social Media Analysis, Text Analysis, Clustering Model, Sentiment Analysis, Thematic Analysis, TikTok

Table of Contents

List of Figures & List of Tables

Chapter 1: Introduction

This chapter gives the background of online social media (TikTok) and short video brief history, the previous similar studies and then propose the academic motivation and show research structures.

1.1) Background

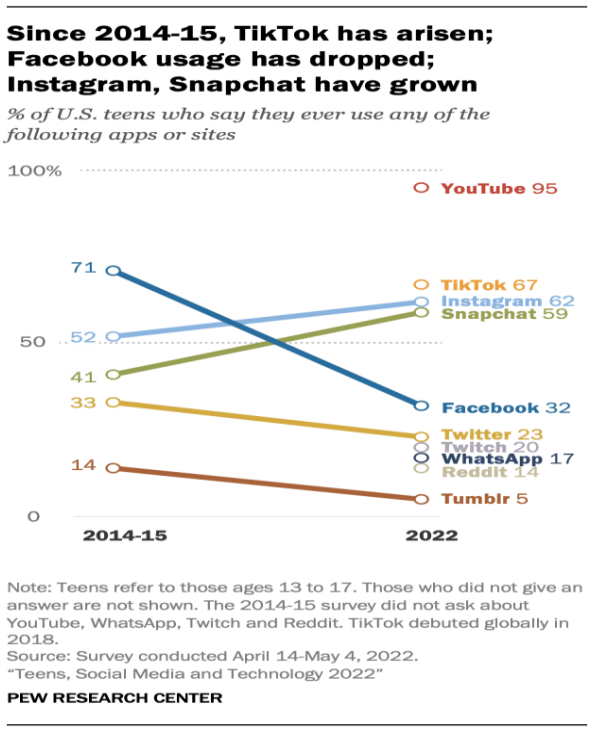
Social media analysis has become a popular research topic due to the ease of accessing public information through data scraping techniques. Many research papers have applied text mining to analyse unstructured text data and gain insights into various social media fields (Stieglitz et al., 2018). These studies often employ Natural Language Processing (NLP) techniques to convert unstructured data into structured data that computers can understand and process to deliver significant outcomes (Hirschberg and Manning, 2015).

Text mining approaches are widespread in academic research because they can handle complex data such as text articles, documents, and news (Aggarwal and Zhai, 2012). These data types are often complicated to analyse manually, but text mining techniques help researchers understand the insights by exploring patterns in large volumes of textual information (Feldman and Sanger, 2006). Furthermore, sentiment analysis, a subset of text mining, allows researchers to analyse the emotional context, categorizing content as positive or negative (Liu, 2012).

Online video social media platforms have gained significant popularity, enabling users to share diverse content and lifestyles. YouTube remains the dominant platform for online videos, hosting an extensive collection of content that facilitates interest exchange among users. However, recent trends indicate a shift in online video consumption habits. Short video platforms such as TikTok and Instagram Reels have rapidly gained interest, approaching YouTube in popularity. This shift has occurred swiftly, with data from just two to three years ago, as illustrated in Figure 1, showing a substantial increase in US teenagers using TikTok and Instagram. However, YouTube remains the leading platform (Vogels, Gelles-Watnick, and Massarat, 2022).

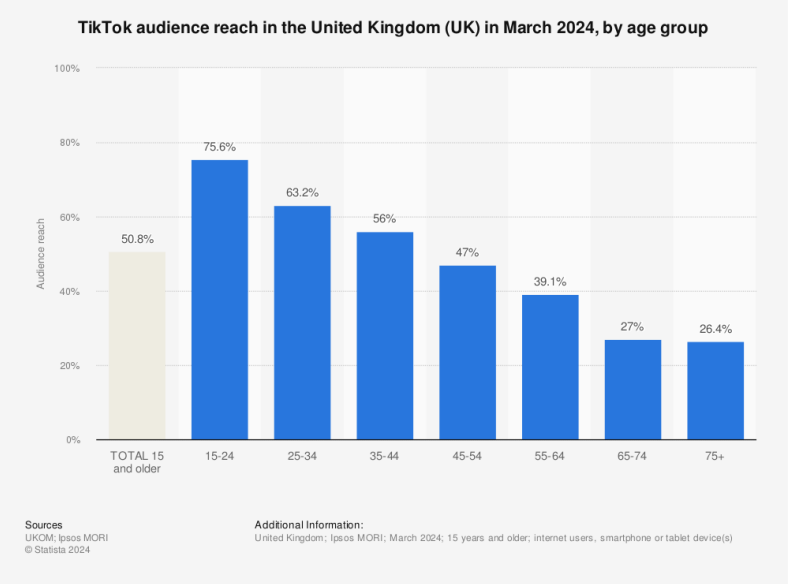
Analysing these data presents an opportunity to comprehend video trends and enhance content quality and online video communities. Applying text mining techniques allows for the classification of video content based on titles or descriptions, enabling cluster creation for further analysis. This approach facilitates understanding user engagement trends across various content types. Examining metrics such as view counts, likes, and shares can provide insights into content influence within online video communities. Employing classification models to categorize video types and visualizing the resultant data through charts can offer valuable insights into audience preferences and content impact over time.

This study will focus on TikTok data, which shows this platform was the hottest common application that expanded faster and attracted 1.5 billion active users of the hundreds of millions of users from teens and young adults (Natalie and Gabriel, 2020). Figure 2 shows that teenagers and young adults are the largest groups of TikTok audience in the UK, around 63-75% of the total survey population (Ceci, 2024). We can understand their social uses and the content they want to watch. In that case, it can help us improve the community, deliver critical insights, or apply in the commercial sector, such as doing marketing campaigns.



Figure

Figure 1: The percentage of U.S teenagers who say they have ever used any of the following apps or sites (Vogels, Gelles-Watnick and Massarat, 2022)



Figure

Figure 2: The percentage of TikTok audience reach in the United Kingdom in March 2024 by age group. (Ceci, 2024).

Text mining techniques can help categorize social media posts into relevant topics or themes to access content safety and identify potential safety issues and concerns about inappropriate content consumption (Isah, Trundle, and Neagu, 2014). We can appropriately use sentiment analysis tools to group those contents from social media platforms. Furthermore, the NLP approach, one of the text mining methods, helps analyse large textual datasets from many public social media platforms.

This process will allow the computer to understand the unstructured data by translating it into patterns as numeric data that it can execute continuously as-built models or other things. Additionally, it can break down complex text data into insights by making tokenization and building topic modelling to quickly identify the kind of information source to visualize the group of words.

Applying unsupervised machine learning concepts to receive insights is a practical idea and quite a complicated one at the same time. However, dealing with text information is complex because researchers must do many complex things before analysing the data. If we can handle those issues, we can reap many advantages.

From the prior study case, Siersdorfer and his team (2010) studied commenting behaviour on YouTube and the information value of comments. They analysed the correlation between comment ratings and different factors, such as sentiment expressed in comments, comment length, and video categories.

They show how their unsupervised clustering model of comments and videos can provide significant insights, sparking excitement about the potential of these methods to drive user engagement and content popularity. This demonstrates that this technique is handy when applied with text analysis to deliver the expected results.

1.2) Similar Works

Organizing unstructured information, such as text, into groups using machine learning has long been a popular area of research. This section will focus specifically on studies that have used text clustering as the core method to solve their research problems. We will explore how these researchers have applied this technique to answer their questions and address specific field challenges.

Ramamonjisoa (2014) proposed the comment analysis, applying topic modelling (LDA) to experiment, using two datasets that received 15 thousand comments: Yahoo and Tokyo Electric Power Company (TEPCO), to study the categories of each opinion of this information. This paper focuses on an experiment on user comments topic modelling, but it found some problems, such as spam or trolls within comments, and ignored them in the process. These things should be developed to find some approach to deal with it.

Prihatini and her team (2018) applied five hundred from five categories of Indonesian news media sites to study which machine learning model is more suitable for clustering Indonesian text. As a result, the LDA method outperformed the

TF-IDF of K-Means in clustering Indonesian text files. Evaluation metrics (Precision, Recall, F-measure) averaged 0.9369, 0.9114, and 0.9148 for LDA, compared to 11.5003, 0.4911, and 0.5304 for TF-IDF of K-Means. These results indicate that LDA is more effective at extracting relevant features from the text.

Phan, Ninh, and Ninh (2020) also produced the LDA application. Nonetheless, it applied a combination of topic distribution-based, building a system that can track trending content on Facebook without accessing the individual user's information to investigate those fan pages in Vietnam. This paper presented how to apply topic modelling tools to analyse text corpora to discover the hidden user's data patterns step-by-step, helping other researchers easily follow when they want to use or develop this idea further. Moreover, it revealed significant insights to develop those fan pages to understand their followers by clustering their users' posts into several to be easy to analyse continuously.

Mann and Kaur (2021) conducted a study using a dataset of over 24,000 research articles from the Journal of Cleaner Productions. They employed topic modelling and K-means clustering to categorize these articles into groups based on their content and visualized the main themes of each group using word clouds. To improve the accuracy and meaningfulness of these clusters, they experimented with contextual embeddings generated by BERT and Sentence-BERT. They could create more conceptually similar and relevant groups by representing the articles in a semantic space.

Dwivedi and Pathak (2022) conducted a study to examine people's feelings about COVID-19 vaccines based on 4,047 tweets from various places worldwide. They aimed to understand the overall sentiment towards the vaccines by categorizing tweets as positive or negative. Their findings indicated that most people on Twitter had a positive view of the COVID-19 vaccines. However, they also identified groups of people who expressed concerns about the vaccines, primarily due to their newness and lack of long-term data about their safety.

Recently, Bakar (2024) used a dataset of around 7,500 records to build text clustering models for grouping the topic of each trending TikTok video in Malaysia using K-means and topic modelling (LDA) methods. This research aims to locate and

extract the most popular TikTok content. From the LDA approach, they received a perplexity score of 287 and a log-likelihood score of 5,579. These values show how well their probability models predict the outcomes.

1.3) Academic Motivation

We aim to apply text mining techniques to build a tool that can group short video content by analysing video descriptions. We strive to explore trends in these videos by using clustering models to examine current user engagement and considering views and likes statistics to identify popular content. Applying text mining approaches can extract meaningful patterns from unstructured text data, providing valuable insights for content classification and trend analysis (Hartmann et al., 2019).

Applications for creating value include developing marketing plans and enhancing content quality for creators in advance. A primary goal is to explore the main theme in TikTok social media and differentiate between using positive and negative titles to describe their content. This distinction is achieved through sentiment analysis of word chunks, a crucial component of content moderation. By examining textual features, potentially harmful content can be identified (Liu, 2012), allowing for the evaluation and categorization of video descriptions.

A compact library determines word emotional scores and classifies content types. This report presents findings from applying various tools to visualize textual datasets, delivering significant results.

1.4) Research Structure

This study is organized as follows:

Chapter 1 Introduction: This chapter provides a background introduction to the paper, a review of previous research relevant to this study, and an outline of the main objectives and research structure. It offers readers a concise overview of the study's central ideas.

Chapter 2 Literature Review: This section comprehensively reviews concepts and theories, providing the theoretical foundation and basic knowledge applicable to this

research. Moreover, it gives both agreement and disagreement and our contribution of different ideas compared with previous research.

Chapter 3 Research Design and Methodology: This chapter details the materials and methodologies employed in this paper, including research design planning, data collection procedures, descriptions of tools used, and the evaluation approaches for measuring the findings.

Chapter 4 Findings: This section presents the study's outcomes and their analysis. It covers handling and cleaning raw data, exploring data analysis, building clustering models, interpreting clusters, and visualizing main themes. It also explains how sentiment analysis separates positive and negative videos using text descriptions.

Chapter 5 Discussion and Analysis: This chapter interprets the findings and analyses from the previous chapter, emphasizing their practical implications. It compares these findings with prior research, explains the challenges encountered, and discusses how our results can be applied in real-world scenarios. Finally, it acknowledges the limitations of this study and suggests future research directions.

Chapter 6 Conclusion: the final chapter summarizes the research process and its consequences. It also proposes recommendations for future improvements and research directions.

Chapter 2: Literature Review

This chapter will present an overview of the text mining process, including a brief background, preprocessing steps, feature extraction, and approaches to quickly converting textual data into numeric data to build a clustering model. Then, we will discuss the comparative analysis, including our agreement, disagreement, and different contribution ideas.

2.1) Relevant Theories and Ideas

2.1.1) Text Mining

Text mining is a significant technique for working with unstructured data, such as news and social media information. Feldman and Dagan (1995) first formally introduced the original concept. They defined "Knowledge Discovery in Textual

Databases" (KDT) as extracting sophisticated or unknown things and trying to handle them to get potentially useful information. One study applied one part of this technique, sentiment analysis, to work with Twitter's social media data. It can help deal with opinion analysis (Pak and Paroubek, 2010). Finding insights from the huge of textual information is fundamental.

This approach is the process of extracting valuable information from text data. This involves finding patterns and trends hidden within vast amounts of text using clever data analysis methods (Aggarwal and Zhai, 2012). By applying text mining techniques, there are many steps and many applied approaches, depending on how complete the data source is. Fundamentally, there are four main processes to work with textual data: data collection, preprocessing, feature extraction, and analysis of the insight information (Miner et al., 2012).

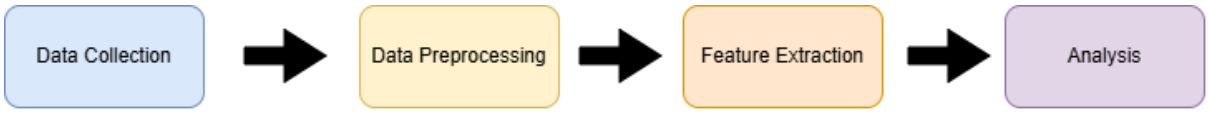


Figure 3: The fundamental processes of text mining

One interesting case study applied the text mining technique to solve real business challenges. In 2013, one study examined the influence of social media competitive analysis and how this approach can gain valuable business intelligence and priceless insights into the pizza industry (He, Zha, and Li, 2013). It shows that text mining can help researchers and analysts gain more insight from complex information, such as text, and how to take advantage of this method.

2.1.2) Text Preprocessing

2.1.2.a) Tokenization

Tokenization involves breaking down character streams into smaller chunks called tokens. Following tokenization, linguistic preprocessing groups these tokens into sets of equivalent terms, then indexed (Manning, Raghavan, and Hinrich Schütze, 2008). It involves a detailed analysis of each word in a sentence, identifying and filtering out noise or irrelevant information to enhance meaningful data extraction.

The primary goal of tokenization is to explore and analyse the words within a sentence, converting the textual data into machine-readable tokens (Verma, Duggal, and Gaur, 2014). Tokenization is a crucial step in the preprocessing phase for machine learning tasks. It includes cleaning the text by removing irrelevant or

unnecessary data, thereby ensuring that only significant information is retained for further analysis.

Using tokenization significantly improves the accuracy and efficiency of machine learning models. This method simplifies the text and enhances its relevance, making it a foundational step in natural language processing and machine learning.

2.1.2.b) Filtering

A common approach involves using pre-compiled lists of stop words. One of the earliest such lists was created by Fox (1989) based on word frequency analysis of the Brown Corpus, including 1,014,000 words drawn from a broad range of English literature. Removing these common words helps to highlight more informative keywords, which can improve the performance of machine learning models.

Text filtering, a fundamental process in data analysis, involves categorizing incoming documents. Belkin and Croft (1992) describe it as the process of separating relevant from irrelevant content. Various techniques, such as converting text to lowercase, removing punctuation, numbers, extra spaces, HTML tags, emojis, and special characters, are applied to achieve this. These steps are crucial in cleaning up the text and making it more amenable to analysis.

Manning, Raghavan, and Schütze (2008) observed a trend that included large lists of words (200-300 terms) to minimal lists of words (7-12 terms) or no-stop word lists at all. Stop word removal is a filtering process designed to identify and eliminate irrelevant words, allowing for a more precise focus on significant information. By reducing noise in the data, it becomes easier to extract meaningful patterns.

2.1.2.c) Stemming and Lemmatization

Stemming is the heuristics algorithm that cuts off common affixes to produce a base form. However, the resulting form is only sometimes an actual word. For example, the word "running" will have the stem "run", but the word "files" will have the stem "file", which is not in the English dictionary. The most widely used stemmer in the English language is an algorithm by Porter (1980).

In the next two decades, this algorithm will be developed in a high-level computer programming language as a package or library tool named Snowball (Porter, 2006). It has been designed to provide more concise results but still provides an unambiguous description of the rules for a stemmer in some cases.

Lemmatization is grouping inflected word forms, identified by the words "lemma". The word lemma is a valid word with a dictionary meaning, so the lemma of "flies" would be "fly." This means the word's reduced form will still have a meaning linguistically (Manning, Raghavan, and Hinrich Schütze, 2008). Both stemming and lemmatization aim to reduce inflectional word forms and sometimes derivationally related forms of a word to be a common base form.

2.1.3) Feature Extraction

2.1.3.a) N-grams

The N-gram is a significant frequent word technique for visualization that is highly effective for classifying documents. It uses samples of the desired categories rather than more complicated and costly methods such as natural language parsing or assembling detailed Lexicons (Canvnar and Trenkle, 2001). This technique can help visualize the chuck of words efficiently and clearly by considering single words or several words (bigram or trigram) to see the significant word or phrase of the dataset.

El Atawy and Abd ElGhany (2018) demonstrated the versatility of the N-gram frequency approach by applying it to a diverse set of problems, including speech recognition, translated words, and spelling correction. Their model, when tested on standard English datasets of misspelled words, achieved an accuracy rate of approximately 93%, highlighting the potential of n-grams to significantly enhance model performance.

2.1.3.b) Dimensionality reduction

Dimensionality reduction helps simplify high-dimensional data by converting it into a more manageable, lower-dimensional format while retaining key features. This technique reduces computational demands and facilitates the extraction of meaningful insights. Two common approaches used in this process are feature extraction and feature selection, which help eliminate irrelevant data and lead to more accurate machine learning models (Pudil and Novovičová, 1998).

Moreover, this method is particularly useful in data analysis tasks, such as creating regression or classification models. By reducing the data's complexity, it's possible to achieve better results compared to using the original, more complex data (Sulayes, 2017). Eventually, this approach enhances model performance and minimizes error rates by concentrating on the most important data.

2.1.3.c) Term-Frequency Inverse-Document Frequency

TF-IDF is a vital measure of words to document in a corpus or collection because some words appear more frequently in general (Leskovec, Rajaraman, and

Ullman, 2011). Term frequency (TF), introduced by Luhn (1957), is the weighting of a term within a document that is simply proportional to the number of times it occurs.

$$TF = \frac{Number\ of\ times\ a\ word\ \text{"X"}\ appear\ in\ a\ Document}{Number\ of\ Document\ of\ words\ present\ in\ a\ Document}$$

This metric measures the ratio of the word "X" appearing in a sentence by dividing the number of times it seems by the total number of words in the document. It shows how frequently these words occur, which can be used to analyse their significance further.

