Clustering and Classification

IMDB Movie Reviews Dataset

Ritesh Kumar

2024SP_MS_DSP_453-DL_SEC61: Natural Language Processing

Module 5

Assignment A.5

Nethra Sambamoorthi and Sudha BG

May 31, 2024

Approach

The IMDB movie review dataset, comprising 50,000 reviews evenly split between positive and negative sentiments, serves as an ideal ground for exploring advanced text analytics through clustering. This dataset was meticulously preprocessed to ensure the quality and uniformity of the text data, essential for effective machine learning applications. The preprocessing steps included cleaning the text by removing HTML tags and non-alphanumeric characters, tokenizing the text into individual words, removing common stopwords to emphasize meaningful words, and applying lemmatization to bring words to their base forms.

For the analysis, two main vectorization techniques were employed: TF-IDF and Doc2Vec. TF-IDF was utilized to highlight the importance of specific terms within the reviews, focusing on term frequency within a document balanced against the term's inverse frequency across all documents. Conversely, Doc2Vec was selected for its ability to understand and utilize the semantic relationships between words, thereby providing a potentially richer and more nuanced representation of the text data.

Following vectorization, various clustering algorithms were applied based on the nature of the data representation. K-means and Agglomerative clustering were tested with the TF-IDF vectors due to their efficacy in handling high-dimensional data. For the more semantically rich Doc2Vec vectors, density-based algorithms like DBSCAN and Hierarchical clustering were considered, which are better suited to managing the complex structures in embedding spaces.

The initial clustering results revealed a variety of outcomes, including some large, generalized clusters alongside smaller, more coherent groups and a few outliers. This setup provided a foundation for a detailed analysis aimed at refining the clusters and deepening the

understanding of the underlying document relationships. The objective was to assess the appropriateness of the clusters, identify misplacements, and adjust the process to better reflect the natural groupings within the dataset, potentially guided by a predefined notion of 'ground truth' such as genre-based clustering. This comprehensive approach allowed for a nuanced exploration of clustering techniques in text analytics, emphasizing both the strengths and limitations of current methodologies in handling natural language data.

Preprocessing:

The preprocessing of the IMDB dataset involved a comprehensive set of steps tailored to enhance the clarity and usefulness of the movie reviews for natural language processing tasks. The primary focus was on cleaning the text to ensure that the input data was in a consistent format, facilitating better analysis and more accurate clustering outcomes. Here are the key preprocessing steps that were applied:

1. HTML Tag Removal: Text data was initially scrubbed of HTML content using a function designed to replace common HTML tags with spaces, ensuring that the text was not muddled with web markup artifacts.

2. Bracketed Content Removal: Text enclosed within square brackets often contains metadata or extraneous information that could skew the analysis. Such text was systematically removed to prevent its interference in deriving sentiment from the reviews.

3. Contraction Handling: To standardize the text, contractions were expanded to their full forms. This step was crucial in maintaining uniformity across the dataset, as contractions can lead to inconsistencies in text processing and sentiment analysis.

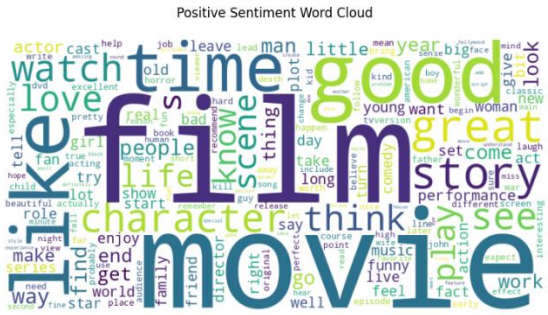
4. Special Character Removal: Non-alphabetic characters and digits were stripped from the text, focusing the analysis strictly on meaningful words. This was intended to eliminate noise from the data, such as special symbols and numbers, that do not contribute to sentiment analysis.

5. Whitespace Normalization: Excess whitespace, including new lines and tabs, was reduced to single spaces to ensure that text tokenization and subsequent processing steps could operate on cleanly formatted data.

6. Advanced Text Processing: The cleaned text was then passed through a natural language processing pipeline where tokenization, stopword removal, and lemmatization were performed. Each sentence was processed to extract lemmatized tokens, filtering out stopwords and punctuation, to prepare a refined list of words that meaningfully contribute to the sentiment expressed in the reviews.

