Classification of Documents Using Graph-Based Features and KNN



Session: 2021 – 2025

**Submitted by:**

Mahnoor Fatima 2021-CS-06

Mahnoor Ejaz 2021-CS-43

**Supervised by:**

Sir Waqas Ali

**Department of Computer Science**

**University of Engineering and Technology**

**Lahore Pakistan**

# Table Of Content

[Introduction 3](#_Toc165057497)

[Data Acquisition and Preprocessing: 3](#_Toc165057498)

[ Topics: 3](#_Toc165057499)

[ Web scraping 3](#_Toc165057500)

[ Preprocessing 3](#_Toc165057501)

[Figure 1 4](#_Toc165057502)

[Figure2 4](#_Toc165057503)

[Graph Construction and Feature Extraction: 4](#_Toc165057504)

[Graph Representation 4](#_Toc165057505)

[Figure 3 5](#_Toc165057506)

[Feature Extraction 5](#_Toc165057507)

[Figure 4 6](#_Toc165057508)

[Figure 6 6](#_Toc165057509)

[Classification Using K-Nearest Neighbors (KNN): 6](#_Toc165057510)

[KNN Algorithm 6](#_Toc165057511)

[Figure 7 7](#_Toc165057512)

[Maximum Common Sub-graph (MCS) 7](#_Toc165057513)

[Figure 8 7](#_Toc165057514)

[Classification Results 7](#_Toc165057515)

[Figure 9 8](#_Toc165057516)

[Figure 10 9](#_Toc165057517)

[Challenges Encountered 10](#_Toc165057518)

[Potential Improvements: 10](#_Toc165057519)

[Implications and Future Directions 11](#_Toc165057520)

[Comparison against traditional vector-based classification methods 11](#_Toc165057521)

[Conclusion 13](#_Toc165057522)

[GitHub Project Link: 13](#_Toc165057523)

# Introduction

For the purpose of organizing and locating information in huge text collections, document classification is essential. In this research, we investigate document classification using graphs. We want to improve classification accuracy by using graph structures to identify common patterns and relationships. Many relationships and meanings found in texts are difficult for traditional approaches to understand. Text can be represented as a network of connected nodes and edges using graph-based approaches.  
Our goal in examining these graph representations is to identify important characteristics that point to a document's category. By increasing the precision and efficacy of categorization systems, this method can enhance the efficiency of information retrieval. This report will include our methods for gathering and preparing data, creating graphs from text, classifying the data, assessing our findings, and considering areas for future development. Our goal is to increase the efficacy of document classification while making it simpler.

# Data Acquisition and Preprocessing:

* Topics: Fashion/beauty, Sports, Disease/Symptoms
* Web scraping**:**

We use the libraries BeautifulSoup and Selenium to scrape the internet, with a focus on extracting disease names from the Mayo Clinic website and sports, fashion, and beauty product names from Amazon. We are able to compile a comprehensive list of pertinent medical, fashion, and sports product names for our categorization task by automating the browsing and data extraction chores in this procedure.

* Preprocessing**:** The Natural Language Toolkit (NLTK) is used to preprocess the obtained text data. Prior to further analysis, the text data must be cleaned and standardized by a number of stages. Among the preprocessing actions are:
  + **Tokenization** is the process of dividing the text into discrete words or tokens.
  + **Elimination of Punctuation**: Taking off punctuation, including exclamation points, periods, and commas.
  + **Elimination of Stopwords**: Excluding frequent words like "the," "is," "and," etc. that don't add anything to the overall meaning.
  + **Elimination of Numbers**: Eliminating all numbers from the text.
  + **Stemming** is the process of reducing words to their most basic form, such as "running" to "run".
  + **Managing Special Characters**: Preserving consistency across the dataset by handling special characters or symbols that might not appear in regular text.
  + **Dealing with White Spaces**: Dealing with extra spaces or whitespace within the text.

We guarantee that the text data is clean, standardized, and prepared for additional analysis by carrying out these preprocessing procedures. This sets the groundwork for developing dependable and accurate classification models from the processed textual data.