Karen Spärck Jones (1972) proposed a statistical interpretation of term-specificity called Inverse Document Frequency (IDF). This concept, which is a fundamental of term weighting, plays a significant role in the field of natural language processing, allowing us to quantify the specificity of a term as an inverse function of the number of documents in which it appears.

$$IDF = \log\left(\frac{Number\ of\ Documents\ present\ in\ a\ Corpus}{Number\ of\ Documents\ where\ word\ \text{"X"}\ \text{has appeared}}\right)$$

It calculated the logarithm of the fraction of all documents, showing in a corpus (list of documents) and document number where the word “X” was present. This value measures how important a word is within a document.

$$TF - IDF = TF * IDF$$

The combination of Term Frequency and Inverse Document Frequency measures the frequently appearing words in a specific document and finds how that word is essential in the corpus. This technique was often applied as a weighting factor in retrieval information and user modelling. Beel and his team (2015) organized a survey that showed that 83% of text-based recommender systems in online libraries used TF-IDF.

2.1.4) Clustering Models

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters) (Jain et al., 1999).

2.1.4.a) K-means

James MacQueen (1967) first introduced the K-means algorithm. He built upon an idea initially proposed by Hugo Steinhaus (1956), who was the first to suggest using K-means in multiple dimensions explicitly. Steinhaus's idea set the stage for what MacQueen later developed into a practical algorithm.

Lloyd (1982) created the K-means algorithm using the first standard method. He originally applied this method to pulse-code modulation, a technique used in signal processing to convert analogue signals into digital form. Lloyd's refinement was crucial in improving the algorithm's ability to handle real-world data, especially when dealing with noisy or complicated patterns. His contributions were essential in shaping the K-means algorithm into a widely used tool.

K-means clustering aims to separate a dataset of n points into k clusters. Each point is assigned to the cluster with the nearest mean, called the cluster centroid, which acts as the prototype for that cluster. The algorithm repeats this process, adjusting the centroids and reassigning points until the clusters are well-organized.

2.1.4.b) Hierarchical

Hierarchical Clustering is a cluster analysis method that builds a hierarchy of clusters, which can be visualized as a tree diagram called a "dendrogram" (Johnson, 1967). Unlike other clustering methods, it does not require the user to specify the number of clusters in advance. There are two types of this clustering method:

- Agglomerative (Bottom-Up) starts with each data point as its cluster, iteratively merging the closest cluster pairs until only one cluster remains or hits the stopping criterion.
- Divisive (Top-Down) begins with all data points in a single cluster, iteratively splitting the most dissimilar clusters until each data point is its cluster or another stopping criterion is reached.

When calculating the distance measure, Euclidean, Manhattan, and cosine distances are commonly used because they provide different perspectives on the similarity or dissimilarity between data points, each suited to specific types of data and clustering needs.

Hierarchical clustering has some drawbacks, such as an intensive computational cost, especially for large datasets. Additionally, splitting and merging occur; they cannot be undone because the algorithm is greedy and makes decisions based on local information without considering potential future implications.

2.1.4.c) Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a generative probabilistic model used to analyse a collection of documents (Yau et al., 2014). The essence concept is that documents are represented as random mixtures of latent topics, each defined by a word distribution.

This method was proposed by Pritchard, Stephens, and Donnelly (2000) and then was applied in machine learning by Blei, Ng, and Jordan (2003). They defined LDA as identifying the words associated with each topic and the topics that best describe each document.

One practical application of LDA is to discover topics in a collection of documents and then automatically classify each document based on its relevance to the topics discovered. LDA operates by assuming that each document is a mixture of topics and that a distribution over words represents each topic. In the topic discovery phase, LDA identifies these topics by analysing the co-occurrence patterns of words across the entire corpus.

2.1.5) Evaluate Models

2.1.5.a) Within-Cluster Sum of Squares (WSS)

WSS is a metric used to evaluate the quality of clusters in clustering algorithms, such as K-means. It measures the compactness of the clusters by calculating the sum of squared distances between each point and the centroid of the cluster to which it belongs. If this value is low, the clusters are more compact, which generally suggests good clustering quality (MacQueen, 1967). It helps determine the optimal number of clusters, often through the Elbow method, which is a technique to select the optimal number of clusters by plotting inertia against the cluster's number.

2.1.5.b) Silhouette score

Rousseeuw (1987) proposed this technique to interpret and validate the consistency of data clusters. It provides a clear graphical representation of how well each object has been classified or how similar it is to its cluster (cohesion) compared to other clusters (separation). The silhouette score is specialized for measuring cluster quality when the clusters are convex-shaped and may not perform well if the data clusters have irregular shapes (Monshizadeh et al., 2022)

Rousseeuw (1987) defined the significant silhouette score by this value ranging from -1 to +1, where a high value indicates that the object is well-matched to its cluster and poorly matched to neighbouring clusters. The clustering arrangement is appropriate if most objects have a high value. If many points have a low or negative value, the clustering configuration may have too many clusters. A clustering with an average silhouette width of around 0.25, 0.5, and 0.7 are weak, reasonable, and strong, respectively.

2.1.5.c) Davies-Bouldin Index (DBI)

Davies and Bouldin (1979) proposed the Davies-Bouldin Index (DBI), a metric used to evaluate the quality of clustering models. It measures each cluster's average "similarity ratio" with the one most like it. This ratio of each cluster is the sum of the intra-cluster distance (how close the points are within the cluster) and the inter-cluster distance (how far the cluster is from the nearest cluster), divided by the distance between cluster centroids.

A lower DBI indicates better clustering performance, implying that clusters are compact and well-separated from each other. This metric is beneficial for comparing different clustering models or selecting the optimal number of clusters.

2.1.5.d) Coherence score

This metric checks how well topics from topic modelling algorithms make sense. It looks at how closely the top words in each subject are related (Mimno et al., 2011). A high coherence score means the words in a topic often appear together in similar contexts, making the topic easier to understand and more meaningful.

The C_V Coherence value is particularly used for this purpose. Its score typically ranges from 0 to 1, with scores closer to 1 indicating higher coherence. It means that the words in the topic are highly related and appear together frequently in documents.

2.1.6) Sentiment analysis

Sentiment analysis uses natural language processing, text analysis, and computational linguistics to detect and analyse emotions and subjective information. It is commonly applied to customer reviews, survey responses, healthcare documents, and social media, with uses in customer service, medicine, and marketing (Liu, 2012).

Basically, it is classifying the polarity of a given text in the document or sentence. Whether the expressed opinion in a document, a sentence, or an entity feature is positive, negative, or neutral. The rise of deep language models, such as Roberta, has enabled sentiment analysis in more challenging data domains, news text information where opinions or sentiments are expressed less explicitly (Hamborg and Donnay, 2021)

2.2) Comparative Analysis

This section presents our agreement and disagreement ideas from the previous study: using LDA topic modelling to extract the related topic and missing examining the topic modelling with the dataset and visualization insights. We aim to contribute the application of topic modelling integrated with sentiment analysis to receive and understand enriching, informative insights.

2.2.1) Agreement and Disagreement Ideas

2.2.1.a) Agreement with LDA Model

This method can highlight uncovering hidden unstructured information rather than another clustering model, such as K-means. However, we will experiment to study whether this assumption is valid in the Findings section.

LDA is a powerful tool for analysing complex textual data. It can automatically identify and categorize related words into meaningful topics, even without prior knowledge of the data (Blei, Ng, and Jordan, 2003). This unique ability allows researchers to uncover hidden patterns and gain valuable insights into user engagement.

This capability is crucial for making sense of large volumes of TikTok video, enabling the identification of popular trends, themes, and user interests across a massive and constantly evolving platform.

2.2.1.b) Disagreement with Previous Study

The primary disagreement with the prior study is testing the topic modelling with the Trending TikTok videos data of Bakar's (2024) study. They implemented the text clustering model and provided the performance findings. Nevertheless, they

should have mentioned applying their application to discover insights and present visualizations of their trending video types.

The main missing point of her work should be highlighted more: We want to explore more TikTok datasets to display central themes and popular videos by applying the concept of Phan, Ninh, and Ninh's (2020) study to understand more about the trending of TikTok social media.

2.2.2) Contributing Different Ideas

2.2.2.a) Application to TikTok Content

We plan to execute three clustering models, K-mean, Hierarchical, and LDA, to explore which model should be implemented as the best-fitted application with our dataset to produce the video categories.

The insights that were explored were then produced as a thematic analysis to identify the hidden trend and theme of TikTok social media.

2.2.2.b) Integration of Sentiment Analysis

Applying the sentiment analysis idea to the dataset with video type attributes produced by the LDA model to identify the emotional declaration of each video from the description, we studied the trend of selecting positive or negative language for presenting their content on the TikTok community.

This tool is used to identify which content will reflect happiness or sadness by using words or sentences in their description. It helps researchers understand the potential mood of users who use TikTok. Additionally, understanding their user's interests and what they want to present on this social media is crucial because if it tends to be negative, we can avoid it just in time and suggest people who have never used TikTok choose to use other online social media instead.

Chapter 3: Research Design and Methodology

This chapter shows the processes that are designed to produce our interesting results from the TikTok dataset. First, the research design ideas will be presented, including defining the research questions and proposing the whole plan and analysis unit to address these questions. Then, explain our steps to gather the dataset from the TikTok website and briefly describe each data attribute. Eventually, we will

provide the approaches to handle many processes in our plan: analytical methods, description of tools and techniques, and concepts to track the validity of findings.

3.1) Research Design

This part presents our plan to solve the challenges in this work by defining the research questions and then proposing the flowchart to show the plan to address every question overall. In addition, we provide our unit of analysis to highlight our studied section. These will help readers understand the processes and target findings we want to explore, which can help them if they work on similar research in the future.

3.1.1) Research Questions

According to the literature review, we propose four research questions which this work focuses on to solve.

1. How can we classify unlabelled contents for video description?
2. Can we separate inappropriate contents?
3. How can we understand the main theme of TikTok social media used?
4. Which contents is the most popular content?

3.1.2) Overall Plan

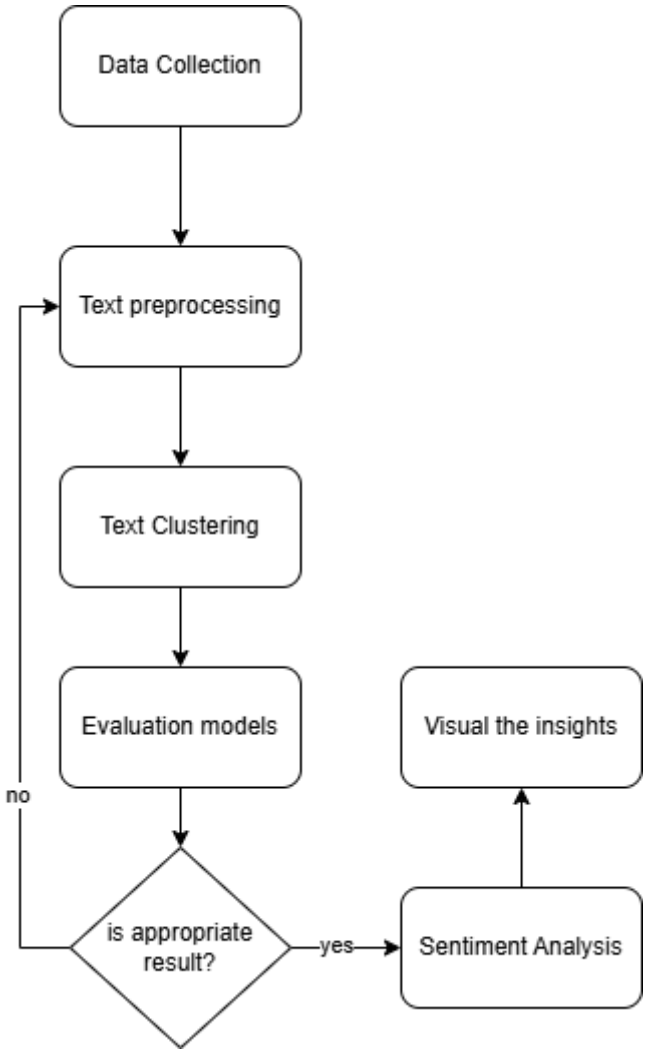


Figure 4: The flowchart shows the plan to solve all research questions

Figure 4 displays the main processes for addressing our research questions. We begin by collecting datasets discussed in more detail in [3.2]. Then, the dataset will be applied to text preprocessing to filter out unnecessary information before implementing the unsupervised clustering models. Not doing the preprocessing step affected these models' findings and prevented them from extracting enough relevant information; these are presented frequently in [4.1.1]. This step will address the first question.

Then, we measure the reliability and coherence of our clustering in the evaluation model's section. Suppose it cannot provide reasonable results. In that case, we will move back to do the preprocessing again to filter more irrelevant

information out as a refinement parameter to improve the performance of the model and, following that, apply the sentiment analysis tools to measure how much positive or negative language is used in each video to describe their content as a description, addressing the second question.

The final step is designed to solve both the third and fourth questions by discovering the insights of TikTok datasets. This will help us see the central theme and more clearly describe the popular contents of this information since the text preprocessing process to the last step will present more details in the next section.

3.1.3) Unit of Analysis

In this study, we have chosen individual video descriptions from TikTok as our unit of analysis. This choice is crucial because it enables us to examine closely how different types of content are categorized, which is essential for developing the application outlined in the Finding section. This application is part of our strategy to address the first research question.

To tackle the second research question, we will employ a sentiment analysis tool to examine our assumption whether this tool can separate inappropriate content within these video descriptions or not.

Our dataset includes various content types, such as variety shows, sports clips, and challenge videos. By analysing these diverse categories, we hope to identify the dominant trends on TikTok and understand the types of content that resonate most with users.

This analysis will help answer our third and fourth research questions, which focus on understanding content popularity and exploring how our findings can be applied to broader contexts.

3.2) Data Collection

This section shows the overall data collection process, beginning by presenting step-by-step instructions on gathering the dataset from the TikTok website and then briefly explaining the data description so readers can understand how we collect this dataset and what critical features we focus on.

3.2.1) Instruments Used

The collection data method, web scraping, can help researchers gather information on online websites by writing custom scripts. In theory, Mitchell (2018) stated that web scraping is collecting data through any means other than a program interacting with an API, which is the application program that people write to do something; it is easy to implement. It is most accomplished by writing an automated program that queries a web server, requests data, and then parses it to extract the required information. We selected writing JavaScript code to scrap the dataset, using puppeteer libraries, a compact tool for web scraping to capture those APIs and get the required data.

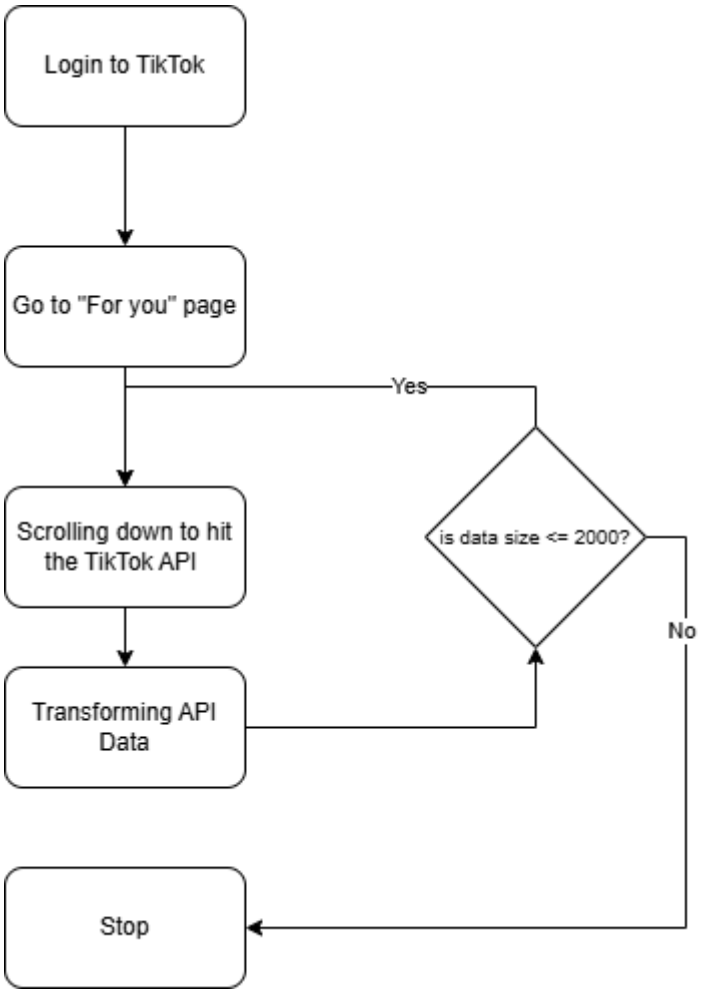


Figure 5: The automated web scrapping data processes

Figure 5 depicts the automated process of gathering TikTok video information, starting by logging into the account used to watch content, mainly used in the United Kingdom area, following that, moving to the “For you” to enter the main feel that we can grab some needed TikTok API that provides the video information and then parsing it into our expected data form, doing iteration until reach to the limit our

dataset length, such as data size = 2,000. We kept datasets for around two weeks, from 25 July 2024 to 8 August 2024. By each day, we scrapped the data around 1000 – 2000 videos.

3.2.2) Data Description

We collected the TikTok dataset 20,243 videos, including ten attributes (id, description, collectCount, commentCount, likeCount, playCount, shareCount, duration, quality, and author). This dataset will be gathered from the TikTok website through logging by our private user; we will write an automated script to scrape this data that will be mentioned in the previous section [3.2.1]

The statistical information of contents will be analysed after building the text clustering model to visualize the insights of each video group and understand the primary theme of TikTok social media. Regarding training the clustering model, the video description is used as a critical variable for separating each video's cluster by preparing the dataset before bringing it to train the model, as mentioned in [3.1.2]. Table 1 displays a brief description of each attribute.

Features	Description	Data type
id	The unique number of video data, indicating the distinct of each content	number
description	Text description of the video, including hashtags	text
collectCount	The amount of collected content	number
commentCount	The amount of commented of content	number
likeCount	The amount of liked of content	number
playCount	The amount of viewed of content	number
shareCount	The amount of shared of content	number
duration	How long is the video (second)	number
quality	The quality of video (pixel)	text
author	Nickname of creator	text

Table 1: The data description of the TikTok video dataset of each attribute

3.3) Data Analysis Methods

3.3.1) Analytical Process

This part presents more details of each analytical process of overall planning, including what methods and concepts are applied to address our research questions.

3.3.1.a) Text preprocessing

1. Inputting the textual data into the text cleaning function that removes the punctuation, emoji characters, single, double, and triple-character words, and non-English words

2. Then, implement the text filtering to remove the stop words and use the lemmatization to convert various words into their standard form, such as flies -> fly

3. After that, we apply exploratory data analysis to understand the dataset more fully, and then we will train the model and analyse the insights, respectively.

3.3.1.b) Text clustering

1. Importing the TfidfVectorizer to convert the document information into numeric matrix data type, using to train with the partitional clustering models

2. Before training models, convert the dataset by using PCA to reduce the dimensionality of features, extracting more relevant data

3. Doing the iteration of training clustering models with various k variables, gathering the WSS metrics to plot the elbow chart to indicate the optimal k value

4. Implementing the clustering model with the optimal k value and measuring the evaluation metrics (Silhouette, Davies Bouldin)

5. Before building the LDA model, the tokens are generated, and the n-grams technique is applied to make the trigram (triple-related words) tokens.

6. Then, transforming the token into dictionary type and converting the dictionary into a corpus, the textual word information is used to separate the topics of each video.

7. Do the hyperparameter tuning with LDA with various k-value, alpha, and beta variables to find the best parameters to build the optimal LDA model

8. This will address the first question after receiving the final LDA model, measuring the coherence score, and plotting the visualization of the distribution findings of each topic by the LDA model.