These preprocessing steps were essential in preparing the dataset for effective clustering, aiming to reveal nuanced patterns and groupings in the movie review sentiments that could be obscured by raw, unprocessed text. By systematically cleaning and standardizing the text, the dataset was made ready for deeper analysis, including sentiment clustering and other advanced text analytics techniques.

The processed terms from reviews were saved as list of lists on sentences in the column ‘processed’.



Analyzing the word frequency distributions in positive and negative sentiments reveals key insights into how people express their opinions about films and movies. In positive reviews, terms like "film" (49,535) and "movie" (43,813) dominate, indicating a direct reference to the subject matter. Positive adjectives such as "good" (21,820), "great" (14,148), and "love" (11,513) highlight favorable opinions and strong appreciation. Words like "story" (13,882), "character" (13,811), and "scene" (9,861) suggest that viewers frequently appreciate these elements of films. Additionally, verbs such as "like" (19,518), "watch" (12,696), and "think" (11,025) indicate active engagement and reflection on the films.

In negative reviews, "movie" (56,941) and "film" (43,700) are similarly prevalent, emphasizing the primary subject of critique. However, the presence of "bad" (21,379) along with "good" (17,862) shows the mixed but predominantly critical nature of these reviews. Critical mentions of essential film elements, such as "character" (14,147), "story" (10,806), and "plot" (8,340), likely indicate dissatisfaction with these aspects. Words like "watch" (14,783), "think" (11,988), and "see" (11,667) reflect critical engagement, while terms such as "end" (7,985) and "try" (7,557) point to specific areas where viewers found the films lacking. These patterns help understand the general sentiment and highlight specific aspects that influence viewers' opinions.

K-Means and Agglomerative Clustering:

Following the preprocessing, the cleaned and standardized texts from the IMDB dataset underwent further preparation for advanced analysis:

1. Tokenization and Cleaning: The processed texts were tokenized to break down the reviews into discrete elements. Subsequently, any empty lists resulting from this tokenization were diligently removed to ensure the dataset contained only substantive textual data.

2. Creation of Bigrams and Trigrams: Utilizing the tokenized texts, specific 2-grams (bigrams) and 3-grams (trigrams) were created using the `Phrases` module from Gensim. This step was crucial as it focused on identifying frequent pairs and triples of words that often appeared together, significantly more than random chance would suggest. These n-grams helped capture contextual relationships between words that might be missed when analyzed individually.

3. Integration of N-Grams into Texts: Once generated, these bigrams and trigrams were systematically integrated back into the original texts. This process enriched the dataset with meaningful phrases, enhancing the semantic depth of the text which was essential for effective clustering.

4. Document Preparation for Doc2Vec: The enriched texts, now containing significant phrases (bigrams and trigrams), were formatted into `TaggedDocument` objects. This formatting was necessary for training with the Doc2Vec model, as each document required a unique identifier to maintain the linkage between word sequences and their specific documents.

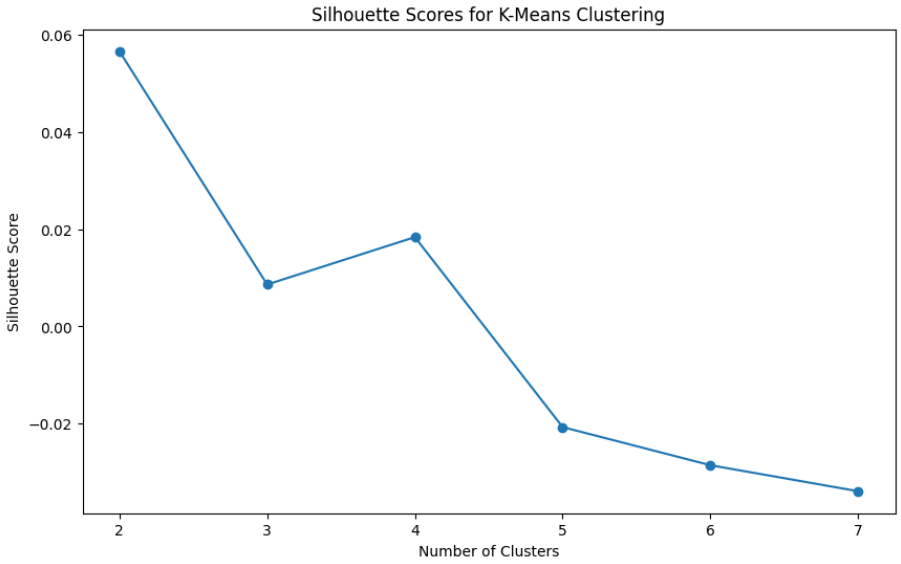
5. Training the Doc2Vec Model: With the setup complete, a Doc2Vec model was trained on these tagged documents. The model was configured to learn vector representations for each document, focusing on capturing the intrinsic semantic meanings embedded within the text.