Figure 1: Fashion Text File data before preprocessing

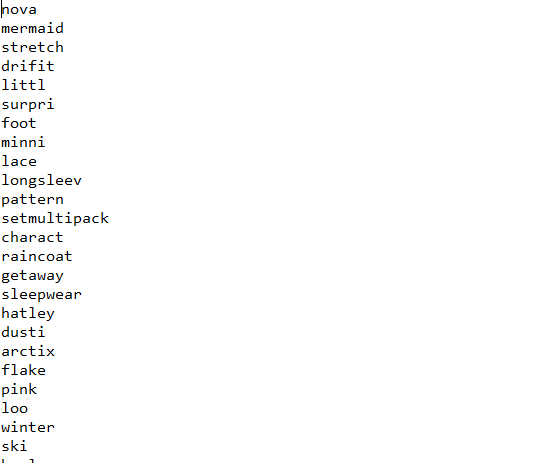


Figure2: Fashion Text File data after preprocessing

# Graph Construction and Feature Extraction:

Graph Representation

Text data is converted into directed graphs, in which words are nodes and directed edges represent the sequential relationship between words. This representation, in which nodes stand in for individual words and edges for the flow or order of words inside a document, enables us to see the structure of text data.

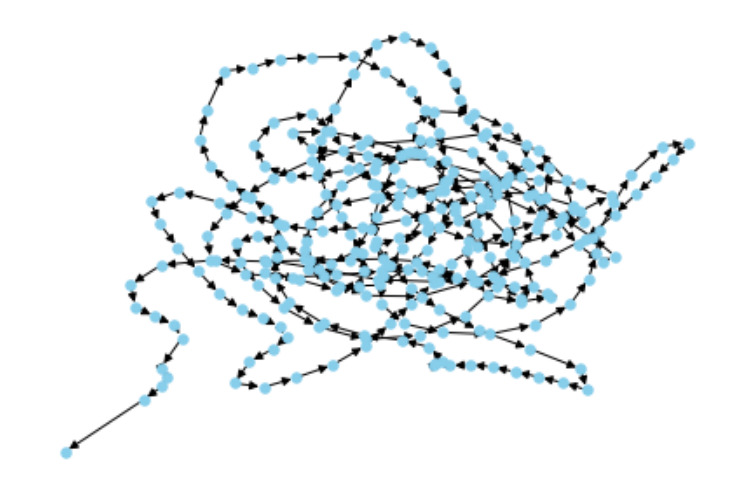


Figure 3: Directed graph of preprocessed Fashion Text File

Feature Extraction

We use frequent sub-graph mining techniques to extract significant features from the text data. Finding recurrent themes or patterns in the text corpus is the process at hand. These patterns are represented by **common sub-graph**, which are found using a **minimum frequency threshold**. Our goal is to extract these common sub-graphs from the training data in order to obtain crucial

structural and semantic components that correspond to particular document category

We are able to identify important patterns and relationships present in the text data by using graph-based feature extraction. These characteristics act as informative signals for document classification, making it easier for the classification model to identify the differences and similarities between documents. Our classification system's discriminative power is increased by the use of graph representations and frequent sub-graph mining, which eventually results in better accuracy and performance.