3.3.1.c) Evaluation models

1. Select the best model between K-means, Hierarchical, and LDA models to implement the clusters.

2. Interpret the cluster label of the optimal model using human judgment ideas to see if our findings make sense; if they do not, we will move back to text preprocessing to filter out more irrelevant information to extract the meaningful data as much as possible.

3.3.1.d) Sentiment Analysis

1. Applying the Vader sentiment analysis tool to measure the positive and negative score of each video

2. Determining the emotional type of each content based on these scores

3. Visualizing the findings to understand the hidden insights will address the second question.

3.3.1.e) Visualization Insights

Implementing the dataset with topic categories by topic modelling to plot various findings to display insights for studying the primary theme of the TikTok dataset, understanding the trending of each video category, and discovering the popular contents in this social media, addressing the last two questions.

3.3.2) Tools and Techniques

This part explains the tools and analytical techniques that we used for each analytical process [3.1.2], providing brief information.

3.3.2.a) Text mining

Generally, the NLTK Python package is used for text preprocessing, such as text filtering, including removing stopwords (stopwords) and lemmatization (wordnet). It is also used for sentiment analysis to find the emotional tone of textual information.

Additionally, I imported some additional information to work with the "words" English word dictionary information, using it to check English words.

The other significant package is "word could," which presents the frequent words in a beautiful, easy-to-understand chart.

3.3.2.b) Exploratory Data Analysis

They use commonly used libraries for data analysis, including pandas, NumPy, matplotlib, and seaborn, to explore the overall dataset and provide tools to visualize various charts. The SciPy package calculates the mathematical formula to find the significant value to analyse the results: the distance between each point of clustering models. It also provides the functions to find the hierarchical models: linkage, dendrogram, and cluster.

3.3.2.c) Machine learning models

- Sklearn tools are used to build the unsupervised model, do feature extraction (TF-IDF and PCA), and calculate the evaluation metrics (Silhouette score and Davies Bouldin Score)
- Gensim package is used to build topic modelling, including simple text preprocessing function, corpora (used to transform the text data into the dictionary and corpus data types), and calculation coherence metrics. Another additional tool, "pyLDAvis," plots a pretty chart to show the topic distribution of topic modelling; it helps users understand the LDA model's overall findings.

3.3.3) Validity Concepts

The primary objective of our application's validity is to check the mathematical metrics that measure whether a model is well-organized. We separated similar information in the same group and then applied the model to produce the cluster topics to consider whether those categories are appropriate with sample data. Suppose it found a lot of unreasonable findings.

In that case, we will adapt by moving back to preprocessing the dataset more or adjusting the parameters to build the clustering model to receive suitable results with our dataset. Then, we interpreted the categories based on our decision that they were appropriate. We expect only some samples to get perfect results, but they should have most reasonably more than mismatched categories.

3.4) Research Gap

This study only uses text description and hashtag information to train the clustering model. It will ignore comments data because the number of comments on each video is diverse. Gathering those data may be complicated and take a lot of time, compared with our research period time constraint, so we will analyse only the video title and hashtags.

We mainly focus on English, which will filter out other languages. Still, some other languages need to be added because they may use the English alphabet, making them lose sight of the analysis data. We cannot apply sentiment analysis to analyse the video and music file information because we do not have this data. Another constraint is emoji analysis. This research filters emoji characters out. It analyses text only. Thus, there are research gaps in this study.

Chapter 4: Findings

This chapter outlines the iterative process used to achieve our desired results. We begin with text preprocessing to filter out irrelevant information, comparing the original and cleaned datasets using word cloud charts. Next, we explore the datasets for an overview, including a statistical summary and feature distributions. Then, we implement partitional clustering models, using iterative methods to determine the optimal number of topics.

Following this, we build a topic model (LDA) using hyperparameter tuning to identify the dominate topic of each video and select the suitable model to interpreter the video category. We then conduct sentiment analysis to assess the emotional tone in video descriptions. Finally, we present our findings in bar charts, highlighting the significance of the main themes, popular content, and insights to understand our TikTok datasets.

4.1) Text preprocessing

First of all, visualizing the word cloud of the whole dataset without refinement to see the overview of information and then produce text cleaning to remove noise information, such as punctuation and standalone prefix and suffix word, and unmeaning words to following that execute text lemmatization to reduce the several

forms of a word, converting into common base or root form, it can help improving analyse to extract more meaningful data.

Figure 6: the word cloud list of frequent texts without doing text preprocessing.

Figure 6 displays the most frequently words are using on the description. Although we can grab some words, such as funny, food, game, movie, football, it does not enough or clear to categorize the topic of each content because editors often use a lot of unmeaning words: fyp, viral, trending. These will affect to our decision with we analyse the machine learning with those data. Thus, before building unsupervised model, we will filter weird or noise text data out to enhance the performance of model when apply this information, following by the process on Figure 8.

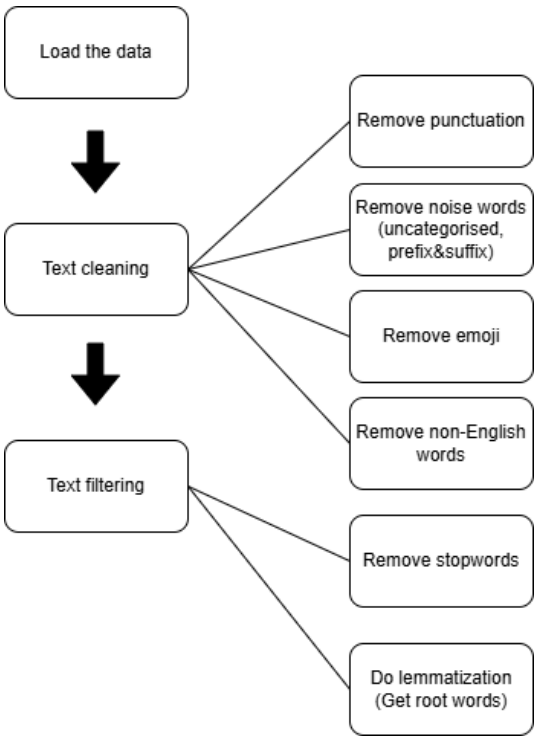


Figure 7: the text preprocessing that filters the redundant textual data out to extract more relevant information.

Figure 7 displays the step will remove unnecessary information, before taking the textual data to produce the text clustering model. We focus on four things that are removed from the dataset: punctuation, uncategorised and single prefix, suffix words, emoji, and non-English words. These removing values will list on the tables that show on the appendix section.

4.1.1) Text Filtering

After finishing text cleaning process, we will put the data into text filtering function that will filter the list of stopwords, such as i, he, she, do, does. Additionally, applying “Lemmatizer”, tools from NTLK python package in wordnet field, to convert the various word forms to common form, decreasing the duplication information when we build machine learning in the next section. Following that we will apply the filtering text information, plotting on Word Cloud chart again to compare with the previous chart (without preprocessing).

Figure 8: the word cloud list of frequent texts with doing text preprocessing.

Figure 8 demonstrates that it can extract more meaningful information, such as movie, family, game, song, and football. We can see the main theme of TikTok. Rather than consider Figure ii, it will be hard to classify the theme. We will deeply analyse this information in the following subsection.

From this process, we must take a lot of videos out, roughly half of the total, to execute the machine learning model because we try to remove other languages videos (non-English), spam videos (often using single or double-character words), and video without description to scope the analysis section narrowing to receive more relevant results. Moreover, non-description videos are our limitation. We cannot categorize them by considering only text descriptions. Even if some weird words still appear in the findings, it makes it more understandable to apply these text data to the next step.

4.1.2) Unigram and Bigram Analysis

This part shows counting the top frequent words as single words and bigram, a sequence of two adjacent elements from a string of tokens, typically referring to words or characters in a text. It is used to capture the relationship between consecutive items.

Displaying the comparison of the findings between applying text filtering and without using it, describing why this technique is crucial, and then presenting filtering outcomes as bar charts to identify the significant words and give some comments to explain how they influence.

Bags of words without text filtering

Words	Frequency
fyp	3037
foryou	2148
viral	1721
foryoupage	1339
tiktok	831
funny	759
fypシ	658
trending	476
uk	400
viralvideo	339

Bags of words with text filtering

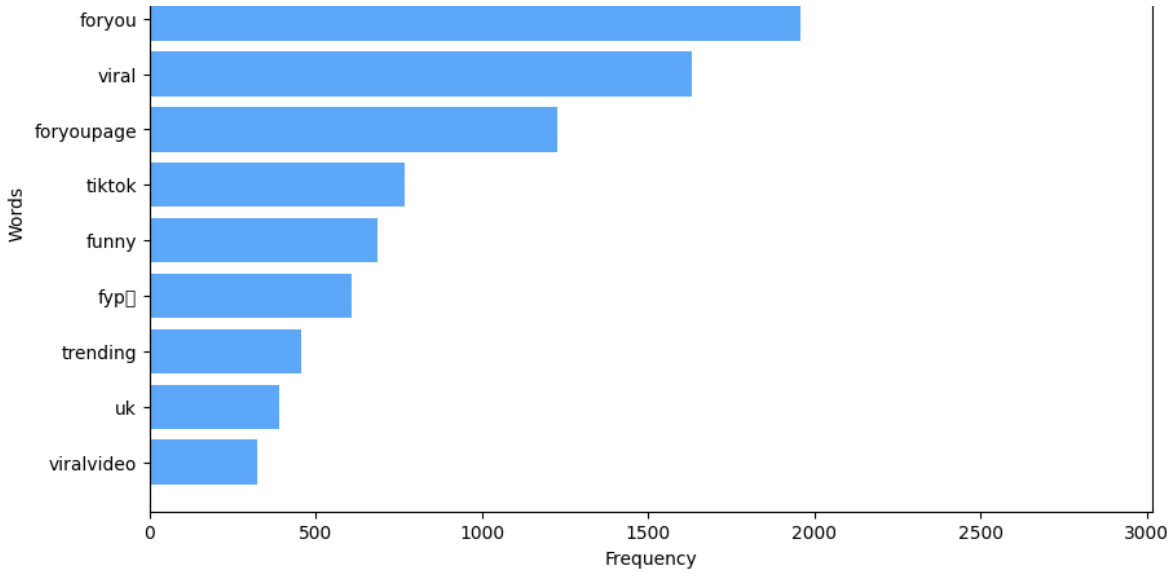
Words	Frequency
funny	759
funnyvideos	246
love	214
london	204
like	206
movie	201
prank	193
football	182
time	177
game	175

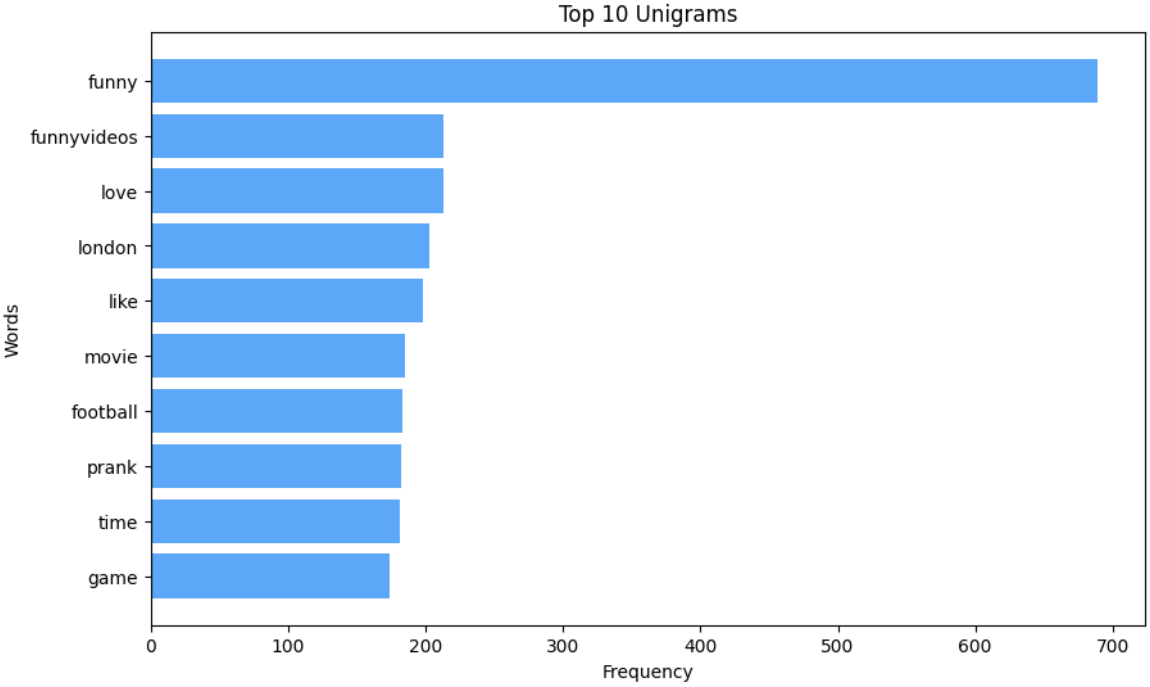
Bigram	Frequency
fyp foryou	480
foryou foryoupage	409
fypシ viral	295
fyp viral	217
foryou fyp	214
fyp foryoupage	181
foryoupage foryou	127
foryoupage fyp	122
fyp fypシ	119
fyp fyp	119

Bigram	Frequency
funny funnyvideos	92
mufc manutd	63
chelseafc premierleague	50
prank prank	35
football soccer	28
dune dune	25
original content	24
simonsquibb entrepreneurship	23
entrepreneurship angelinvestor	23
valorant valorantclips	23

Table 2: the comparative analysis of findings: with vs. without text filtering.

Table 2 shows that the findings without text filtering received various unmeaningful words, both unigram and bigram, such as type, for you, and groupage; these words are used as keywords hoping to reach the TikTok algorithm to bring their video to the trending feed. On the other hand, applying text filtering helps to extract more meaningful information and eliminate those previous words; it can improve the clustering model to classify the video information efficiently by considering more relevant text information. Figure 9 shows that words about humour, such as funny and prank, are frequently mentioned in the content. Other interesting words are football, movies, and games.


foryou
viral
foryoupage
tiktok
funny
fyp
trending
uk
viralvideo
Words
0
500
1000
1500
2000
2500
3000
Frequency


Top 10 Unigrams
funny
funnyvideos
love
london
like
movie
football
prank
time
game
Words
0
100
200
300
400
500
600
700
Frequency

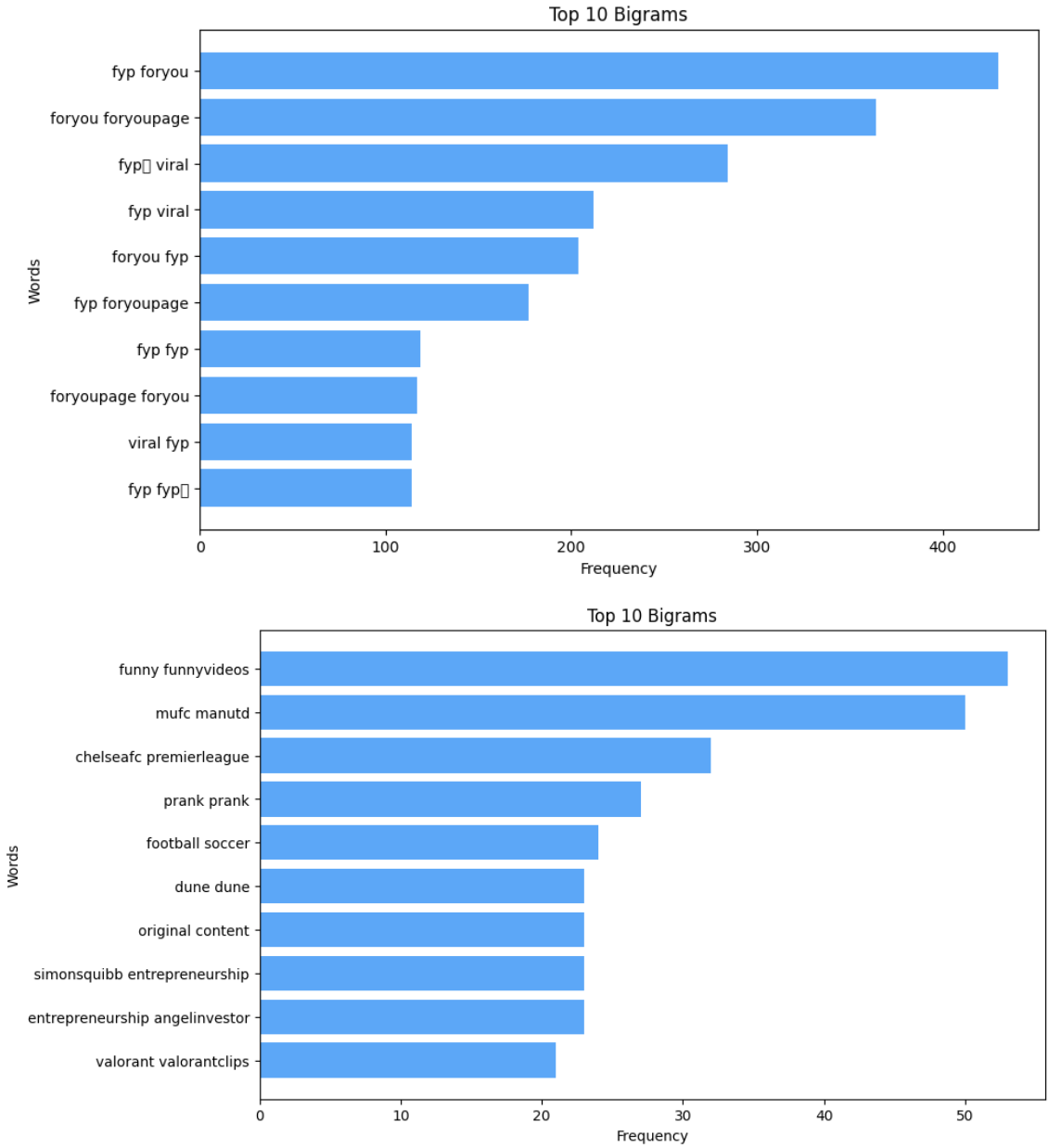


Figure 9: The bags of frequent words from the pre-processed and original datasets, including unigrams and bigrams.

Regarding Bigram's analysis, they provided insights that this platform often talks about football clubs, such as Manchester United and Chelsea FC teams. Interestingly, the entrepreneurship content shows that some people use TikTok to share their experiences and build a business community here.

4.2) Exploratory Data Analysis

4.2.1) Statistics Summary

Feature	count	mean	std	min	25%	50%	75%	max
collectCount	9836	12,334	64,986	0	140	901	4,292	2,200,000
commentCount	9836	1,609	9,507	0	26	141	616	405,800
likeCount	9836	172,940	990,010	0	2,153	13,000	59,300	37,700,000
playCount	9836	2,134,906	11,382,166	0	62,775	320,250	959,325	552,500,000
shareCount	9836	9,171	59,454	0	46	416	2,886	2,000,000
duration	9836	70	80	0	22	60	78	912

Table 3: the statistics summary of the numeric attributes

Confident Interval 95%	Min	Max
playCount	1,909,940	2,359,872
likeCount	153,373	192,508
commentCount	1,421	1,797
collectCount	11,050	13,619
shareCount	7,996	10,346

Table 4: the 95% confident interval of numeric features

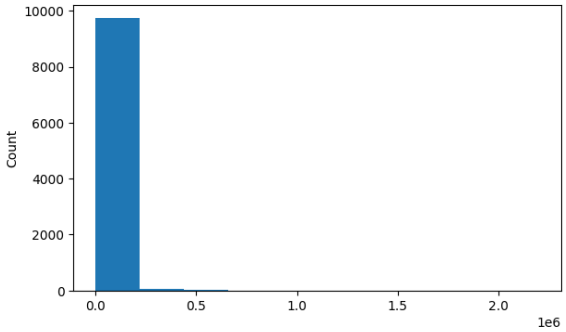
Table 3 shows the basic statistics of each numeric feature: the number of comments collected, play, share, and video duration. Table 4 illustrates a confidence interval of 95% for significant attributes, with the amount of play and comment videos roughly between 1.9 and 2.3 million views and 1,421 and 1,797 comments, respectively.