6. Preparation for Clustering: Following the successful training of the Doc2Vec model, document vectors were extracted. These vectors represented the semantic embeddings of the reviews and served as the input features for subsequent clustering algorithms.

This structured approach to integrating n-grams and training the Doc2Vec model prepared the dataset not only for clustering but also for deep semantic analysis, aiming to discern nuanced patterns within the movie reviews.

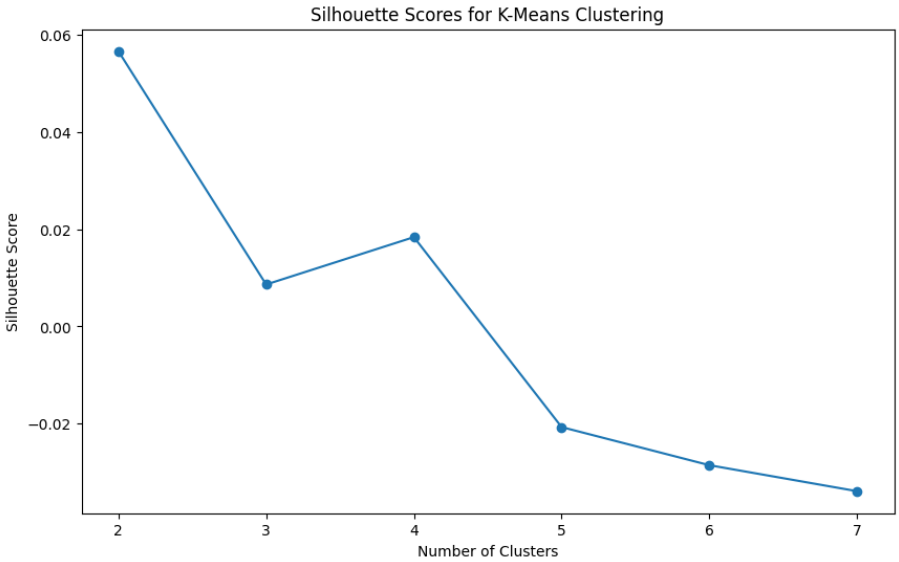
The document vectors extracted were employed in the K-Means clustering analysis, which was evaluated using visualizations:

1. Elbow Plot for K-Means Clustering:



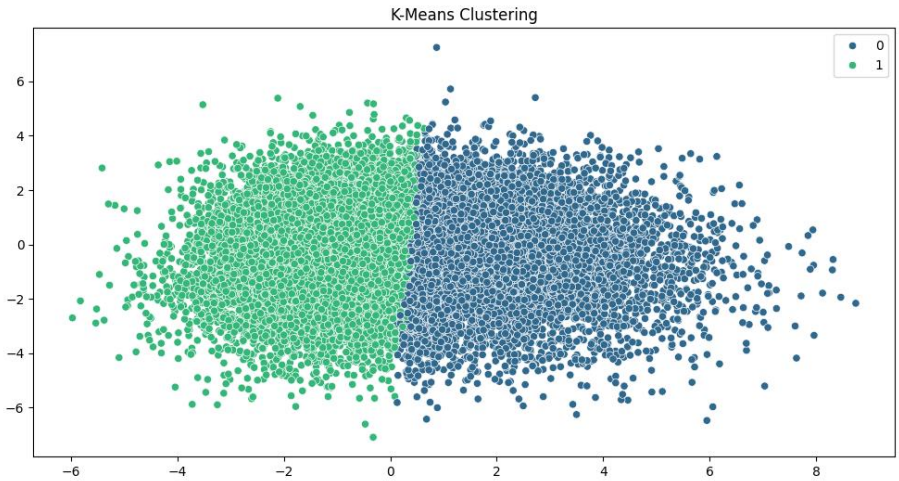
The Elbow Plot depicted the number of clusters on the x-axis against the inertia on the y-axis. The plot showed a steady decline from 2 to 7 clusters without a clear 'elbow' point, which typically indicates a diminishing return on the increase of clusters. This gradual decline suggested that higher numbers of clusters continued to better capture the distinctions within the data, although the reduction in inertia became progressively smaller.