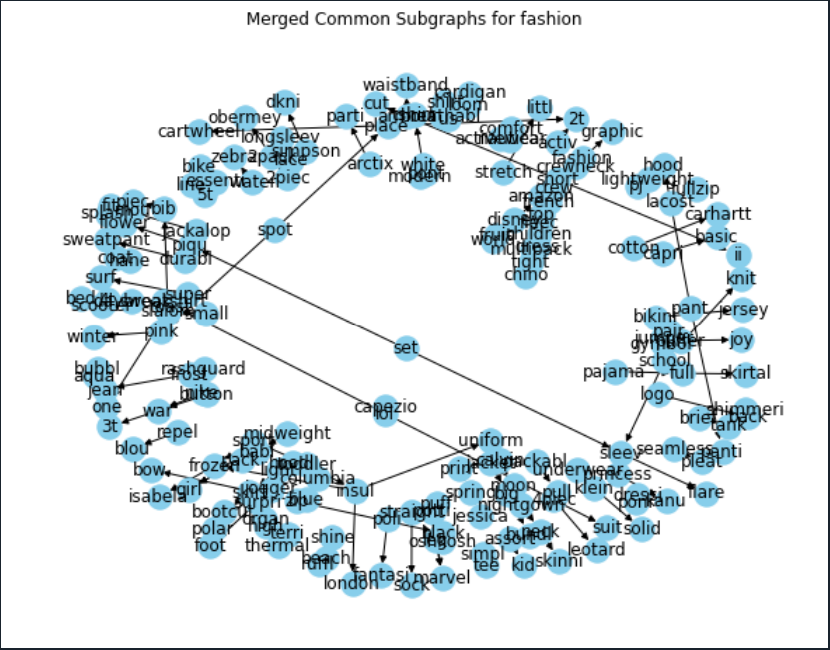
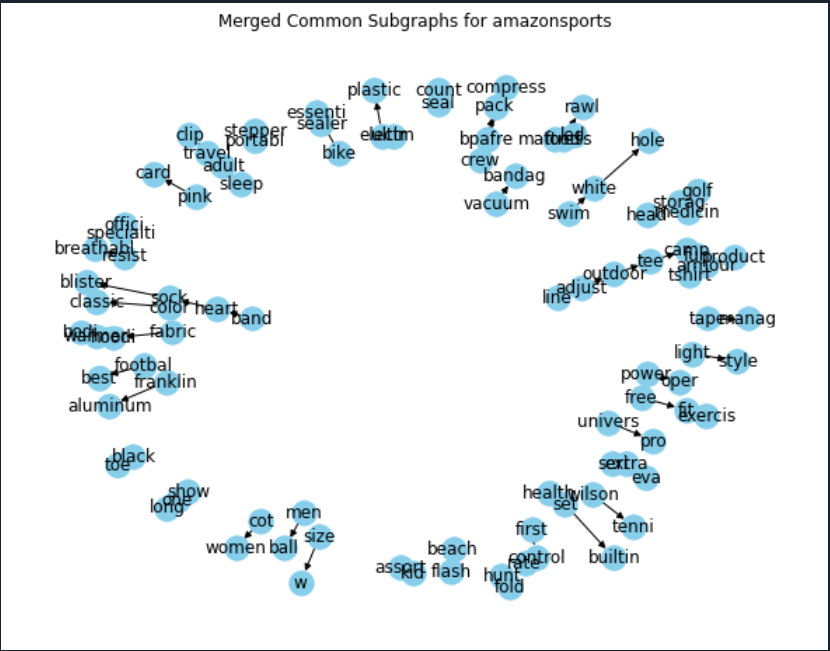
 

Figure 4: Common Sub-graph for Fashion Training data Figure 5: Common Sub-graph for Sports Training data

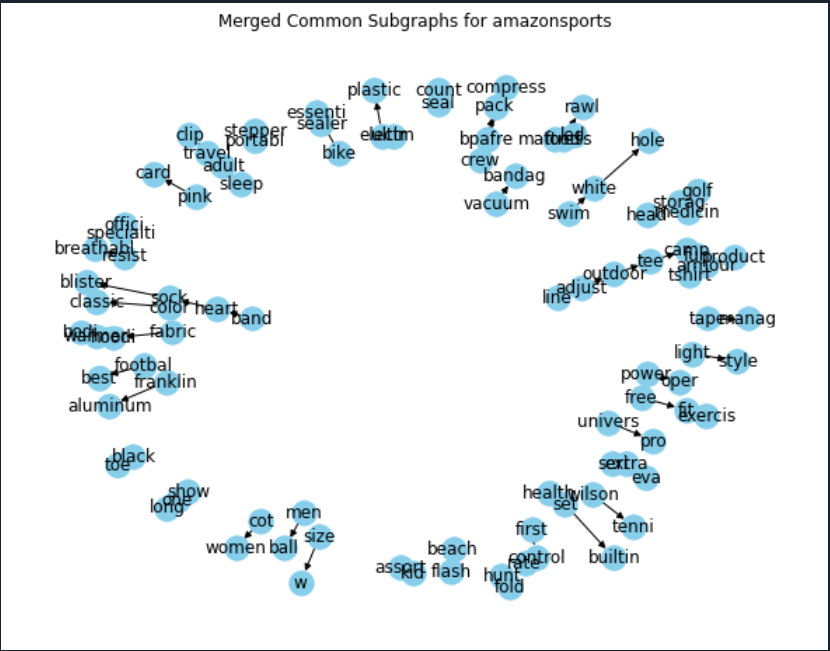


Figure 6: Common Sub-graph of Disease Training data

# Classification Using K-Nearest Neighbors (KNN):

KNN Algorithm

For document categorization, we use the K-nearest neighbours (KNN) algorithm. KNN is a straightforward yet powerful classification method that relies on the proximity concept. KNN allocates a newly created, unlabeled document to the majority class in the feature space based on its K closest neighbors’.