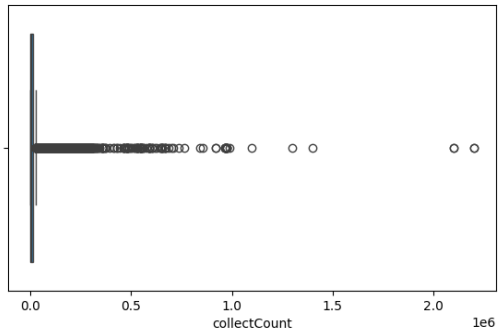
The proportion of comments per view is around 0.07%, reflecting that social media rarely exchanges opinions, just participating in watching the content. However, the standard deviation of these statistics is unusual and is greater than the average; the leading cause may have some minor popular content that has an enormous watching rate, more than the average of the whole content, which may influence the main trending of this social media. These popular contents will be analysed in [4.5.2] to address the last research question.

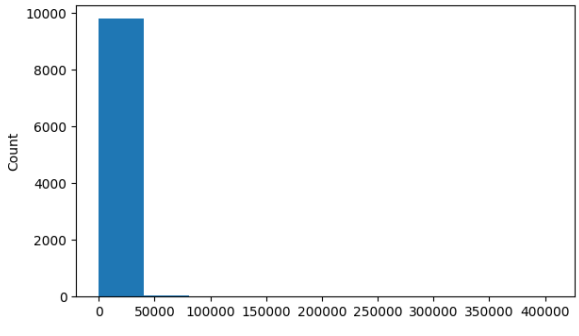
In addition, as the like rate count is around 150,000 to 192,000 likes, roughly 8 percent, it is not much engagement, but it is okay for the participating part. Interestingly, the collected and shared content is 11,000 to 13,600 collected videos and 7,900 to 10,300 shared contents. It is around 0.03 and 0.05 percent, showing that users keep the inspiration or interesting clips to watch again or share them with their colleagues to exchange information in their communities. This information is one significant part we use to explore and address our third question.

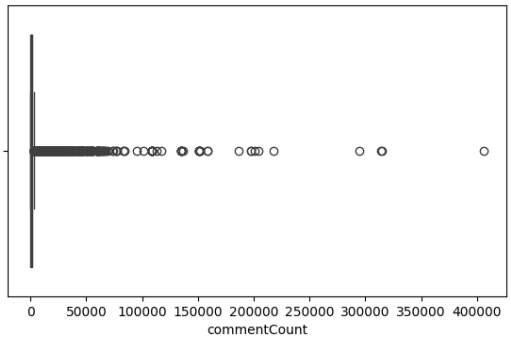
4.2.2) Observing the distributions

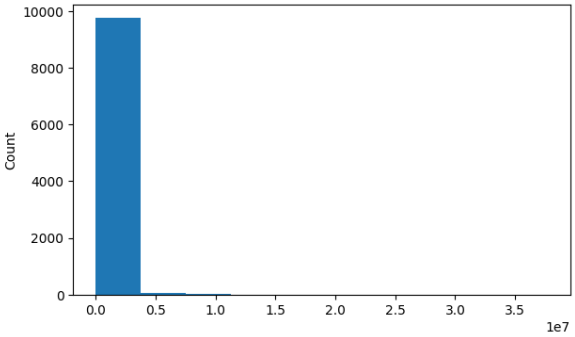
The final step for exploring dataset, considering the distribution of numeric features to know the range of value of these important attributes for applying this information to understand the participation and engagement values to indicate the influence of each content that how those videos have more power to the online community?

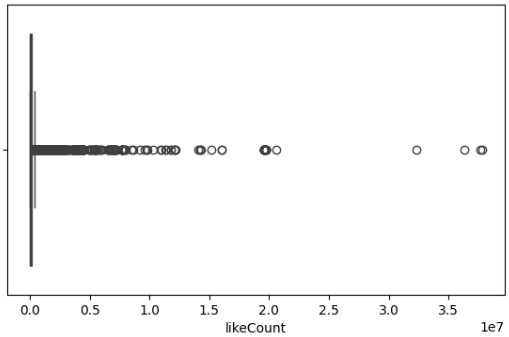

Count
1e6

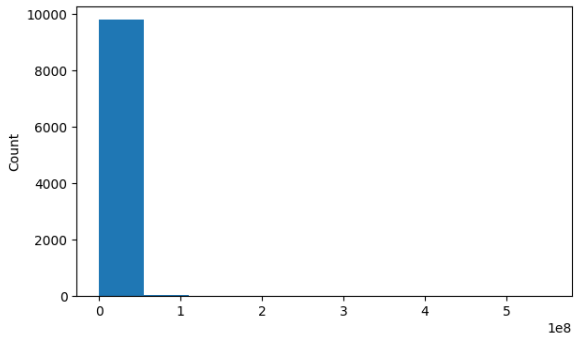

collectCount
1e6

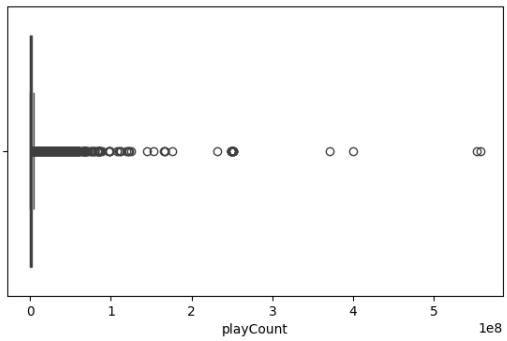

Count


commentCount


Count
1e7


likeCount
1e7


Count
1e8


playCount
1e8

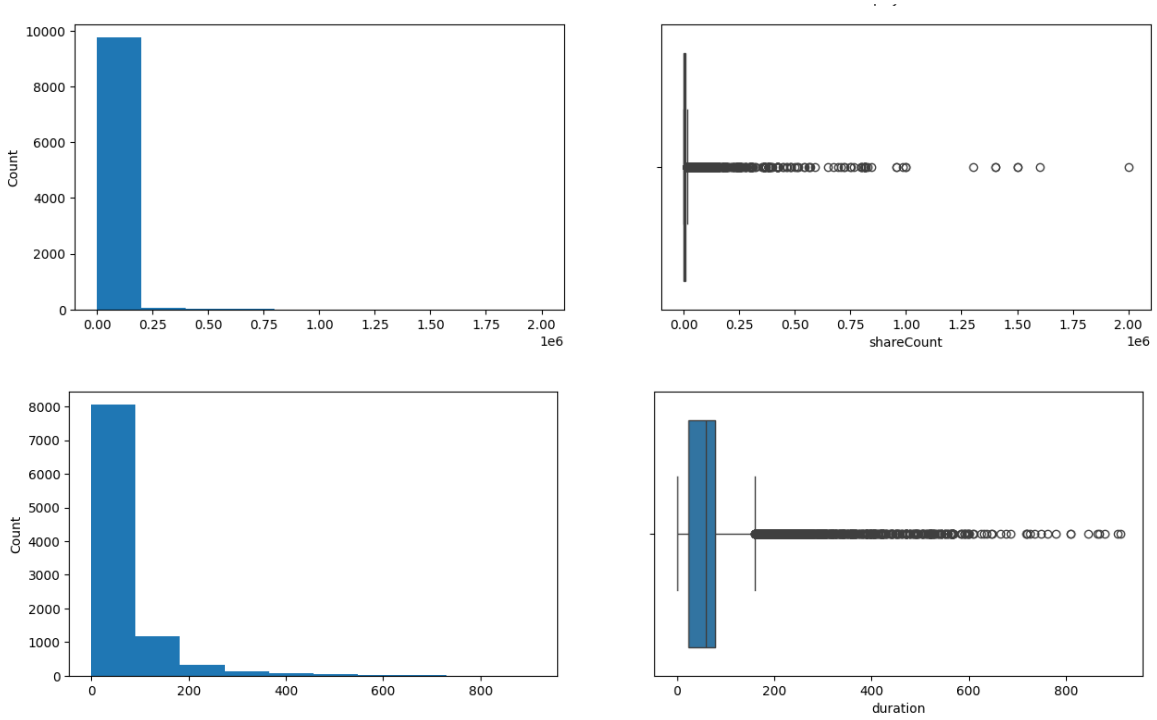


Figure 10: the distribution of each numeric features, showing that they are right-skewed distribution.

According to the Figure 10, content statistics stats have right skewed distribution, meaning that they have some outliers that those videos are popular contents, showing outstanding watching stats, it has more influence on every feature, as we can see the similarities scatter plot patterns. These participation stats help us understand that most videos are watched but have few views compared to some videos that can produce outstanding engagement stats. The last part of this chapter provides more details to address the main theme of our datasets, and we can search for popular content as follows to solve our last question.

4.3) Text Clustering

This step presents the step to build the text clustering models using one topic modelling algorithm and two partitional clustering algorithms, which provide the video groups of each video.

4.3.1) Partitional Clustering

The figure below shows the step to produce the partitional clustering, this subsection will present two unsupervised machine learning: K-means and Hierarchical algorithms have been chosen because our dataset is small.

They can provide distinct cluster centroids, making it straightforward to understand and visualize with fewer data points. They can also provide a visual dendrogram, which may help to interpret easily. As a first step, transforming the textual data into document term frequency matrix, making the computer can compute the numeric information to build the model.

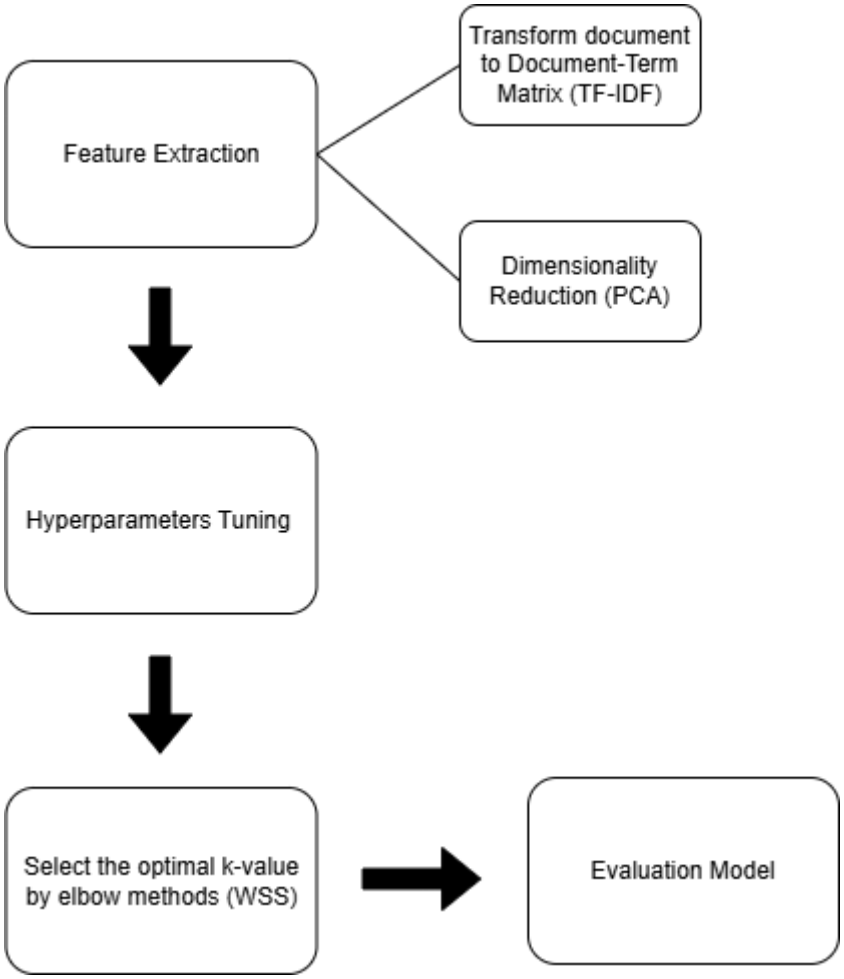


Figure 11: the empirical steps to build the partitional clustering (K-means, Hierarchical)

After that, we will apply principal component analysis (PCA) to transform the high-dimensional matrix data into lower-dimensional form; it can help eliminate

irrelevant information. Then, we will do the tuning hyperparameter, execute iterations of clustering algorithms with various k-values, and plot the within-cluster sum of square (WSS) to plot the elbow method to observe the optimal k-value.

The last step is to apply the optimal value to the clustering model and measure two evaluation metrics (Silhouette and Davies Index) to see how our model performs.

4.3.1.a) Term Frequency and Inverse Document Frequency

We will use sklearn's TfidfVectorizer tool to transform our dataset in the feature extraction step. First, we convert the text to Unicode, so it is compatible with the computer. Then, we use the vectorizer to create a document-term matrix, specifying an n-gram range of [1,3] to capture single, double, and triple-word combinations. Finally, the matrix is converted into an array for further feature extraction and complexity reduction in the next steps.

4.3.1.b) Principal Component Analysis

The PCA function was applied to transform the list of documents matrix, using (n=50) as the experimental value. Then, the line chart was plotted to observe the potential optimal number of components based on the elbow concept, noticing the critical point where it dramatically and gradually dropped.

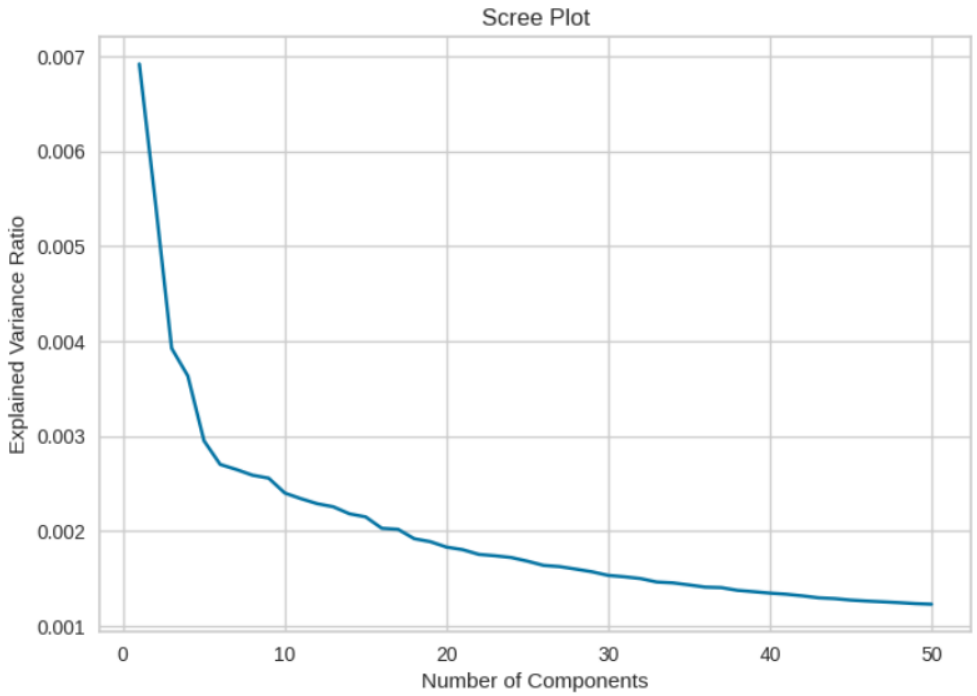


Figure 12: the explained variance ratio of principal component analysis with trial number = 50.

Figure 12 depicts that in the range n = 1-10, the variance ratio drops significantly since n > 10. This value consistently declines to change, and at the position, component numbers equal around seven. We can see the elbow pattern; we selected that point because it is the required point mentioned above. Subsequently, the document term matrix lists retransform using PCA tools with n=7. These data are ready to build the machine learning step, which will be explained in more detail in the next subsection.

4.3.1.c) Hyperparameter Tunning

This process will do the experiment of two clustering, defining range of k-value since two to nineteen. Using the document term matrix data that applies the PCA approach and without transform PCA method to compare two approaches to see the different outcomes metrics how PCA can help to enhance the performance of model. Then exploring the optimal k-value by applying the elbow concept to find this value. Finally, we will measure evaluation metrics to determine if our clustering model is well-organized.

4.3.1.d) K-means

Table 5 compares two input features: one without using PCA and one with using PCA for data reduction. Dimensionality reduction improves K-means performance, as evidenced by closer distances between data points and centroids, leading to lower WSS values. It indicates better data organization around centroid points.

Clusters	WSS
2	9,542.84
3	9,521.19
4	9,508.59
5	9,503.00
6	9,466.95
7	9,469.95
8	9,437.92
9	9,423.56
10	9,424.88
11	9,397.41
12	9,374.40
13	9,393.74
14	9,352.40
15	9,369.49
16	9,357.34
17	9,330.00
18	9,331.36
19	9,279.70

Clusters	WSS
2	212.22
3	179.20
4	148.61
5	128.52
6	109.43
7	91.51
8	76.86
9	63.42
10	55.70
11	50.83
12	43.13
13	39.39
14	34.76
15	30.49
16	27.90
17	26.32
18	25.20
19	23.66

Table 5: Comparing the WSS metrics between feature data without PCA and with applying PCA (K-means Clustering).

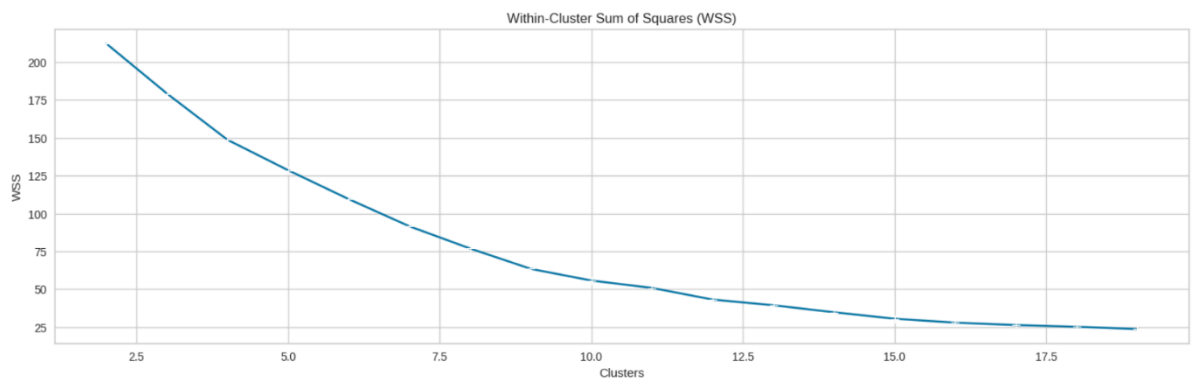


Figure 13: plotting the WSS to indicate the optimal value of K-means models (Elbow method)

Figure 13 displays the elbow chart for WSS values, identifying k=11 as the optimal point where changes become gradual. We used this k-value in the K-means algorithm to build the clustering model. The model is well-structured, with a silhouette score over 70%, indicating well-organized centroids, and a low Davies-Bouldin Score, reflecting good separation and compactness of clusters.