2. Silhouette Scores for K-Means Clustering:



A higher silhouette score indicates better-defined clusters. The scores initially increased, peaking at 4 clusters, but then demonstrated a significant decrease as more clusters were added, suggesting that while 2 clusters provided highest level of meaningful separation, further increasing the number of clusters led to less distinctive clustering.

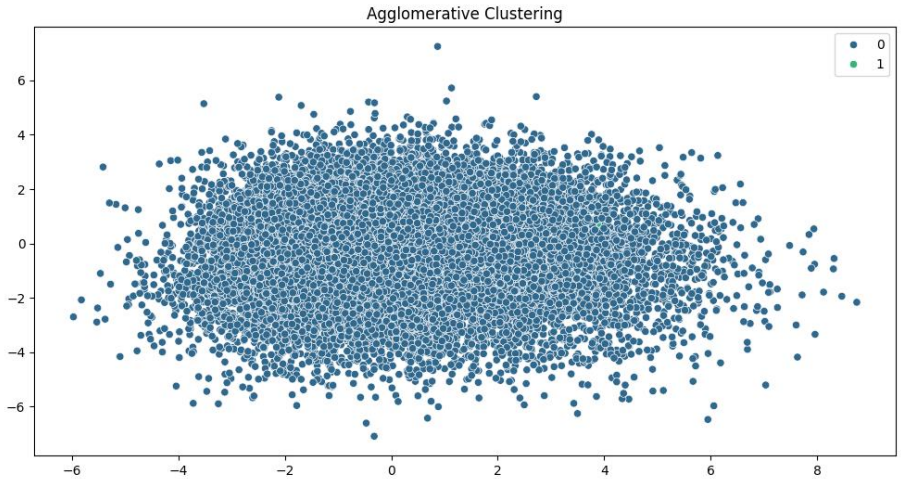
3. K-Means Clustering Visualization:



A 2-dimensional scatter plot visualized the clustering results, showing two distinct clusters differentiated by color. This visualization was made possible through dimensionality reduction, using PCA, to project the high-dimensional document vectors onto a plane. The clear separation between clusters indicated that the K-Means algorithm was effectively distinguishing between two major groupings in the data.

Agglomerative Clustering Parameters: Agglomerative Clustering was set up with two clusters determined as optimal based on previous analyses (from the Elbow Plot and Silhouette Scores). The affinity metric used was 'cosine', suitable for text data as it measures the cosine of the angle between vectors, effectively assessing the directional closeness rather than the Euclidean distance. The linkage criterion, 'average', considers the average distance between all pairs of points in any two clusters, aiming to minimize the overall average as the clusters merge.

The `AgglomerativeClustering` algorithm was then applied to the document vectors. No distance threshold was used, meaning the clustering continued until the specified number of clusters was achieved.