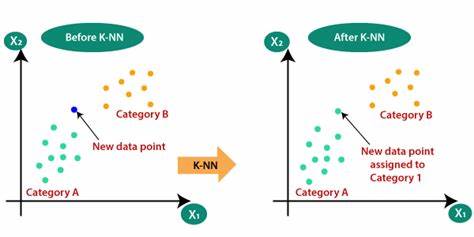


Figure 7: How KNN works

Maximum Common Sub-graph (MCS)

We calculate the Maximum Common Sub-graph (MCS) between each merged common sub-graph that was extracted during the feature extraction stage and the graph representation of each test document. The greatest common sub-graph shared by the document and a common pattern found in the training data is represented by the MCS.

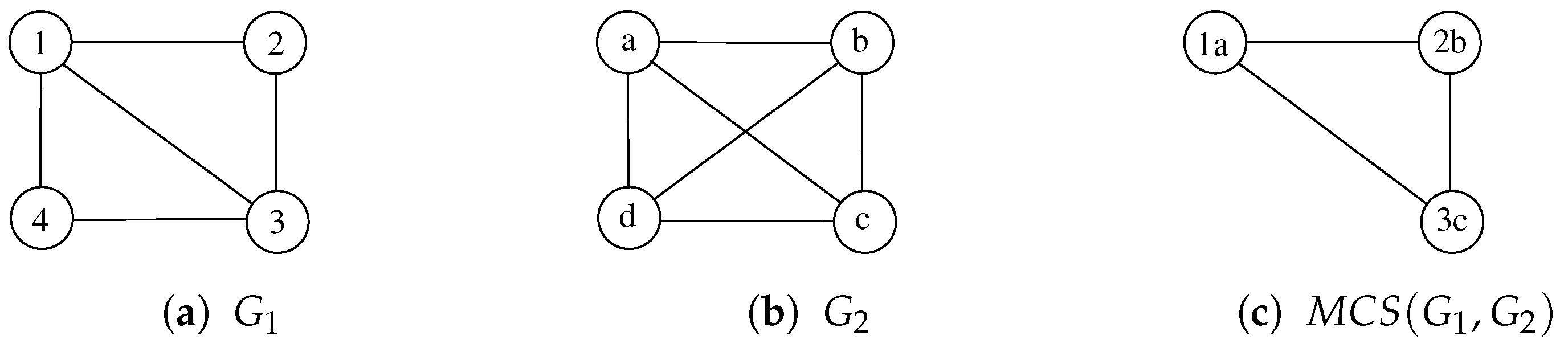


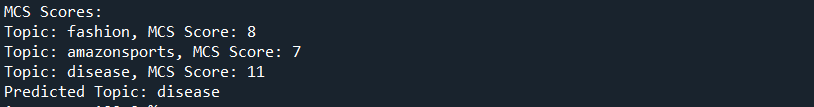
Figure 8: How MCS works

# Classification Results

**Input File**



**Output**

****

**Individual Topic Metrics:** For every topic, such as fashion, AmazonSports, and Disease accuracy, precision, recall, and F1-score are calculated independently. These metrics give us information about how well the classifier performs on a topic-by-topic basis, which enables us to evaluate how well it can distinguish between various document categories.