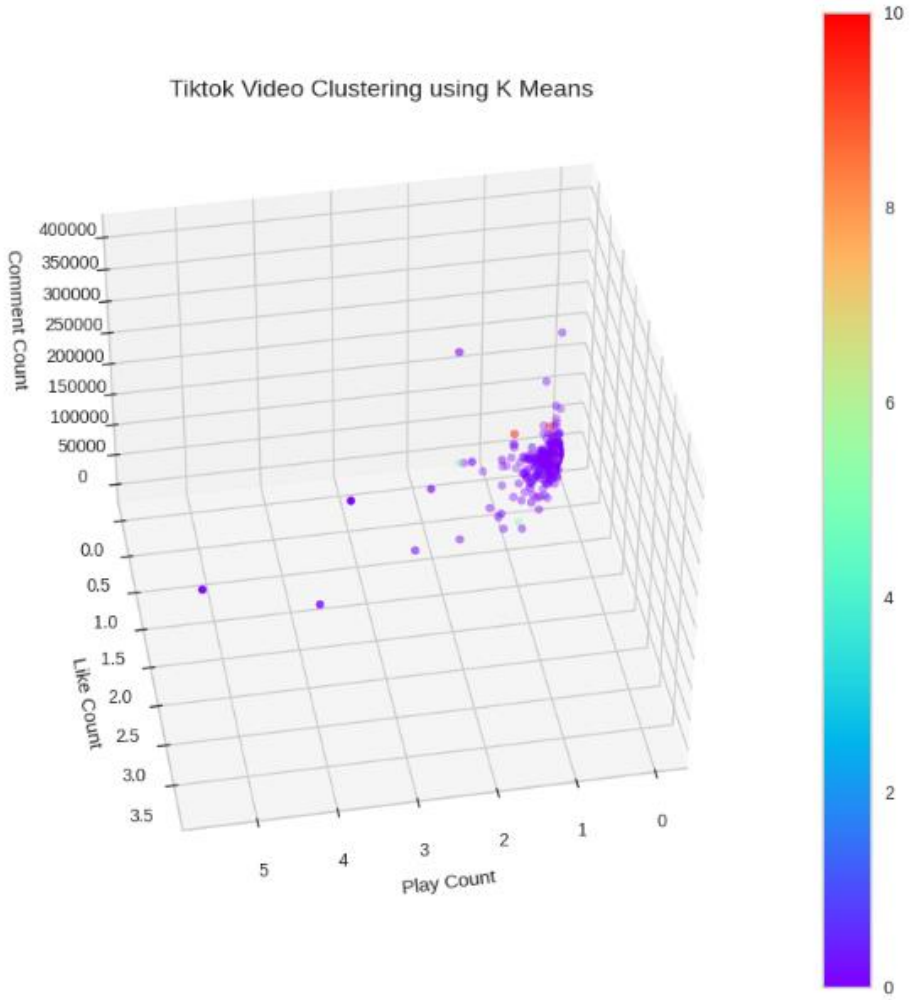


Figure 14: The 3D scatter plot of the number of watching, like, and comment times of TikTok videos.

4.3.1.e) Hierarchical

This method creates a dendrogram chart that shows a hierarchical structure by merging smaller clusters into larger ones based on their similarities. The data points are grouped according to how similar they are, and the dendrogram displays this by positioning each piece of information to show the distance, or dissimilarity, between clusters. We can make horizontal cuts across the dendrogram at different levels to find the best number of clusters.

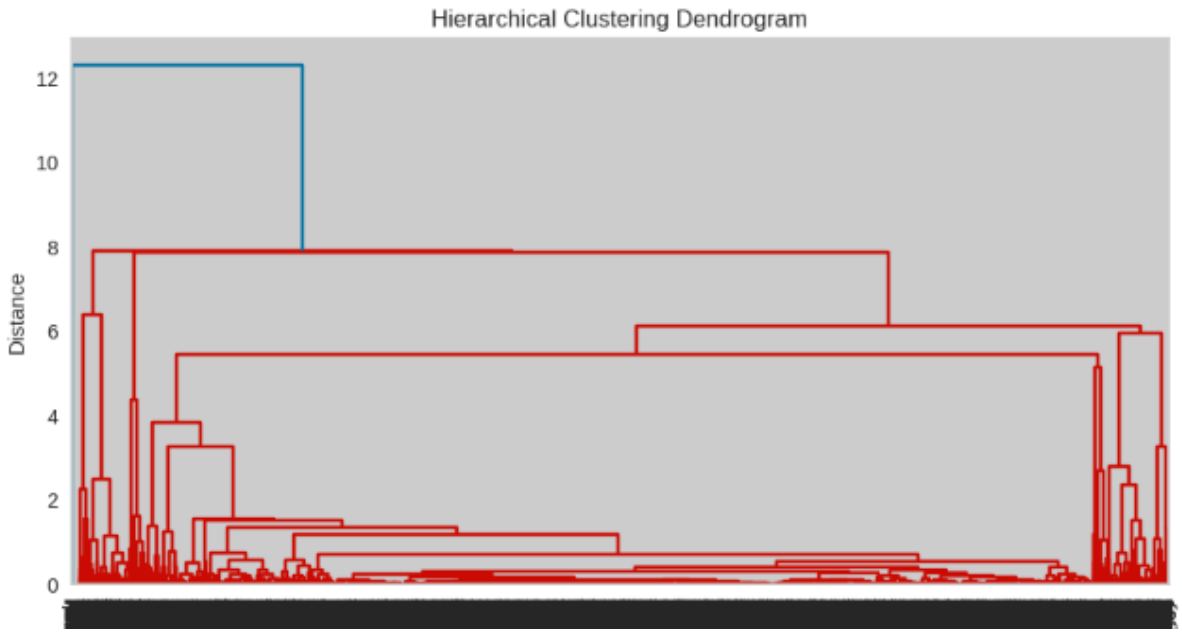


Figure 15: Dendrogram Showing the Hierarchical Separation of Videos into Categories based on Content Similarity

Clusters	WSS
2	9,549.25
3	9,520.88
4	9,495.66
5	9,476.45
6	9,459.74
7	9,443.08
8	9,426.70
9	9,410.58
10	9,395.57
11	9,381.03
12	9,366.60
13	9,352.48
14	9,339.09
15	9,326.11
16	9,313.16
17	9,300.76
18	9,288.62
19	9,277.00

Clusters	WSS
2	218.78
3	187.61
4	156.63
5	136.26
6	117.54
7	99.86
8	84.93
9	71.70
10	62.18
11	54.81
12	49.45
13	44.13
14	40.18
15	36.51
16	33.38
17	30.56
18	28.01
19	26.68

Table 6: Comparing the WSS metrics between feature data without PCA and with applying PCA (Hierarchical Clustering).

According to the above table, presenting two different outcomes from two input features. It can produce the similar results when compared to K-means clustering, but it has a bit different finding.

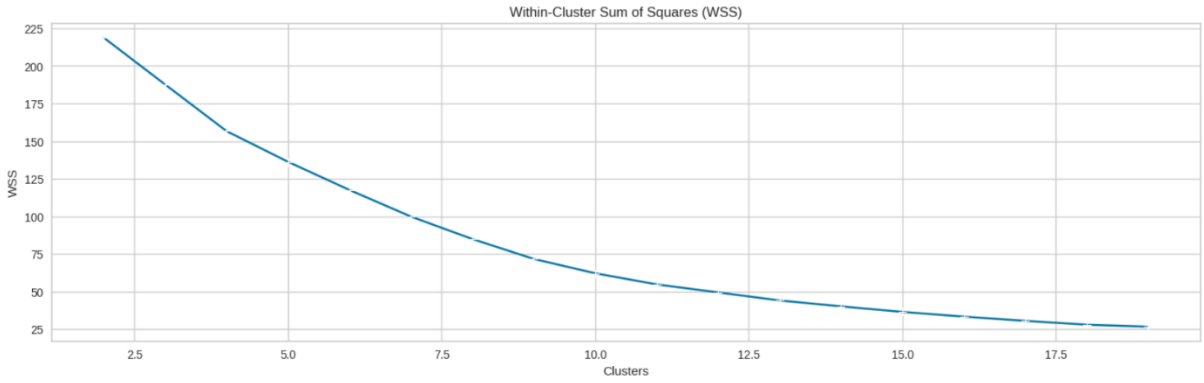


Figure 16: plotting the WSS to indicate the optimal value of hierarchical models (Elbow method)

Figuring out the optimal number of groups for the data is challenging because the graph looked unclear. When we explored these numbers in Table 6, we noticed a possible turning point in nine groups. It is where the numbers started to slow down, suggesting it is a suitable candidate for the optimal number of clusters.

The unsupervised model demonstrated impressive performance in the evaluation section, achieving a Silhouette coefficient of 71.09% and a Davies-Bouldin Index of 0.698. These metrics indicate that the model effectively clustered TikTok video information into distinct groups. The data points within each cluster are very similar and positioned relatively close to their respective cluster centres.

4.3.2) Topic modelling (Latent Dirichlet Allocation)

4.3.2.a) Step-to-implementation

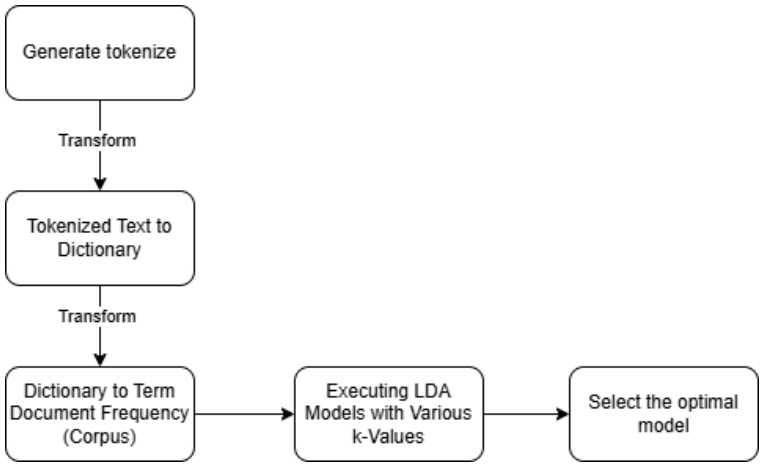


Figure 17: the steps to produce the topic modelling (LDA).

We generate tokens by converting the textual data from a selected TikTok video dataset. These tokens will be transformed into a dictionary datatype. We will then transform these dictionaries into a corpus format through a two-step process.

After that, we will execute topic modelling using LDA and experiment with several parameters: alpha and beta (0.01, 0.11, 0.21, …, 0.91) and the number of topics from 2 to 9. By applying the concept of hyperparameter tuning, we aim to find the optimal configuration that yields the best results by calculating the coherence score to measure its performance and effectiveness in each iteration.

Finally, we visualize the LDA clustering to represent the topic distributions and provide comments and insights on the model's performance and organization, highlighting any significant findings or observations.

4.3.2.b) Analysis Findings

Table 7 shows the outcomes from executing the LDA with n=8 and various alpha and beta values. We chose the eight topics because this value produces a high coherence score. Next, we observe the parameters in topic = 8. We can find the best performance in the model. Obviously, presenting this position in Figure 18 can provide the highest coherence score at 0.632, which shows that this model is well-structured and offers meaningful findings.

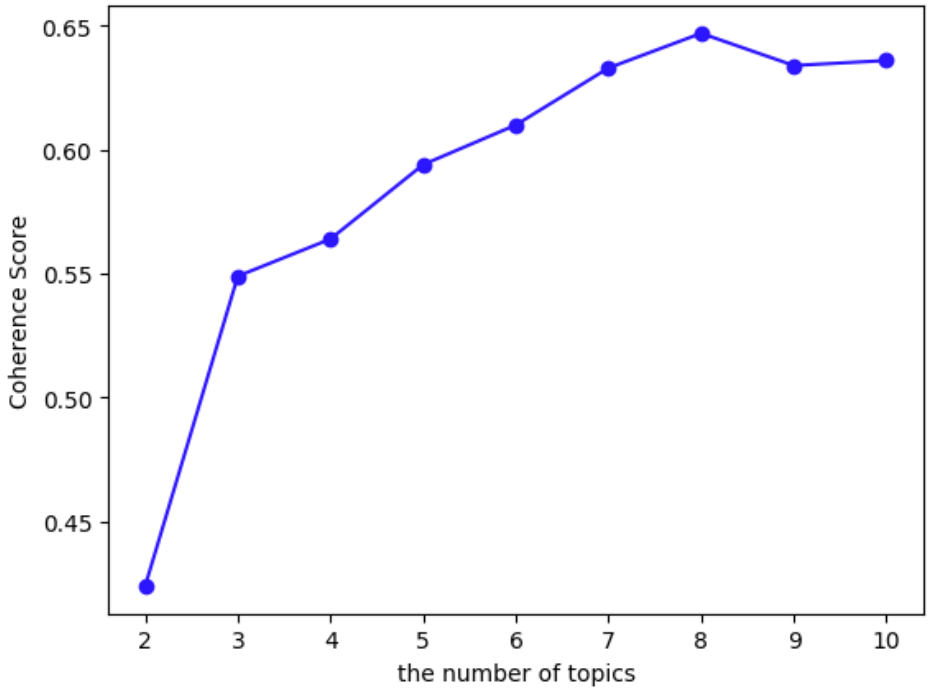


Figure 18: Determining the Optimal Number of Topics: A Coherence-Based Approach

Topics	Alpha	Beta	Coherence
8	0.71	0.61	0.515553
8	0.71	0.71	0.508503
8	0.71	0.81	0.445771
8	0.71	0.91	0.459579
8	0.81	0.01	0.632546
8	0.81	0.11	0.584414
8	0.81	0.21	0.553754
8	0.81	0.31	0.557925
8	0.81	0.41	0.552067
8	0.81	0.51	0.500252
8	0.81	0.61	0.469152

Table 7: One part of the results from the hyperparameter tuning of the LDA model.

Figure 19: visualizing the topic modelling with topic number is eight

Figure 19 shows how our data is divided into different topic groups. Most groups contain a good amount of data (over 10%), but some are smaller. It lists the ten most important words to help each group understand each other better. We can see what makes each group unique by comparing these words to the whole dataset.

4.3.3) Model Selection

MODEL	AMOUNT OF CATEGORY	SILHOUETTE	DAVIES BOULDIN	COHERENCE SCORE
K-means	11	73.79%	0.588	-
Hierarchical	9	71.09%	0.698	-
LDA	8	-	-	63%

Table 8: the evaluation findings of three clustering models (K-means, Hierarchical, and LDA)

All the clustering models perform well, producing high metrics scores. Thus, we decide to select the best model by considering the distribution of each cluster, and if which model can provide the appropriate consequence, it will be chosen as the suitable option.

Cluster	Amount	% Amount
1	8511	86.53
2	81	0.82
3	113	1.15
4	369	3.75
5	17	0.17
6	110	1.12
7	145	1.47
8	123	1.25
9	112	1.14
10	224	2.28
11	31	0.32

Cluster	Amount	% Amount
1	79	0.8
2	409	4.16
3	191	1.94
4	160	1.63
5	142	1.44
6	155	1.58
7	154	1.57
8	119	1.21
9	8427	85.68

Cluster	Amount	% Amount
1	5849	59.47
2	598	6.08
3	492	5
4	611	6.21
5	478	4.86
6	778	7.91
7	518	5.27
8	512	5.21

Table 9: the distribution of the clusters in the TikTok dataset of three clustering models (K-means, Hierarchical, and LDA)

Table 9 displays partitional clustering models separated content on one cluster in the enormous ratio of around 80% compared to LDA of roughly 60% in just only one cluster. Although this kind of model produces unexpected results, we decided to select the LDA model because it can separate content more reasonably than the other models; it means that when we apply it to analysis, we may receive the potential make-sense results to answer our research questions. We will discuss this issue in more detail in the next chapter.

4.3.4) Interpretation of Cluster Labels

This part explains the categories of determination cluster name to understand more detail in each video and analyse the insights in the [4.5].

Cluster	Word list	Category
1	funnyvideos, work, money, story, asmr, content, hate, team, recipe, friend	Entertainment
2	time, movie, music, sport, birmingham, meme, check, word, credit, math	Movie and Music
3	summer, country, home, news, quiz, everything, goat, funnymoments, stitch, thank	Travel
4	life, police, city, house, comment, storytime, america, baby, share, president	Lifestyle
5	game, humor, england, cover, interview, movieclips, book, podcast, power, greenscreen	Hobby
6	football, funny, london, song, history, guess, family, fact, medium, fire	Sport and Comedy
7	challenge, love, makeup, reaction, girl, order, help, guy, thing, kind	Social Media
8	prank, parati, dance, cute, albania, moment, motivation, scene, skit, relationship	Humour

Table 10: the determining topic name of each video cluster

Table 10 outlines the assigned topic names for each cluster. It is important to note that some clusters may encompass a broader thematic range due to the inherent challenges of clustering unlabelled data.

Each category is declared to cover the whole video of each cluster by using familiar or common words: Entertainment, Movie and Music, Travel, Lifestyle, Hobby, Sport and Comedy, Social Media, and Humour. These categories have some clusters with similar contents. For example, clusters 4 and 5 may have overlapped content types, or clusters 6 and 8 have identical funny videos.

4.4) Sentiment Analysis

In this section, we will use the VADER Sentiment Analyzer from the NLTK Python library to analyse the emotional scores of text descriptions. We will compare these scores with a training sentiment dataset to determine the appropriate sentiment content for each text: negative, neutral, or positive. Using this information, we will categorize the text description as positive or negative based on sentiment.

The compound score, calculated from the three sentiment attributes, will be used for this categorization. The compound score is a single metric that summarizes the overall sentiment of the text.

- if it is greater than 0, sentiment type is positive
- else if it is less than 0, sentiment type is negative

- else sentiment type is neutral

Figure 20: the bar chart of the number of TikTok content, showing in terms of sentiment (positive, neutral, negative)

Figure 20 illustrates that most of the content on TikTok is neutral, without explicitly using good or bad words in the descriptions. Approximately half of the total content falls into this neutral category. The percentage of positive content is significantly greater than that of negative content, with positive content making up over 30 percent, roughly three times higher than negative content. Although the TikTok community is generally good for sharing humorous and happy stories, some instances of using toxic words to describe those videos, around 10%.

In the next section, we will delve deeper into the sentiment analysis, identifying the ratio of good and bad videos type and highlighting some points that this tool produces the irrelevant data. This approach aims to foster a more satisfying and positive environment on the platform while reducing toxic activities.

4.5) Visualizing Insights

This section will show the interesting insights by using the video category and emotional feature to study the trend of TikTok social media data, it separates into three topics: the proportion of content type of each cluster, top performance video category analysis by using engagements (view, comments), the ratio of good and bad content of each cluster.