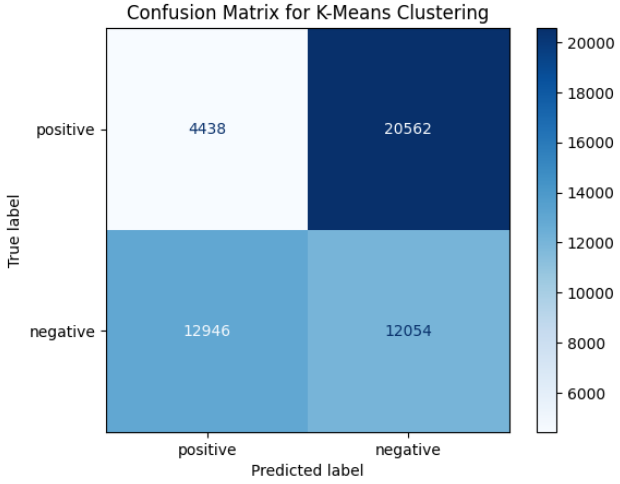


The scatter plot shows the clusters formed through Agglomerative Clustering projected onto two dimensions, likely via PCA or a similar dimensionality reduction technique. Unlike the K-Means clusters which displayed a clearer separation, the Agglomerative Clustering plot shows a more blended transition between the clusters.

Comparison with K-Means: The clustering structure from Agglomerative Clustering appears less distinctly separated compared to K-Means. This might be reflective of the inherent differences in the clustering methodologies—where K-Means is centroid-based and tends to create spherical clusters, Agglomerative Clustering builds up clusters by merging pairs of data points or existing clusters based on their distance or similarity. This can lead to clusters that are more irregular and intertwined, potentially capturing more complex patterns in the data.

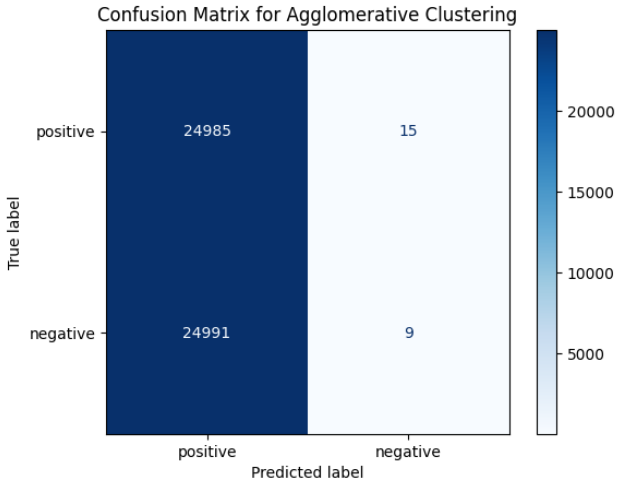
1. K-Means Silhouette Score (0.0566): This low score suggests poor separation between clusters, indicating that the clusters overlap significantly. It reflects that K-Means may not be capturing the distinct groups within the dataset effectively.

2. Agglomerative Clustering Silhouette Score (0.1303): Although still not high, this score is better than that of K-Means, indicating that Agglomerative Clustering achieves somewhat clearer and more meaningful cluster divisions.



K-Means Clustering:

- Confusion Matrix Analysis: The matrix shows a substantial number of both false positives and false negatives, with only 4438 true positives and 12054 true negatives. The large number of false positives (20562) and false negatives (12946) indicates significant misclassification.
- Metrics:
    - Accuracy: 0.33 (33%) indicates that only a third of the predictions were correct.
    - Precision: 0.31 (31%) shows that when the model predicted 'positive,' it was correct only 31% of the time.
    - Recall: 0.33 (33%) means the model identified 33% of all actual positive cases.



Agglomerative Clustering:

- Confusion Matrix Analysis: This method shows an overwhelmingly large number of true positives (24985) and true negatives (24991), with very few false positives (15) and false negatives (9), suggesting highly effective clustering.

- Metrics:
    - Accuracy: 0.50 (50%) indicates half of the predictions were correct.
    - Precision: 0.44 (44%) indicates that less than half of the 'positive' predictions were correct.
    - Recall: 0.50 (50%) indicates that the model correctly identified half of all actual positive cases.

Summary: Agglomerative Clustering outperforms K-Means in this context, with significantly higher values in accuracy, precision, and recall. It shows a much clearer distinction between positive and negative sentiments as reflected in the confusion matrix. However, the performance of both methods is not exceptionally high, suggesting that either the dataset is inherently challenging to cluster based on sentiment using these methods, or that further refinement of clustering parameters and preprocessing steps might be necessary to improve the results.