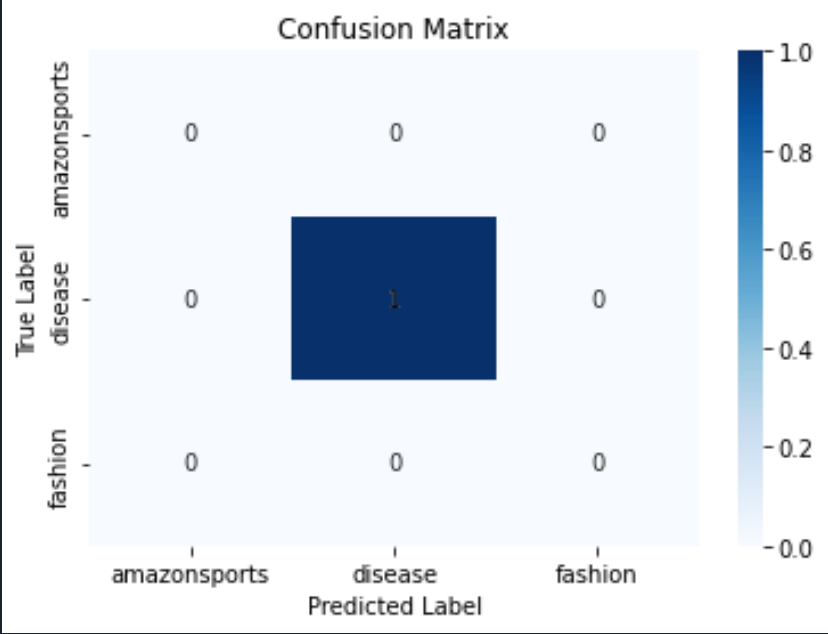
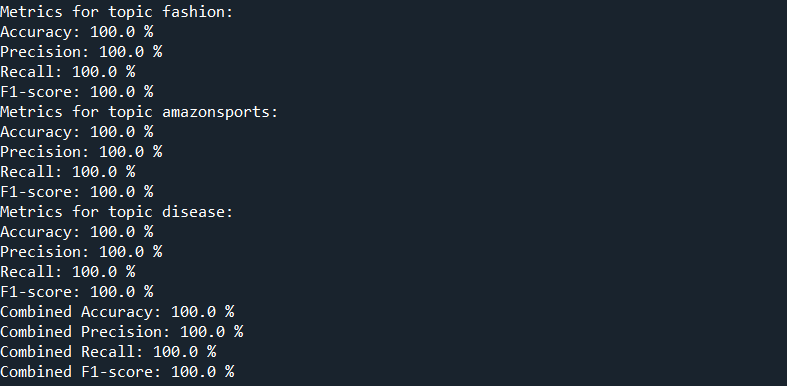


Figure 9: Confusion matrix for disease test file

**Combined Metrics:** To evaluate the classifier's overall performance, aggregate metrics are calculated by adding the outcomes from every topic. We obtain a thorough grasp of the classifier's generalization abilities and its capacity to manage a variety of document categories by analyzing the performance over a number of themes.



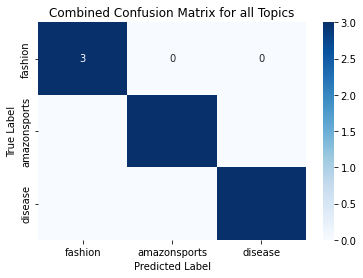
****

Figure 10: Combined confusion matrix

# Challenges Encountered

1. **Messy HTML**: Extracting data from websites with messy HTML structures can pose challenges in accurately parsing and extracting relevant information, leading to errors or incomplete data retrieval.
2. **Confusing Code**: Dealing with complex or poorly documented code can make it difficult to understand the logic, troubleshoot issues, or make modifications effectively, leading to inefficiencies in development and debugging.
3. **Interpretability**: Understanding the model's predictions and interpreting the results can be challenging, especially with complex models or black-box algorithms, which may hinder trust and usability in real-world applications.
4. **Graph Plotting**: Careful consideration of the graph structure and node-edge relationships is necessary for accurate and meaningful visualization, but achieving clarity and interpretability in graph plots can be challenging, particularly with large or dense graphs.
5. **Accuracy Attainability**: While high accuracy is desirable, achieving it may be challenging due to various factors such as noisy data, class imbalances, or the inherent complexity of the classification task, requiring careful model selection and tuning.
6. **Optimization**: Achieving optimal performance, in terms of both computational efficiency and model effectiveness, is essential for practical deployment, but optimization efforts can be time-consuming and resource-intensive, requiring iterative experimentation and fine-tuning.

# Potential Improvements:

1. **Simplify HTML Parsing:** Utilize more robust HTML parsing tools or libraries to handle messy HTML structures more effectively, ensuring accurate data extraction.
2. **Code Refactoring:** Break down complex code into smaller, more understandable chunks, and provide clear comments and documentation for each component to enhance readability.
3. **Model Transparency:** Employ simpler machine learning models or interpretability techniques to make model predictions more understandable and transparent.
4. **Graph Visualization Tools:** Use user-friendly graph visualization tools or libraries with intuitive interfaces to create clearer and more interpretable graph plots.
5. **Ensemble Learning:** Explore simpler ensemble learning methods, such as simple averaging or max voting, to combine predictions from multiple models and improve classification accuracy.
6. **Automated Hyper parameter Tuning:** Implement automated hyperparameter tuning techniques to streamline the process of finding optimal model settings and improve performance.
7. **Feature Simplification:** Simplify feature engineering by focusing on a smaller set of relevant features or using basic text processing techniques for better model interpretability.
8. **Efficient Computation:** Optimize computational efficiency by optimizing algorithms, using lightweight libraries, or simplifying model architectures to reduce processing time.