4.5.1) Analysis of Video Category Volume

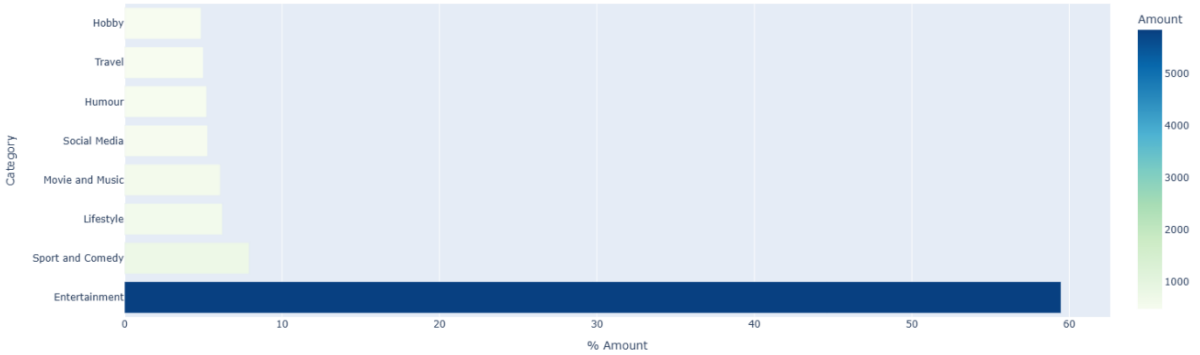


Figure 21: the percentage of the video of each cluster

Figure 21 displays that Entertainment content is the majority of this dataset, comprising over 50%, with Sport/Comedy and Lifestyle as the second and third largest categories, at approximately 6% to 7%. It indicates that many TikTok users tend to share their entertainment or hobbies, fostering a sense of community by inviting others to participate in their lives. The entertainment category may have other subcategories that may be included, but our model cannot separate the proper type, which we will discuss in the next chapter. This social media platform allows users to relax with various overall content, especially entertainment and comedy content.

4.5.2) Top Performance Contents Analysis

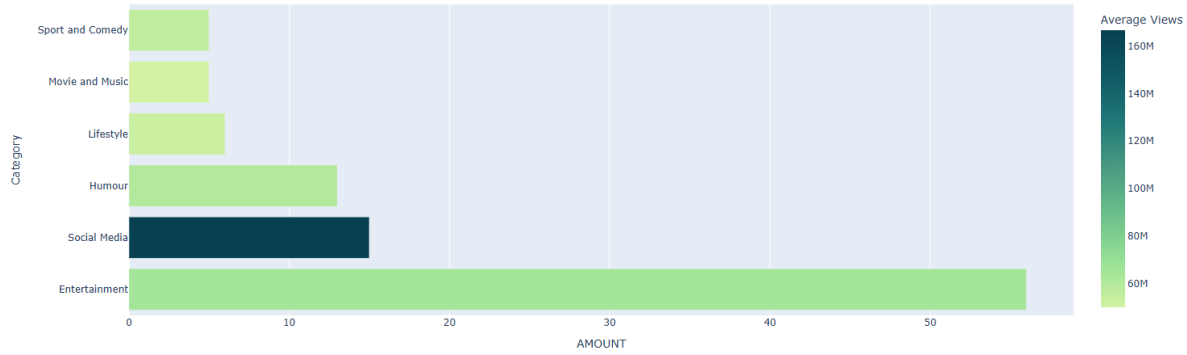


Figure 22: top hundred popular contents(views) of TikTok dataset.

Figure 22 provides a snapshot of the hundred most popular TikTok videos measured by views used in the United Kingdom area. Entertainment dominates on

this chart, occupying over 50% of the total. Still, social media has the top one in terms of the most average views, roughly 140-160 million views, compared to entertainment, which has around 80 million views.

The reason is that the number of contents in this category is the most videos; it may have a lot of unpopular content existing in this category, which makes average views at the medium level. Other content watched is similar in range to the entertainment category, with approximately 60 million views. This information supports our previous presentation that these contents are in strong demand for people who want to relax from their work or living. Many users chose to turn to TikTok to unwind after a long day by watching entertaining or relatable content.

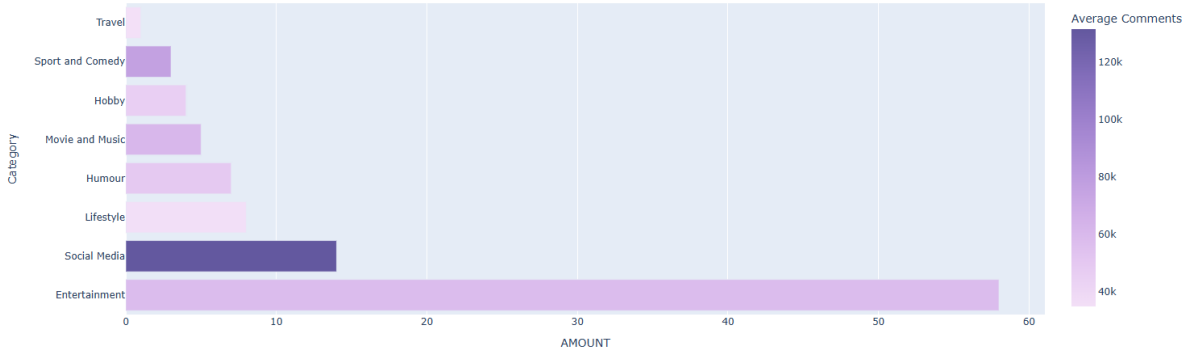


Figure 23: Top hundred engagement contents(comments) of TikTok dataset.

Regarding the engagement section, the analysis of popular videos shows that entertainment and social media are many top engagement videos, around over 50% and 10%, respectively. Both categories have the same amount of engagement in a similar direction, like the popular rate that the social media type can receive the most average comments, around 120,000 comments, compared with entertainment at around 60,000-80,000 comments. In addition, the number of humour category numbers will decrease, and travel and hobby content will increase, which means that both categories tend to have a higher participation rate than funny videos. It makes sense because travel and personal interests can be discussed or exchanged more information in many aspects, such as place, event, and specialist, than humour videos that only give happy or witty comments.

4.5.3) Sentiment Ratios within Cluster Groups

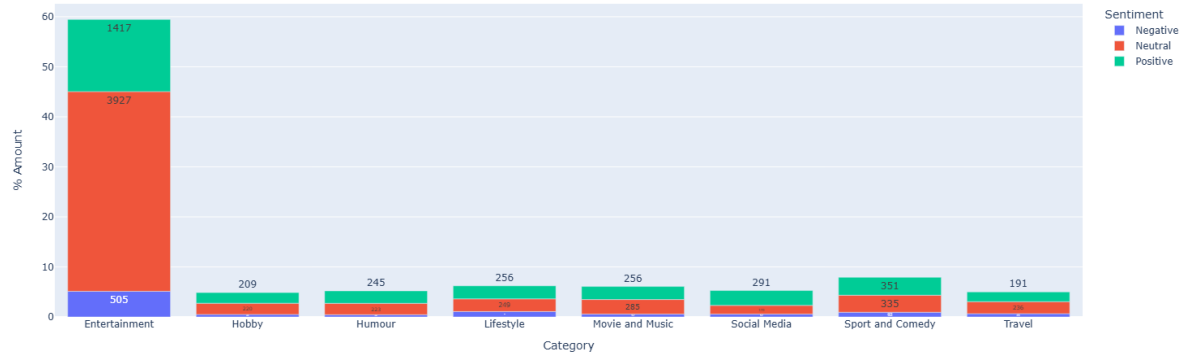


Figure 24: the percentage of the sentiment score of each cluster

Figure 24 shows the utilization of positive and negative language in the video description of each video category. It is obvious that other categories, except entertainment, have the level of using positive and neutral words closely, which means that creators tend not to bring their emotions to introduce their content, even sometimes use positive language to make audiences understand what is the main objective that content creator wants to communicate with their audience.

On the other hand, entertainment content has obviously used neutral descriptions to explain their videos that they do not put bias introducing by using emotional words, making us receive the most entertainment content not using emotionally biased-presenting descriptions, at roughly over 60% of the total. However, it has some minor videos that communicate the contents by using negative words.

It may present sad or angry stories to implore their followers and audiences to feel interested, hoping those people will watch and comment on their videos to receive more engagements and raise their content, appearing on the trending feels.

Content creators should have a significant responsibility to shape a positive online community. Avoiding using mean or negative words can create a space that more people will want to join and stay in for a long time.

In conclusion, these findings show that many TikTok users in the United Kingdom often watch entertainment, social media, lifestyle, and humour content.

Videos about social media are the most popular category, with the most average views from the audiences. Although these categories generally receive positive feedback, some videos use mean or negative language. The text clustering model and sentiment analysis tools can offer insights, but they also have limitations, which are described more in the next chapter.

Chapter 5: Discussion and Analysis

This chapter reveals a detailed discussion and analysis of this study from many perspectives by covering five main topics. First, it summarizes the key results, underscoring how they are significant. Second, it interprets these results in more detail to explain why they are essential. Next, it compares the findings with the previous study, identifies similarities and differences, and discusses any unexpected outcomes during the research. Then, we will explore the study's implications, the challenges we found, and what we contribute that can be applied in the real scenario. Finally, it outlines the limitations and proposes future suggestions and potential areas for further investigation.

5.1 Summarize Key Findings

Regarding our research question, we want to explore four main things: build a tool to organize the video categories by considering video description only, prove that sentiment analysis tools can separate the unappropriated content from the appropriated content, understand the central theme of TikTok data, and find the popular content in this online social media.

Our results in Chapter 4 show that topic modelling (LDA) is a suitable model that can divide the video group into clusters appropriately. Additionally, we can apply this tool to provide the video category to produce more insights continuously.

Then, we build the bar charts to help readers understand the information quickly. They highlight the primary theme in TikTok's online social media by showing the proportion of each category and then displaying the top one hundred popular content to see the hottest content type that will be considered for understanding the current trending content. This can benefit the business section, which is explained more in [5.3].

Although these findings can provide many potential benefits for improving online business by using TikTok to understand customers(audience), some issues occur, which are discussed in more detail in the next section.

5.2 Interpretation and Contextualization

We describe the significant meaning of our findings to indicate how they address our research questions. Then, we compare the results with those of previous related studies to emphasize the agreements and discrepancies. Eventually, we will present any unexpected outcomes and propose potential hypotheses for solving these results.

5.2.1) Interpret Results

Before explaining the results from the machine learning clustering model, we will explain why the PCA tool works well with our experiment. This tool can provide many benefits. For example, we can improve the model performance by eliminating non-sense or unmeaningful words to extract more relevant data.

In addition, this method can save computation costs by reducing the dimensionality of textual information. Before, without this tool, we spent a lot of time running the clustering model like K-means and Hierarchical model, around 30 minutes, as did the hyperparameter tuning process, inputting textual dataset size of around ten thousand videos using 18 different k-values; when we applied this method helped speed up running time around one to two minutes when compared it can save a huge computational cost, roughly 30 times and receive the remarkable. However, the computational time depends on the number of components used in PCA, for which we selected a small number of components at seven, as shown in [4.3.1.b].

From our results, LDA topic modelling can provide the category of each video. These findings are used to analyse the main theme of the TikTok dataset in the next step. It can only address the first question of classifying unlabelled videos using text descriptions. For the second question, we conclude that we cannot separate unappropriated content by applying sentiment analysis to measure only the positive and negative video descriptions.

In both the third and fourth questions, we can understand the primary theme of TikTok social use by doing the thematic analysis to see the proportion of each category and a deeper analysis of more insights about the hottest category to explore the current popular videos and trending contents, which we found the entertainment is the most category in TikTok and content about the social media is the most popular contents that are received the highest participate(view, comment).

These consequences still have some problems and lack of completion, which we will discuss in section [5.4], which focuses on explaining the limitations of our research

5.2.2) Compare with Existing Literature

We have taken inspiration from Bakar's (2024) research studying TikTok clustering. This section will present a comparison of similar and different findings. We received more topics than her for our results, and we have identical categories, such as music and movies. This means TikTok platforms have the most entertainment content in Asia and the United Kingdom.

Bakar built the model to predict video type using a few new video description samples in contrast with our results that applied to our prior dataset, a familiar one. We agree that machine learning, like the LDA model, can classify video description categories well. However, her research gave a few limitations of this machine learning and text analysis with social media, where we found hidden issues with our model even though it can provide a good evaluation score like her model's performance, which it needed to explain. This point will be discussed in the next topic.

5.2.3) Discuss Unexpected Results

We found surprising findings, such as the distribution of each category with many proportions in entertainment categories. Some videos may also need to be organized in the proper category. We assume that the reason our result is still not good enough is that we have not implemented the many filtering conditions of the data collection process that can detect more irrelevant data, and unexpected results, such as contents from other languages (Vietnam, Arabic, Chinese).

Our findings have a direct impact on the overall context of our research. We declare the common category words, hoping to cover the overall contexts, but it cannot cover the unknown meaning of other languages and some unmeaningful words. These are the unexpected findings that we found, and we will explain in more detail about the limitations of our work in section [5.4]

5.3 Implications of the Study

This part describes our primary challenges in this study and how our findings can contribute to online business. It can be applied in real-world situations.

The main challenge is similar to the problem that Ramamonjisoa (2014) found enormous spam and troll text description information that affects our text preprocessing, text clustering, and clustering interpretation processes. We tried

many times to handle them, but this information is still in our cleaning dataset, which affects our model performance and results. Thus, although we can eliminate all unnecessary information, we cannot know the real video category because our dataset did not provide the original video type for checking the accuracy of our model; it is one of the unsupervised limitations that we can explore whether these results are appropriate or not only.

Another challenge is doing the experiment with various parameters to find the optimal value; the research period is short, so we must scope the plan strictly. However, even though we handle many processes to reduce the computational cost, sometimes, when we train the machine learning models, it takes a long time (many hours) to get the findings because we must try with various parameters to take the suitable parameter to build the final model to produce the outcomes for analysing in the next step. These challenges are one part that we spend the most time exploring and implementing.

In terms of contributions, our results can propose the steps to handle textual data. They contribute the concepts and methods to work with this kind of data type, which are applied to work with any textual information in future work. Depending on the study's expectations, some processes may need to add a few more steps to reduce unnecessary data.

One of our main contributions is applications that can produce the category label by considering the text description. These findings can be applied to categories of the video type and give some ideas to understand the audiences to help people or companies that use the TikTok application to promote their online business, especially marketing strategies or decision-making, such as understanding the most of customers to produce the relevant contents and making ads in the popular category to receive the potential of enormous users' engagement by TikTok had 1.5 billion monthly active users in 2023 (Iqbal, 2024). If they can propose suitable services for each level of customer, they can raise more profit and make their brands more popular.

Furthermore, our clustering model can be applied to classify the new social media dataset (text datatype) to explore the overall category of each social media approximately. It helps users understand the overall theme of their social media information by analysing the proportion of each type of social media or exploring the trending or unhidden patterns of their data to produce insights to support their business decisions in the future. In the next part, we will present the limitations that we found in the study.

5.4 Study Limitations

We found two main constraints in this research: separate inappropriate content and textual data manipulation. First, when we applied sentiment analysis to measure the emotional score of each video, we expected this tool to help find appropriate content. Still, it cannot confirm whether outcomes are suitable, such as whether videos about self-sacrifice or sad stories are measured as inappropriate content because we only consider video descriptions.

They may be good content that uses drama words to introduce their content and attract audiences. It affects our findings and cannot be used to decide whether any content is good or bad by using sentiment analysis tools; it should have many specific methods for analysing more details and organizing the appropriate content in future studies.

The other main limitation is the issue of data manipulation for textual datasets. It depends on the quality of the datasets and the processes used to collect data. We found that the problem was that our data needed more accurate and relevant words, but we received a lot of irrelevant text information. Despite trying to filter out non-English videos and single and double-character words in the video description as much as possible, we cannot categorize many unmeaningful video descriptions that we cannot categorize.

The challenge of determining category names is significant. It underscores the complexity of the task and the need for more advanced methods to produce accurate results from our clustering model. These limitations were found in the experiments to see the results. In the next section, we propose potential ideas for developing this relevant kind of research in the future.

5.5 Future Research Directions

Based on our findings, we propose some advice on investigating a further research section, underscoring gaps that still need to be addressed and potential research questions that could be explored in future studies.

5.5.1) Suggestions for Future Research

- Gathering more information: It can help researchers if they have a large textual dataset to build the corpus as a dictionary for training the text machine learning models to find more hidden patterns to expand more analysis sections to explore and give more insights.

- Add more conditions to filter spam and irrelevant content: This can reduce the clustering application's work in computing unnecessary things and produce unexpected findings. If we can define more conditions to support the filter process to eliminate unnecessary information, we can receive better results and easy analysis.
- Investigation of distance metric's findings: developing the visualization processes and exploring the similarity or distance metrics of each sample to capture the textual data nuances to enhance the accuracy and reasonable of clustering outcomes
- Apply complex tools: using advanced machine learning, such as deep learning, to build more accurate and sophisticated text clustering models that can understand more complicated patterns in a textual dataset.

5.5.2) Potential Research Question

A future topic that should be addressed is comment analysis, one of the valuable techniques that classifies the audience's emotional reactions more precisely, which our study cannot follow in this section. Understanding which content may be trending and that many people agree or disagree with a deeper analysis of what they think about those videos is crucial. It can be tracked to improve the service for producing new videos to serve audiences appropriately. Moreover, it can detect trends, identify frequent discussions in the dataset, and analyse emerging trends.

Chapter 6: Conclusion

This research objective is to study the main theme of TikTok social media by applying text mining and text clustering concepts to help categorize the video type to understand the TikTok community, which has most of the content about entertainment, social media, and comedy short videos. We propose the differences between the previous studies in terms of thematic analysis and study sentiment analysis together to explore more hidden patterns of social media datasets to understand surprising insights. Moreover, our findings provide LDA topic modelling by evaluating with a coherence score = 0.632. This metric shows that the model is well-organized and can be applied to produce the video categories to analyse the

current trending video. These findings can benefit future social media analysis since fundamental processes work with large textual datasets into sentiment analysis to produce insights to underscore the trending contents and primary theme of the community that can be used to support the decision-making of online marketing campaigns and enhance the user experience in the future. However, we found the challenges with text preprocessing and clustering interpretation that have many spam and uncategorized words that affect our results, which worried us when we determined the topic name. We found some videos that were categorized as unreasonable. When we applied the sentiment analysis, we concluded that we could not separate the inappropriate content and only used this tool with video description; overall, we were satisfied with the results, which can help us to understand the primary theme of the TikTok dataset. Thus, these are the limitations that we struggled with, which for sentiment analysis should be explored more with comment information instead of opinion analysis, and the data collection should be scoped narrower to filter unnecessary information to extract more relevant data to improve the efficiency of the outcomes in the further studies.

References

1. Aggarwal, C.C. and Zhai, C., (2012). *Mining Text Data* [Online]. Boston, MA: Springer US. Available from: https://doi.org/10.1007/978-1-4614-3223-4 [Accessed 2 July 2024]
2. Bakar, J.A., Nur, Mohd, Azmi, N.S., Harun, N.H., Awang, H., and Nur (2024). TikTok Video Cluster Analysis Based on Trending Topic. *Communications in computer and information science*, pp.193–205.
3. Beel, J., Gipp, B., Langer, S. and Breitinger, C., (2015). Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries*. 17(4), pp. 305-338. Available from: https://doi.org/10.1007/s00799-015-0156-0.
4. Belkin, N.J., and Croft, W.B., (1992). Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12), pp. 29–38.
5. Blei, D.M., Ng, A.Y., and Jordan, M.I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3, pp. 993–1022.
6. Ceci, L. (2024). TikTok audience reach in the United Kingdom (UK) in March 2024, by age group. [Online] *Statista*. Statista Inc. Available from: https://www.statista.com/statistics/1464776/uk-tiktok-use-by-age/ [Accessed 2 July 2024].
7. Canvnar, W., and Trenkle, J.M., (2001). N-Gram-Based Text Categorization. *Environmental Research Institute of Michigan*, 134001, pp. 48113-4001.
8. Davies, D.L., and Bouldin, D.W., (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2), pp. 224-227.
9. Dwivedi, D.N., and Pathak, S. (2022). Sentiment Analysis for COVID Vaccinations Using Twitter: Text Clustering of Positive and Negative Sentiments. *International Series in Operations Research & Management Science*. Cham: Springer, 320, pp.195-203.
10. El Atawy, S. and Abd ElGhany, A., (2018). Automatic spelling correction based on n-gram model. *International journal of computer applications*, 182(11), pp.0975–8887.
11. Feldman, R. and Dagan, I., (1995). *Knowledge Discovery in Textual Databases (KDT)*. KDD, vol. 95, pp.112.
12. Feldman, R. and Sanger, J. (2006). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. [Online] Cambridge University Press. Cambridge: Cambridge University Press. Available from: https://www.cambridge.org/core/books/text-mining-handbook/0634B1DF14259CB43FCCF28972AE4382 [Accessed 2 July 2023].