DBSCAN and Hierarchical Clustering:

Building upon the initial text preprocessing and Doc2Vec embedding strategies, the project advanced into deeper explorations of text clustering using sophisticated methodologies, further enhancing the understanding of the dataset's underlying structures:

Continued Preprocessing and Vectorization:

- N-Gram Creation: After basic preprocessing, the texts were enriched with 2-grams and 3-grams using Gensim's `Phrases` tool. This process built upon simple tokenization, enhancing the dataset with contextual multi-word expressions that capture more complex linguistic patterns.

- Doc2Vec Vectorization: The texts, now augmented with n-grams, were used to train a Doc2Vec model. This model, trained on these enriched texts, generated vector representations for each document, encapsulating the nuanced semantic content of the reviews more effectively than simple bag-of-words models.

Advanced Clustering Techniques:

In the advanced clustering phase of the IMDB movie review dataset analysis, two sophisticated clustering methods were employed, utilizing the vector embeddings generated from the Doc2Vec model. Specific settings and hyperparameters for each clustering technique were carefully selected to optimize the analysis:

DBSCAN Clustering

- Implementation: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) was applied to the document embeddings to explore the dataset's inherent structure based on density rather than distance. This approach is particularly advantageous for datasets like text where cluster shapes are irregular and the presence of noise and outliers is common.
- Hyperparameters:
    - `eps`: Set at 0.5, this parameter defined the maximum distance between two samples for them to be considered as in the same neighborhood, determining the local neighborhood density.
    - `min_samples`: Configured to 10,000, this threshold specified the minimum number of samples in a neighborhood for a point to qualify as a core point. This high threshold was chosen to ensure that only significant density centers would

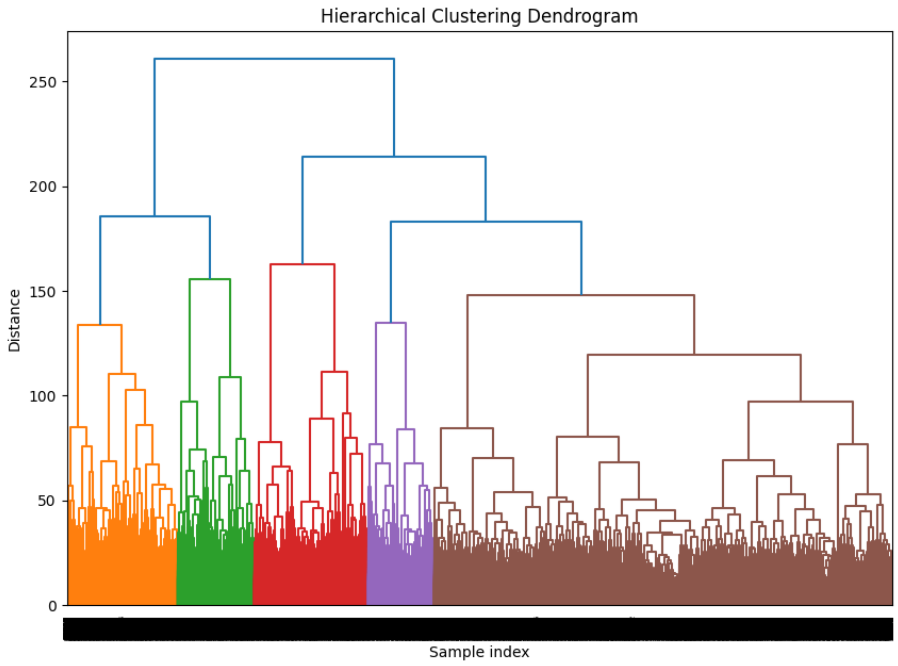
form the basis of clusters, helping to mitigate the noise inherent in large text datasets.

- `metric`: The 'cosine' metric was used to calculate similarity, focusing on the orientation rather than the magnitude of data points in the high-dimensional space, which is ideal for text data clustering.