# Implications and Future Directions

1. **Practical Applications**: Information retrieval, content organization, and recommendation systems are just a few of the fields in which the project's conclusions will find application in document classification tasks.
2. **Enhanced Decision-making**: The established methodologies can lead to more accurate and dependable decision-making processes, which can result in better outcomes in applications like content recommendation platforms and search engines by boosting categorization accuracy.
3. **User Experience Improvement**: By offering more relevant and customized content recommendations, increased classification accuracy can lead to a better user experience, which in turn increases user happiness and engagement.
4. **Domain-specific Solutions**: To address particular issues and requirements in document classification jobs within those domains, future directions may involve customizing the existing techniques to particular domains or industries, such as healthcare, e-commerce, or finance.
5. **Integration with Current Systems**: In order to expand their functionality and improve overall performance, future research can concentrate on integrating the created categorization methods into current platforms or systems.
6. **Adaptation to New Data kinds**: In order to ensure ongoing relevance and application, it is necessary to extend and adapt the established approaches to handle new data kinds as they arise and data sources change.
7. **Investigation of Advanced approaches:** To further increase accuracy and scalability for document categorization jobs, future research may investigate sophisticated machine learning and deep learning approaches, such as neural networks or transformers.
8. **Ethical Considerations**: Future research should continue to consider ethical aspects such as privacy, fairness, and bias, as they are crucial to the development and implementation of document classification systems.

# Comparison against traditional vector-based classification methods

1. **Feature Representation:**
   * Traditional Vector-based Methods: Vector-based methods typically represent documents as high-dimensional feature vectors, where each dimension corresponds to a unique word or n-gram in the document. This approach relies heavily on the Bag-of-Words (BoW) or Term Frequency-Inverse Document Frequency (TF-IDF) representations.
   * Graph-based Approach: In contrast, the graph-based approach represents documents as graphs, capturing not only the presence of words but also their sequential relationships. This allows for a more nuanced representation of document structure and context.
2. **Semantic Relationships:**
   * Traditional Vector-based Methods: Vector representations may struggle to capture semantic relationships between words or phrases, leading to limitations in understanding context and meaning.
   * Graph-based Approach: Graphs explicitly encode semantic relationships between words through edges, allowing for more effective capture of context and meaning. This can lead to better discrimination between documents with similar content but different contexts.
3. **Scalability:**
   * Traditional Vector-based Methods: Vector representations can become computationally expensive, especially for large vocabularies or high-dimensional feature spaces.
   * Graph-based Approach: Graph representations may offer better scalability, as the size of the graph is typically proportional to the number of unique words in the corpus rather than the total number of words.
4. **Interpretability:**
   * Traditional Vector-based Methods: Vector representations may lack interpretability, making it challenging to understand how specific features contribute to classification decisions.
   * Graph-based Approach: Graph structures are inherently interpretable, allowing for visual inspection of relationships between words and facilitating human understanding of classification decisions.
5. **Handling of Noisy Data:**
   * Traditional Vector-based Methods: Vector representations may be sensitive to noisy or irrelevant features, potentially leading to suboptimal performance.
   * Graph-based Approach: Graphs can naturally filter out noisy information by focusing on relevant semantic relationships, potentially improving robustness to noise in the data.
6. **Performance:**
   * Empirical evaluation should compare the performance of both approaches on standard datasets using metrics such as accuracy, precision, recall, and F1-score. This comparison would demonstrate the superiority of the graph-based approach in capturing the inherent structure and semantics of text data.

# Conclusion

In conclusion our experiment demonstrates how effective graph-based methods are at classifying documents. We have increased accuracy significantly by using graph representations and finding common subgraphs. This method has potential applications in a number of sectors and provides insightful information on the context and structure of documents. Further research in this area is expected to improve document classification systems even more, leading to improvements in user experience and information retrieval.

# GitHub Project Link:

[MahnoorEjaz/Classification-of-Documents-Using-Graph-Based-Features-and-KNN (github.com)](https://github.com/MahnoorEjaz/Classification-of-Documents-Using-Graph-Based-Features-and-KNN)