13. Fox, C., (1989). A stop list for general text. *ACM SIGIR Forum*, 24(1–2), pp.19–21.
14. Hamborg, F., and Donnay, K., (2021). NewsMTSC: A Dataset for (Multi-)Target-dependent Sentiment Classification in Political News Articles. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*: Main Volume, pp. 1663-1675.
15. Hartmann, J., Huppertz, J., Schamp, C. and Heitmann, M. (2019). Comparing automated text classification methods. *International Journal of Research in Marketing* [Online], 36(1), pp.20–38. Available from: https://doi.org/10.1016/j.ijresmar.2018.09.009 [Accessed 1 July 2024].
16. He, W., Zha, S. and Li, L. (2013). Social media competitive analysis and text mining: A case study in the pizza industry. *International Journal of Information Management*, 33(3), pp.464–472. Available from: https://doi.org/10.1016/j.ijinfomgt.2013.01.001 [Accessed 4 July 2024].
17. Hirschberg, J. and Manning, C.D. (2015). *Advances in natural language processing. Science,* [Online] 349(6245), pp.261–266. Available from: https://doi.org/10.1126/science.aaa8685 [Accessed 2 July 2024].
18. Iqbal, M. (2024). *TikTok Revenue and Usage Statistics (2024)*. [online] Business of Apps. Available from: https://www.businessofapps.com/data/tik-tok-statistics/.
19. Isah, H., Trundle, P., and Neagu, D., (2014). Social media analysis for product safety using text mining and sentiment analysis. *2014 14th UK Workshop on Computational Intelligence (UKCI)*, Bradford, UK, pp. 1-7.
20. Jain, A.K., Murty, M.N., and Flynn, P.J. (1999). Data clustering: a review. *ACM computing surveys* (CSUR), 31(3), pp. 264-323.
21. Johnson, S.C., (1967). Hierarchical clustering schemes. *Psychometrika*, 32, pp. 241–254. Available from: https://doi.org/10.1007/BF02289588 [Accessed by 8 July 2024].
22. Jones, K.S., (1972). A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *Journal of Documentation*, 28(1), pp. 11-21.
23. Leskovec, J., Rajaraman, A. and Ullman, J., (2011). *Mining of Massive Datasets.* pp. 8.
24. Liu, B. (2012). *Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies*. Springer Cham.
25. Lloyd, S. P., (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2), pp. 129-137.
26. Luhn, H.P., (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4), pp.309–317.
27. MacQueen, J. B., (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on*

mathematical statistics and probability. California: University of California Press, 1, pp. 281–297.
28. Mann, Kaur, J., (2021). *Semantic Topic Modeling and Trend Analysis*. Thesis (M.S.). Linköping University, Sweden.
29. Manning, C.D., Raghavan, P. and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge: Cambridge University Press.
30. Mimno, D., Wallach, H.M., Talley, E., Leenders, M., and McCallum, A. (2011). Optimizing Semantic Coherence in Topic Models. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pp. 262-272.
31. Miner, G., Delen D., Elder, J., Fast, A., Hill, T., and Nisbet, R.A., (2012). *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*. Academic Press.
32. Mitchell, R. (2018). *Web scraping with Python: Collecting data from the modern web*. Sebastopol, CA: O'Reilly Media.
33. Monshizadeh, M., Khatri, V., Kantola, R., and Yan, Z., (2022). A deep density based and self-determining clustering approach to label unknown traffic. *Journal of Network and Computer Applications*, 207, pp. 103513.
34. Natalie, M. and Gabriel, W. (2020). *The Virus of Hate: in Terrorism Right - Far Cyberspace*. [Online] Available from: https://voxpol.eu/wp-content/uploads/filebase/report/Dark20Hate.pdf [Accessed 2 Jul. 2024].
35. Pak, A., and Paroubek, P., (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of the International Conference on Language Resources and Evaluation*, 17-23 May 2010, Valletta, Malta: LREC, pp.1320-1326.
36. Phan, V.H., Ninh, D.K. and Ninh, C.K., (2020). An effective vector representation of Facebook fan pages and its applications. In: M. Hernes, K. Wojtkiewicz, and E. Szczerbicki, eds. *Proceedings of the 12th International Conference on Computational Collective Intelligence (ICCCI 2020)*, 27-29 September 2020, Da Nang, Vietnam. Cham: Springer, pp. 674–685.
37. Porter, M.F., (1980). *An algorithm for suffix stripping*. Program, 14(3), pp.130–137.
38. Porter, M.F. (2006), *Stemming algorithms for various European languages*. Available from: www.snowball.tartarus.org/texts/stemmersoverview.html [Accessed 5 July 2024].
39. Prihatini, P.M., Suryawan, I.K. and Mandia, I.N., (2018). Feature extraction for document text using Latent Dirichlet Allocation. *Proceedings of the IOP Conference Series: Materials Science and Engineering*, 5-7 March, 2018, London, UK. Bristol: IOP Publishing, 953, pp. 012047.
40. Pritchard, J.K., Stephens, M., and Donnelly, P., (2000). Inference of population structure using multilocus genotype data. *Genetics*. 155(2): pp. 945-959.

41. Pudil, P., and Novovičová, J., (1998). Novel Methods for Feature Subset Selection with Respect to Problem Knowledge. In Liu, Huan; Motoda, Hiroshi (eds.). Feature Extraction, Construction and Selection. pp. 101.
42. Rousseeuw, P.J., (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J*ournal of Computational and Applied Mathematics*, 20, pp. 53-65.
43. Siersdorfer, S., Chelaru, S., Nejdl, W., and San Pedro, J., (2010). How useful are your comments? analyzing and predicting youtube comments and comment ratings. *Proceedings of the 19th International Conference on World Wide Web,* 26-30 April 2010, Raleigh, North Carolina, USA. New York: ACM, pp. 891-900.
44. Steinhaus, H., (1956): Sur la division des corps matériels en parties. *Bulletin de l'Académie Polonaise des Sciences, Classe III*, 4(12), pp. 801-804.
45. Stieglitz, S., Mirbabaie, M., Ross, B. and Neuberger, C. (2018). Social media analytics – Challenges in topic discovery, data collection, and data preparation. *International Journal of Information Management* [Online], 39(39), pp.156–168. Available from: https://doi.org/10.1016/j.ijinfomgt.2017.12.002 [Accessed 2 July 2024].
46. Sulayes, A.R., (2017). Reducing Vector Space Dimensionality in Automatic Classification for Authorship Attribution. *Revista Ingeniería Electrónica, Automática y Comunicaciones*, 38(3), pp. 26-35.
47. Vogels, E.A., Gelles-Watnick, R. and Massarat, N. (2022). Teens, Social Media and Technology 2022. [Online] *Pew Research Center*. Available from: https://www.pewresearch.org/internet/2022/08/10/teens-social-media-and-technology-2022/ [Accessed 2 July 2024].
48. Verma, T., Duggal, R., and Gaur, D. (2014) Tokenization and Filtering Process in RapidMiner. *International Journal of Applied Information Systems (IJAIS),* 7(2), pp. 16-18. Available from: 10.5120/ijais14-451139 [Accessed 7 August 2024].

Appendix

This list of spam words below:

Words				
viral	trend	xuhuong	video	fyp
foryou	fypviral	fypage	tiktok	fypforyou
fouryou	fypppppppppppppppppppppppp	viralvideo	foryourpage	pov
trending	capcut	viraltiktok	new	сериал
bio	link	foryoupageofficial	fypp	xuhuongtiktok
xybca	fordig	youtube	name	edit
edits	xyzbca	know	part	back

JavaScript code for scrapping TikTok dataset:

app.js

```
import puppeteer from "puppeteer-extra";
import StealthPlugin from "puppeteer-extra-plugin-stealth";
import * as readline from "readline";
import * as fs from "fs";

async function cin(prompt) {
  const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout,
  });

  return new Promise((resolve) => {
    rl.question(prompt, (answer) => {
      rl.close();
      resolve(answer);
    });
  });
}

(async () => {
  puppeteer.use(StealthPlugin());
  const browser = await puppeteer.launch({
    executablePath:
      "/Applications/Google Chrome.app/Contents/MacOS/Google Chrome",
    headless: false,
    defaultViewport: null,
    args: [
      "--window-size=1920,1080",
      "--no-sandbox",
      "--disable-setuid-sandbox, --auto-open-devtools-for-tabs",
      "--disable-dev-shm-usage",
    ],
  });
```

```
  try {
    const page = (await browser.pages())[0];
    let i = 0;
    await page.goto("https://www.tiktok.com");
    let obj = {
      itemList: [],
    };

    // Function to read existing data from data.json
    const readData = async () => {
      return new Promise((resolve, reject) => {
        fs.readFile("data24.json", "utf8", (err, data) => {
          if (err) {
            if (err.code === "ENOENT") {
              // File does not exist, return empty array
              resolve({ itemList: [] });
            } else {
              reject(err);
            }
          } else {
            resolve(JSON.parse(data));
          }
        });
      });
    };

    await cin("Enter to loop...");

    await page.on("response", async (response) => {
      if (response.url().includes("www.tiktok.com/api/recommend/item_list/")) {
        const data = await response.json();
        i++;
        console.log(i);
        if (data && data?.itemList) {
          // Read existing data from the file
          let existingData = await readData();
          obj.itemList = existingData.itemList;

          // Append new data to existing data
          obj.itemList.push(...data.itemList);
        }

        const json = JSON.stringify(obj);
        fs.writeFile("data24.json", json, "utf8", (err) => {
          if (err) {
            console.error(err);
            return;
          }
          console.log("New data has been added successfully.");
        });
      }
    });

    while (obj.itemList.length <= 2000) {
      await page.keyboard.press("End");
      await new Promise((resolve) => setTimeout(resolve, 5000));
    }

    await page.removeAllListeners();
    console.log("done");
    process.exit()
  } catch (error) {
    await browser.close();
    throw error;
  }
})();
```

package.json

```
{
  ▷ Debug
  "scripts": {
    "start": "ts-node app.ts"
  },
  "dependencies": {
    "cookie-parser": "^1.4.6",
    "cors": "^2.8.5",
    "express": "^4.19.2",
    "node-fetch": "^3.3.2",
    "puppeteer": "^22.12.0",
    "puppeteer-extra": "^3.3.6",
    "puppeteer-extra-plugin-stealth": "^2.11.2",
    "xlsx": "^0.18.5"
  },
  "type": "module"
}
```

convertJsonToExcel.js

```
import fs from "fs";
import { numStr, readExcelFile, writeExcelFile } from "./helper.js";

try {
  let existingDf = readExcelFile("./tiktok_df.xlsx");

  for (let i = 1; i <= 24; i++) {
    const data = fs.readFileSync(`./data${numStr(i)}.json`, "utf8");
    const jsonData = JSON.parse(data);
    const df = jsonData["itemList"];

    const excelDf = df.map((data) => ({
      id: data["id"],
      description: data["desc"],
      collectCount: data["stats"]["collectCount"],
      commentCount: data["stats"]["commentCount"],
      likeCount: data["stats"]["diggCount"],
      playCount: data["stats"]["playCount"],
      shareCount: data["stats"]["shareCount"],
      duration: data["video"]["duration"],
      quality: data["video"]["videoQuality"],
      author: data["author"]["nickname"],
    }));

    // update data
    existingDf = existingDf.concat(excelDf);
    writeExcelFile(existingDf, "tiktok_df.xlsx");
  }
} catch (err) {
  console.error("Error reading or parsing file:", err);
}
```