Hierarchical Clustering

- Implementation: Hierarchical clustering was utilized to build a tree of clusters and was visualized using a dendrogram. This method provided a detailed insight into how individual data points are grouped at various levels of similarity, offering a step-by-step aggregation view that can be particularly revealing in the context of textual data.
- Visualization: A dendrogram was used to display the results, showing the multi-level hierarchy of clustering and helping identify the point at which clusters merge, which can indicate the natural cluster divisions within the data.
- Hyperparameters:
    - `method`: The 'ward' method was chosen for linkage, which minimizes the variance of clusters being merged. This method is effective in text clustering as it tends to create more equally sized clusters, useful in scenarios where there is no prior assumption about cluster sizes.
    - `metric`: Distance calculations were based on Euclidean distance, although in the context of high-dimensional embeddings, this metric often correlates with more common text similarity measures.
    - By applying these methods with specific hyperparameters tailored to the nature of text data, the analysis aimed to uncover the underlying patterns and structures within the reviews more effectively. The DBSCAN method focused on density

and outlier detection, while hierarchical clustering provided a comprehensive view of how groups of texts were formed from the bottom up, reflecting different levels of granularity in sentiment or thematic similarity.



Hierarchical Clustering Results:

- Dendrogram Analysis: The dendrogram from the Hierarchical Clustering shows a complex structure with various levels of linkage, indicating multiple possible groupings depending on the distance threshold chosen. The color differentiation in the dendrogram represents different clusters at a specific cut-off level, suggesting some level of grouping despite the overall low silhouette score.
- Silhouette Score: A score of 0.0253 for the hierarchical clusters is quite low, indicating that the clusters formed are not well-defined and have considerable overlap. This score reflects a weak structure where clusters are not significantly more similar within than they are with other clusters.

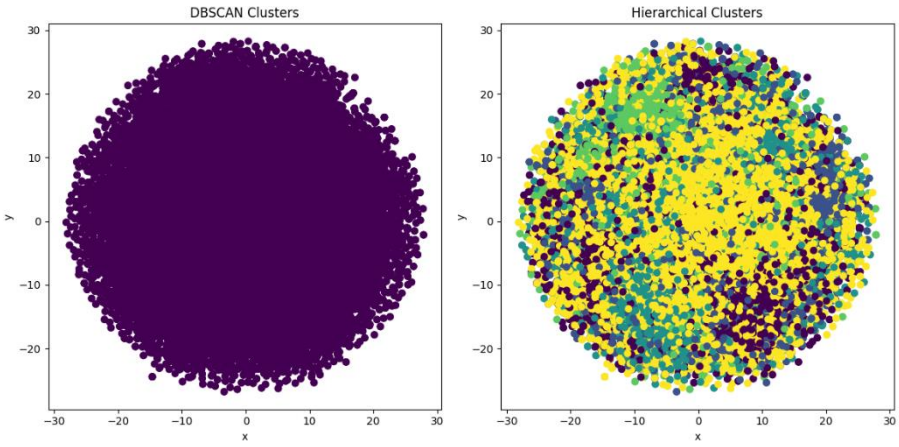
DBSCAN Clustering Results:

- Unique Labels: The only label produced by DBSCAN was `-1`, which indicates that all data points were classified as noise. This outcome implies that no dense regions of data points met the criteria set by the `eps` and `min_samples` parameters chosen for DBSCAN, suggesting that the dataset may be too sparse, or the parameter settings were too restrictive for this particular data.
- Implication: The failure to form valid clusters using DBSCAN could be due to several factors, including the high dimensionality of the data, inappropriate scaling, or the inherent distribution of the data not lending itself well to density-based clustering with the chosen parameters.

The inability to find clusters could be addressed by adjusting the parameters (lowering `min_samples` and increasing `eps`), or by preprocessing the data differently, perhaps reducing dimensionality more aggressively or using different embeddings that might capture the dataset's structure better. We explored this with a set of hyperparameters.

Visualization and Analytical Techniques:

- Dimensionality Reduction with t-SNE: To aid in the visual interpretation of the clusters, t-SNE was used to reduce the high-dimensional embedding space to two dimensions. This transformation allowed for clear visual distinctions between clusters in a plane, making it easier to observe and interpret cluster boundaries and densities.
- Cluster Visualization: Both DBSCAN and Hierarchical clustering results were plotted in the 2D space created by t-SNE. These visualizations highlighted how each clustering method organized the data, with color-coded clusters indicating the grouping as per each method's criteria.