Python codes of this research is shown below:

```
# Download the other packages
!pip install -U kaleido
!pip install pyLDAvis

# Data analysis
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.special import jv
from mpl_toolkits.mplot3d import Axes3D
import seaborn as sns
from scipy import stats
from pprint import pprint
from wordcloud import WordCloud
import warnings
# Text analysis
import re
import nltk
from nltk.corpus import stopwords
from nltk.corpus import words
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
# TF-IDF Vectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from collections import Counter
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import PCA
# Clustering models
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster
from sklearn.metrics import silhouette_score
from sklearn.metrics import davies_bouldin_score
from scipy.spatial.distance import cdist
# LDA topica modelling
import gensim
from gensim.utils import simple_preprocess
import tqdm
from gensim.models import CoherenceModel
import spacy
from gensim import corpora
from gensim.models import LdaModel
import pyLDAvis.gensim
```

```
import pickle
import pyLDAvis
# Sentiment Analysis
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
import plotly.express as px

# Download text mining packages
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('words')
nltk.download('averaged_perceptron_tagger')
nltk.download('vader_lexicon')
warnings.filterwarnings('ignore')

stopwords = stopwords.words('english')
stopwords.extend(['from', 'subject', 're', 'edu', 'use'])

"""# Import dataset"""

from google.colab import files
uploaded = files.upload()

df = pd.read_excel('./tiktok_df.xlsx')
# Select only unique data
selected_df = df.drop_duplicates(subset='id')
selected_df = selected_df.drop_duplicates(subset='description')
selected_df = selected_df[selected_df['description'].notnull() & (selected_df['description'] != '')]
selected_df = selected_df.reset_index(drop=True)

"""# Text preprocessing

## Helper function
"""

# Stemming and Lemmatization
stemmer = PorterStemmer()
lemmatizer = WordNetLemmatizer()
# Create a set of English words
english_words = set(words.words())
english_words = {item.lower() for item in english_words}
spam_words = ['viral', 'trend', 'xuhuong', 'video', 'fyp', 'foryou', 'foryoupage', 'fypviral', 'fypage', 'tiktok', 'fypforyou', 'fouryou', 'fypppppppppppppppppppppp', 'viralvideo',
              'foryourpage', 'pov', 'trending', 'capcut', 'viraltiktok', 'new', 'сериал', 'bio', 'link', 'foryoupageofficial', 'fypp', 'xuhuongtiktok', 'xybca', 'fordig', 'youtube', 'name', 'edit', 'edits']
```

```
nlp = spacy.load("en_core_web_sm", disable=['parser', 'ner'])

def filtering(description):
    def is_valid_word(word):
        return word.lower() in english_words
    words = nltk.word_tokenize(description)
    words = [w for w in words if w not in stopwords and w not in spam_words] # filter spam words
    # Apply lemmatization technique
    processed_words =[lemmatizer.lemmatize(word) for word in words]
    # Filter out non-English words
    filtered_words = [word for word in processed_words if is_valid_word(word)]
    return ' '.join(processed_words)

def remove_emoji(text):
    emoji_pattern = re.compile("["
                               u"\U0001F600-\U0001F64F"  # emoticons
                               u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                               u"\U0001F680-\U0001F6FF"  # transport & map symbols
                               u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                               u"\U00002702-\U000027B0"  # dingbats
                               u"\U000024C2-\U0001F251"  # enclosed characters
                               u"\U0001F900-\U0001F9FF"  # Supplemental Symbols and Pictographs
                               u"\U0001FA70-\U0001FAFF"  # Symbols and Pictographs Extended-A
                               u"\U0001F700-\U0001F77F"  # Alchemical Symbols
                               u"\U0001F780-\U0001F7FF"  # Geometric Shapes Extended
                               u"\U0001F800-\U0001F8FF"  # Supplemental Arrows-C
                               u"\U0001FA00-\U0001FA6F"  # Chess Symbols
                               u"\U00002700-\U000027BF"  # Dingbats
                               u"\U0001F1E6-\U0001F1FF"  # Regional Indicator Symbols
                               u"\U0001F004"             # Mahjong Tile Red Dragon
                               u"\U0001F0CF"             # Playing Card Black Joker
                              "]+", flags=re.UNICODE)
    return emoji_pattern.sub(r'', text)

def text_cleaning_custom(text):
    # Tokenize the text into words
    word_list = re.findall(r'\b\w+\b', text)
    # Filter out single-double-triple characters words
    filtered_words = [word for word in word_list if len(word) > 3]
    # Join the filtered words back into a single string
    filtered_text = ' '.join(filtered_words)

    return filtered_text
```

```python
def text_preprocess (df):
    # Remove punctuations from the Description column
    punctuations = '''()-[]{};:"\,<>./?@#$%^&*£!~ '''
    text_list = []
    for text in df['description']:
        sentence = ""
        for char in text:
            if (char not in punctuations):
                sentence = sentence + char
            else:
                sentence = sentence + " "
        # Removing number
        pattern = r'\d+'
        sentence = re.sub(pattern, '', sentence)

        sentence = remove_emoji(sentence)
        sentence = text_cleaning_custom(sentence)

        # Mutuating the text cleaning column as sentence
        text_list.append(sentence.lower())
    # Apply the function to the Description column and create the Tokens column
    df['token'] = text_list
    df['sentence'] = df['token'].apply(filtering)

def calculate_confidence_interval_95(dataframe, column):
    mean = dataframe[column].mean()
    sem = stats.sem(dataframe[column])

    ci = stats.t.interval(0.95, len(dataframe[column])-1, loc=mean, scale=sem)
    return ci

# Unigram
def get_top_n_words(corpus, n=10):
    vec = CountVectorizer(stop_words='english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq = sorted(words_freq, key=lambda x: x[1], reverse=True)
    return words_freq[:n]

# Bigram
def get_top_n_bigrams(corpus, n=10):
    vec = CountVectorizer(ngram_range=(2, 2), stop_words='english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq = sorted(words_freq, key=lambda x: x[1], reverse=True)
    return words_freq[:n]

def plot_horizontal_bar_chart(data, title):
    words, freqs = zip(*data)
    plt.figure(figsize=(10, 6))
    plt.barh(words, freqs, color='#5ca7f7')
    plt.xlabel('Frequency')
    plt.ylabel('Words')
    plt.gca().invert_yaxis()  # Invert y-axis to have the highest frequency on top
    plt.title(title)
    plt.show()

def print_top_terms_per_k_mean_cluster(tfidf_df, terms, num_terms=10):
    cluster_centers = kmeans.cluster_centers_
    original_centroids = pca.inverse_transform(cluster_centers)  # Transform centroids back to original feature space
    terms = vectorizer.get_feature_names_out()

    for i, centroid in enumerate(original_centroids):
        top_indices = centroid.argsort()[-num_terms:][::-1]  # Get indices of top terms
        top_terms = [terms[idx] for idx in top_indices]
        print(f"Cluster {i + 1}:")
        print(", ".join(top_terms))
        print("\n")

# Hierarchical
def compute_wss_hierarchical(data, labels):
    wss = 0
    for label in np.unique(labels):
        cluster_data = data[labels == label]
        centroid = cluster_data.mean(axis=0)
        wss += ((cluster_data - centroid) ** 2).sum()
    return wss

def print_top_terms_per_hierarchical_cluster(tfidf_df, terms, num_terms=10):
    cluster_terms = {}
    for cluster in range(1, max_clusters + 1):
        cluster_data = tfidf_df[tfidf_df['cluster'] == cluster].drop(columns=['cluster'])
```

```
        cluster_mean = cluster_data.mean(axis=0)
        top_terms = cluster_mean.sort_values(ascending=False).head(num_terms).index.tolist()
        cluster_terms[cluster] = top_terms
        print(f"Cluster {cluster}:")
        print(", ".join(top_terms))
        print("\n")
    return cluster_terms

def tokenize_LDA_model(sentences):
    for sentence in sentences:
        # deacc=True removes punctuations
        yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))


def make_bigrams(texts, bigram_mod):
    return [bigram_mod[doc] for doc in texts]

def make_trigrams(texts, trigram_mod, bigram_mod):
    return [trigram_mod[bigram_mod[doc]] for doc in texts]

def lemmatization(texts, allowed_postags=['NOUN']):
    """https://spacy.io/api/annotation"""
    texts_out = []
    for sent in texts:
        doc = nlp(" ".join(sent))
        texts_out.append([token.lemma_ for token in doc if token.pos_ in allowed_postags])
    return texts_out

def compute_coherence_values(corpus, dictionary, text, k, a, b):
    lda_model = gensim.models.LdaMulticore(corpus=corpus,
                                           id2word=dictionary,
                                           num_topics=k,
                                           random_state=100,
                                           chunksize=100,
                                           passes=5,
                                           alpha=a,
                                           eta=b)

    coherence_model_lda = CoherenceModel(model=lda_model, texts=text, dictionary=id2word, coherence='c_v')

    return coherence_model_lda.get_coherence()
```

```
def get_dominant_topic(lda_model, corpus):
    dominant_topics = []
    for doc_bow in corpus:
        doc_topics = lda_model.get_document_topics(doc_bow)
        dominant_topic = max(doc_topics, key=lambda x: x[1])[0]
        dominant_topics.append(dominant_topic + 1)
    return dominant_topics

# Sentiment analysis
def classify_sentiment(compound):
    if compound > 0:
      return "Positive"
    elif compound < 0:
      return "Negative"
    else:
      return "Neutral"

# Visualization insights
def convert_num_cluster_to_text(num):
  match num:
    case 1:
      return "Entertainment"
    case 2:
      return "Movie and Music"
    case 3:
      return "Travel"
    case 4:
      return "Lifestyle"
    case 5:
      return "Hobby"
    case 6:
      return "Sport and Comedy"
    case 7:
      return "Social Media"
    case 8:
      return "Humour"

def calculate_category_distribution(df, total_count, attribute):
  result = df.groupby(attribute).size().reset_index(name='Amount')
  result['% Amount'] = (result['Amount'] / total_count) * 100
  result['% Amount'] = result['% Amount'].round(2)
  result = result.sort_values(by='Amount', ascending=False)
  return result
```

```
313
314 def find_popular_distribution(df, attribute, label_attribute, n):
315   top_100 = df.sort_values(attribute, ascending=False).head(n)
316   result_df = top_100.groupby('Category').agg({attribute: ['count', 'mean']}).reset_index()
317   result_df.columns = ['Category', 'AMOUNT', label_attribute]
318   result_df = result_df.sort_values('AMOUNT', ascending=False)
319
320   return result_df
321
322 muslim_keywords = [
323     "allah", "muslimtiktok", "اسلام", "مسلم", "اسلامي", "قرآن",
324     "حديث", "الحمدلله", "الله", "الدين", "محمد", "يوزير", "الدعاء",
325     "مسجد", "الحجاب", "الصلاة", "الصوم", "الجهاد", "رمضان",
326     "مكة", "المدينة", "الشهادة", "الزكاة", "الإسلام", "القرآن",
327     "الدعوة", "قريبة", "أمة", "خليفة", "الفتوى", "الحديث",
328     "عمرة", "حج", "الحلال", "الحرام", "النبي", "الصحابة", "الجنة",
329     "جهنم", "يوم_القيامة", "فهيد", "الخلافة", "المسلمون"
330 ]
331
332 selected_df = selected_df[~selected_df['description'].apply(lambda x: any(word in x.lower() for word in muslim_keywords))]
333 text_preprocess(selected_df)
334 selected_df = selected_df[selected_df['sentence'].notnull() & (selected_df['sentence'] != '')]
335
336 """# Exploratory Data Analysis"""
337
338 print(selected_df.shape)
339 print(calculate_confidence_interval_95(selected_df, 'playCount'))
340 print(calculate_confidence_interval_95(selected_df, 'likeCount'))
341 print(calculate_confidence_interval_95(selected_df, 'commentCount'))
342 print(calculate_confidence_interval_95(selected_df, 'collectCount'))
343 print(calculate_confidence_interval_95(selected_df, 'shareCount'))
344
345 # Statistical summary of the dataset
346 selected_df.describe().T
347
348 num_cols = selected_df.select_dtypes(include=np.number).columns.tolist()
349 #num_cols.remove(['id'])
350
351 for col in num_cols:
352   plt.figure(figsize = (15,4))
353   plt.subplot(1,2,1)
354   selected_df[col].hist(grid=False)
355   plt.ylabel('Count')
356   plt.subplot(1,2,2)
357   sns.boxplot(x=df[col])
358   plt.show()
359
360 """# Word Cloud Visualization"""
361
362 long_string_without_cleaning = ','.join(list(selected_df['description'].values))
363 wordcloud = WordCloud(background_color= "white",
364                       max_words= 5000, contour_width= 3,
365                       contour_color= 'steelblue')
366
367 wordcloud.generate(long_string_without_cleaning)
368 # save word clound to jpg image
369 wordcloud.to_file("wordcloud_v1.jpg")
370
371 long_string = ','.join(list(selected_df['sentence'].values))
372
373 wordcloud = WordCloud(background_color= "white",
374                       max_words= 5000, contour_width= 3,
375                       contour_color= 'steelblue')
376
377 wordcloud.generate(long_string)
378 # save word clound to jpg image
379 wordcloud.to_file("wordcloud_v2.jpg")
380
381 """# Unigram and Bigram Analysis"""
382
383 top_10_unigrams = get_top_n_words(selected_df['sentence'], 10)
384 top_10_bigrams = get_top_n_bigrams(selected_df['sentence'], 10)
385
386 # Print top 10 unigrams
387 print("Top 10 Unigrams:")
388 for word, freq in top_10_unigrams:
389     print(f"{word}: {freq}")
390
391 # Print top 10 bigrams
392 print("\nTop 10 Bigrams:")
393 for word, freq in top_10_bigrams:
394     print(f"{word}: {freq}")
395
396 # Plot top 10 unigrams
397 plot_horizontal_bar_chart(top_10_unigrams, 'Top 10 Unigrams')
398
```

```
# Plot top 10 bigrams
plot_horizontal_bar_chart(top_10_bigrams, 'Top 10 Bigrams')

"""# Partitional clustering models
- Build model
- Evaluation model
- Interprete the topic modelling (Word list)

# TF-IDF transformation to distance matrix
"""

documents = selected_df['sentence'].values.astype("U")
# Convert text to distance matrix
# Modify the TfidfVectorizer to include N-grams (bigrams and trigrams)
vectorizer = TfidfVectorizer(ngram_range=(1, 3), max_features=10000)  # You can adjust max_features as needed

features = vectorizer.fit_transform(documents)
features_dense = features.toarray()

"""# Principle Component Analysis"""

# Dimensionality Reduction
pca = PCA(n_components=7)
features_reduced = pca.fit_transform(features_dense)

plt.plot(range(1, len(pca.explained_variance_ratio_) + 1), pca.explained_variance_ratio_)
plt.xlabel('Number of Components')
plt.ylabel('Explained Variance Ratio')
plt.title('Scree Plot')
plt.show()

"""# K-mean text clustering

# Do the hyperparametes tuning
- to find the optimal k-value
"""

K = range(2, 20)
wss = []

for i in K:
  model = KMeans(
      n_clusters=i,
      init= "k-means++",
      random_state= 200
  )

  labels= model.fit(features_reduced).labels_

  wss_iter = model.inertia_
  wss.append(wss_iter)

metrics_centers = pd.DataFrame({
    'Clusters': K,
    'WSS': wss,
})

fig, ax = plt.subplots(1, 1, figsize=(18, 5))

# Plot the elbow method (WSS)
sns.lineplot(ax=ax, x='Clusters', y='WSS', data=metrics_centers, marker='+')
ax.set_title('Within-Cluster Sum of Squares (WSS)')

plt.show()

# Perform K-Mean Clustering with k Clusters.
# Select the optimal k-value = 11, considering by elbow-method at WSS metrics
k = 11
kmeans = KMeans(n_clusters= k, init= "k-means++")

labels= kmeans.fit(features_reduced).labels_

silhouette_score = metrics.silhouette_score(
      features_reduced,
      labels,
      metric= 'euclidean',
      sample_size= len(selected_df),
      random_state= 200
  )

db_index = metrics.davies_bouldin_score(features_reduced, labels)
print(silhouette_score)
print(db_index)

selected_df['k_mean_cluster'] = kmeans.labels_

```

```
# Get feature names (terms) from the vectorizer
terms = vectorizer.get_feature_names_out()

# Create a DataFrame from the original TF-IDF features (before PCA)
tfidf_df = pd.DataFrame(features_dense, columns=terms)
tfidf_df['cluster'] = kmeans.labels_

print_top_terms_per_k_mean_cluster(tfidf_df, terms)

# Plot the 3D k-means clustering distribution of each cluster

fig = plt.figure(figsize=(10, 10))
ax = fig.add_subplot(111, projection='3d', elev=40, azim=80)

# Scatter plot
sc = ax.scatter(selected_df['playCount'], selected_df['likeCount'], selected_df['commentCount'],
                c=selected_df['k_mean_cluster'], cmap='rainbow')

ax.set_xlabel("Play Count", labelpad=10)
ax.set_ylabel("Like Count", labelpad=10)
ax.set_zlabel("Comment Count", labelpad=20)

ax.set_facecolor('white')
plt.title("Tiktok Video Clustering using K Means", fontsize=14)

# Optional: Add a color bar
plt.colorbar(sc)
plt.show()

"""# Hierarchical model"""

ward_cluster = linkage(features_reduced, method= 'ward')

# Plot dendrogram chart
plt.figure(figsize=(10,5))
dendrogram(ward_cluster, labels=documents, leaf_rotation=90, leaf_font_size=10)
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Document')
plt.ylabel('Distance')
plt.show()

# Run Hierarchical algorithm
```

```
K = range(2, 20)
wss = []

for max_clusters in K:
  hirarchical_clusters = fcluster(ward_cluster, max_clusters, criterion= 'maxclust')

  wss.append(compute_wss_hierarchical(features_reduced, hirarchical_clusters))

metrics_centers = pd.DataFrame({
    'Clusters': K,
    'WSS': wss
})

# plot elbow method (WSS)
fig, ax = plt.subplots(1, 1, figsize=(18, 5))
sns.lineplot(ax=ax, x='Clusters', y='WSS', data=metrics_centers, marker='+')
ax.set_title('Within-Cluster Sum of Squares (WSS)')

plt.show()

max_clusters = 9
hirarchical_clusters = fcluster(ward_cluster, max_clusters, criterion= 'maxclust')

sil_score = silhouette_score(features_reduced, hirarchical_clusters)
# 'numpy.float64' object is not callable if you found this error, please move back to run import libraries again (It may lose some required tools)
db_score = davies_bouldin_score(features_reduced, hirarchical_clusters)

print(sil_score)
print(db_score)
selected_df['hirarchical_cluster'] = hirarchical_clusters

# Get feature names (terms) from the vectorizer
terms = vectorizer.get_feature_names_out()

# Create a DataFrame from the original TF-IDF features (before PCA)
tfidf_df = pd.DataFrame(features_dense, columns=terms)
tfidf_df['cluster'] = hirarchical_clusters

cluster_terms = print_top_terms_per_hierarchical_cluster(tfidf_df, terms)

"""# LDA Model"""

data_words = selected_df['sentence'].values.tolist()
```

```
data_words = list(tokenize_LDA_model(data_words)) # tokenization

# Build the bigram and trigram models
bigram = gensim.models.Phrases(data_words, min_count=5, threshold=100) # higher threshold fewer phrases.
trigram = gensim.models.Phrases(bigram[data_words], threshold=100)

bigram_mod = gensim.models.phrases.Phraser(bigram)
trigram_mod = gensim.models.phrases.Phraser(trigram)

# Form Trigrams
data_words = make_trigrams(data_words, trigram_mod, bigram_mod)

# Create Dictionary
texts = data_words
id2word = corpora.Dictionary(texts)

# Term Document Frequency
corpus = [id2word.doc2bow(text) for text in texts]

"""# Hyperparameter Tuning
## C_v
- measure is based on a sliding window, one-set segmentation of the top words and an indirect confirmation measure that uses normalized pointwise mutual information (NPMI) and the cosine similarity
"""

# Topics range
min_topics = 8
max_topics = 9
step_size = 1
topics_range = range(min_topics, max_topics, step_size)

alpha = list(np.arange(0.01, 1, 0.1))
beta = list(np.arange(0.01, 1, 0.1))

# Validation sets
num_of_docs = len(corpus)
corpus_sets = [corpus]

model_results = {'Topics': [],
                 'Alpha': [],
                 'Beta': [],
                 'Coherence': []
                }
data_lemmatized = lemmatization(data_words, allowed_postags=['NOUN'])
```

```
# Can take a long time to run
if 1 == 1:
    for i in range(len(corpus_sets)):
        # iterate through number of topics
        for k in topics_range:
            # iterate through alpha values
            for a in alpha:
                # iterare through beta values
                for b in beta:
                    # get the coherence score for the given parameters
                    cv = compute_coherence_values(corpus=corpus_sets[i], dictionary=id2word, text=data_lemmatized, k=k, a=a, b=b)
                    model_results['Topics'].append(k)
                    model_results['Alpha'].append(a)
                    model_results['Beta'].append(b)
                    model_results['Coherence'].append(cv)

# export the experimental LDA results
file_path = 'coherance_results.xlsx'
pd.DataFrame(model_results).to_excel(file_path, index=False)

# Plot the same of LDA modelling to see the best performance parameter
plt.style.use("_classic_test_patch")
x= [2,3,4,5,6,7,8,9]
y = [0.47502992152505985,
  0.4739038870968999,
  0.5191712392844225,
  0.5696449051925503,
  0.5540307696200729,
  0.5702856314238206,
  0.5962194674972741,
  0.5820308009596259]

plt.plot(x, y, marker='o', color='#2C15FF')
plt.xlabel('the number of topics')
plt.ylabel('Coherence Score')
plt.show()

"""# Build LDA with optimal value"""

num_topics = 8

# Build LDA model
lda_model = gensim.models.LdaMulticore(corpus=corpus,
```

```
                                                id2word=id2word,
                                                num_topics=num_topics,
                                                random_state=100,
                                                chunksize=100,
                                                passes=10,
                                                alpha=0.81,
                                                eta=0.01)

coherence_model_lda = CoherenceModel(model=lda_model, texts=data_words, dictionary=id2word, coherence='c_v')
coherence_lda = coherence_model_lda.get_coherence()
print('Coherence Score: ', coherence_lda)
# Get topic distribution for each document
doc_lda = lda_model.get_document_topics(corpus, minimum_probability=0)
selected_df['LDA_cluster'] = get_dominant_topic(lda_model, corpus)
# Plot the distribution of each topic of LDA model
pyLDAvis.enable_notebook()
vis = pyLDAvis.gensim.prepare(lda_model, corpus, id2word, mds='mmds', R=10)
vis

# print dominant topic with topic_keywords of LDA model
data_dict = {'dominant_topic':[], 'perc_contribution':[], 'topic_keywords':[]}

for i, row in enumerate(lda_model[corpus]):
    row = sorted(row, key=lambda x: x[1], reverse=True)
    for j, (topic_num, prop_topic) in enumerate(row):
        wp = lda_model.show_topic(topic_num)
        topic_keywords = ", ".join([word for word, prop in wp])
        data_dict['dominant_topic'].append(int(topic_num))
        data_dict['perc_contribution'].append(round(prop_topic, 3))
        data_dict['topic_keywords'].append(topic_keywords)
        break

df_topics = pd.DataFrame(data_dict)
print(df_topics)

# mapping the video category to dominant_topic (LDA_cluster)
selected_df["Category"] = selected_df["LDA_cluster"].apply(convert_num_cluster_to_text)

"""# Sentiment Analysis"""

sia = SentimentIntensityAnalyzer()
selected_df['token'] = selected_df['sentence'].apply(lambda x: nltk.word_tokenize(x))
```

```
res = {}
for i, row in tqdm(selected_df.iterrows(), total=len(selected_df)):
  text = row['sentence']
  myid = row['id']
  res[myid] = sia.polarity_scores(text)

vaders = pd.DataFrame(res).T
vaders = vaders.reset_index().rename(columns= {'index': 'id'})
vaders = vaders.merge(selected_df, how= 'left')
# apply the emotional compund score to the dataframe
vaders['Sentiment'] = vaders['compound'].apply(classify_sentiment)

total_count= len(selected_df)
result = selected_df['k_mean_cluster'].value_counts().reset_index(name='Amount')
result['% Amount'] = (result['Amount'] / total_count) * 100
result['% Amount'] = result['% Amount'].round(2)
result

# Plot the proportion of each emotional type (positive, neutral, negative)
total_count= len(vaders)
result = vaders['Sentiment'].value_counts().reset_index(name='Amount')
result['% Amount'] = (result['Amount'] / total_count) * 100
result['% Amount'] = result['% Amount'].round(2)

fig = px.bar(result, x="Sentiment", y="% Amount", color= "Amount", color_continuous_scale='GnBu')
fig.show()

"""# Visual the insights"""

total_rows = len(selected_df)
category_dist_fig = calculate_category_distribution(selected_df, total_rows, 'Category')

fig = px.bar(category_dist_fig, x="% Amount", y="Category", color="Amount", orientation='h', color_continuous_scale='GnBu')
fig.update_layout(yaxis_title="Category")
fig.show()

popular_distribution_top_100 = find_popular_distribution(selected_df, 'playCount', 'Average Views', 100)
fig = px.bar(popular_distribution_top_100, x="AMOUNT", y="Category", color="Average Views", orientation='h', color_continuous_scale='Emrld')
fig.update_layout(yaxis_title="Category")
fig.show()
```

```

engagement_distribution_top_100 = find_popular_distribution(selected_df, 'commentCount', 'Average Comments', 100)
fig = px.bar(engagement_distribution_top_100, x="AMOUNT", y="Category", color="Average Comments", orientation='h', color_continuous_scale='Purp')
fig.update_layout(yaxis_title="Category")
fig.show()

# The percentage of the sentiment score of each category
total_count= len(vaders)
result = vaders.groupby(['Category', 'Sentiment']).size().reset_index(name='Amount')
result['% Amount'] = (result['Amount'] / total_count) * 100
result['% Amount'] = result['% Amount'].round(2)

fig = px.bar(result, x="Category", y="% Amount", color="Sentiment", text="Amount")
fig.show()

# Print the top 100 videos based on views
n = 100
result = selected_df[['id', 'description', 'playCount', 'likeCount', 'Category']].rename(columns={
      'description': 'Video Description',
      'playCount': 'Views',
      'likeCount': 'Likes'
  }).sort_values(by='Views', ascending=False).head(n)

result
```