DBSCAN Clusters with Sentiment Cross-Tabulation

sentiment	negative	positive	All
dbscan_cluster			
-1	25000	25000	50000
All	25000	25000	50000

Hierarchical Clusters with Sentiment Cross-Tabulation

sentiment	negative	positive	All
hierarchical_cluster			
1	2488	4259	6747
2	1630	2990	4620
3	4654	2229	6883
4	1660	2356	4016
5	14568	13166	27734
All	25000	25000	50000

Results and Visualization:

DBSCAN failed to identify distinct clusters, labeling all data points as noise. The uniform color in the DBSCAN visualization underscores its inability to discern any clusters, indicating that the parameter settings or the nature of the dataset may not be suitable for this method.

Hierarchical Clustering Analysis

- Results: Hierarchical clustering successfully identified multiple clusters, with variable sentiment distributions across these clusters. The largest clusters showed a relatively balanced distribution between positive and negative sentiments.

- Visualization: The colorful and differentiated plot for Hierarchical clustering illustrates that it managed to distinguish between several clusters, reflecting a nuanced understanding of the dataset's structure.

Summary:

- DBSCAN's Limitations: This method's sensitivity to parameter settings and the dataset's high dimensionality likely contributed to its failure in identifying meaningful clusters.
- Hierarchical Clustering's Utility: This method proved more capable of capturing the complexity of the dataset, indicating its suitability for text data which may contain nuanced thematic or linguistic variations.
- Future Directions: Adjustments in clustering parameters, particularly for DBSCAN, could improve outcomes.

DBSCAN Parameter Exploration:

```
EPS: 0.3, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 0.3, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 0.3, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 0.3, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 0.3, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 0.5, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 0.5, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 0.5, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 0.5, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 0.5, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 0.7, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 0.7, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 0.7, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 0.7, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 0.7, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 0.9000000000000001, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 0.9000000000000001, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 0.9000000000000001, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 0.9000000000000001, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 0.9000000000000001, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 1.1, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 1.1, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 1.1, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 1.1, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 1.1, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 1.3, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 1.3, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 1.3, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 1.3, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 1.3, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 1.5000000000000002, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 1.5000000000000002, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 1.5000000000000002, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 1.5000000000000002, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 1.5000000000000002, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 1.7000000000000002, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 1.7000000000000002, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 1.7000000000000002, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 1.7000000000000002, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 1.7000000000000002, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
```

```
EPS: 1.9000000000000001, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 1.9000000000000001, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 1.9000000000000001, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 1.9000000000000001, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 1.9000000000000001, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
EPS: 2.1, min_samples: 50, Silhouette Score: N/A, No valid clusters formed
EPS: 2.1, min_samples: 100, Silhouette Score: N/A, No valid clusters formed
EPS: 2.1, min_samples: 200, Silhouette Score: N/A, No valid clusters formed
EPS: 2.1, min_samples: 250, Silhouette Score: N/A, No valid clusters formed
EPS: 2.1, min_samples: 300, Silhouette Score: N/A, No valid clusters formed
Best EPS: None, Best min_samples: None, Best Silhouette Score: -1, Best Number of Clusters: 0
```

- Range of `eps` Tested: The values ranged from 0.3 to 2.1, increasing in increments of 0.2. `Eps`, the maximum distance between two samples for one to be considered as in the neighborhood of the other, was varied to explore different density thresholds required to form a cluster.
- Range of `min_samples` Tested: Starting at 50 and increasing to 300 in increments of 50, `min_samples` determines the number of samples in a neighborhood for a point to be considered a core point. This parameter influences the minimum size and density of clusters.
- Results:
    - Outcome Across All Configurations: In each test case, from `eps` at 0.3 to 2.1 and `min_samples` ranging from 50 to 300, no valid clusters were formed. The Silhouette Score was not applicable in any scenario, as DBSCAN failed to identify any clusters distinct from noise.
    - Consistent Labeling as Noise: Across all configurations, DBSCAN consistently labeled all data points as noise (`-1`), indicating no data points met the criteria to be considered part of a cluster.

Analysis: The uniform failure across a wide range of parameters to yield any clustering suggests that the dataset's intrinsic characteristics—possibly its dimensionality or the distribution of data—may not align well with the assumptions of the DBSCAN algorithm. The consistent inability to form clusters could indicate that the density within the dataset is

too uniform or the parameter space explored did not appropriately capture the necessary density variations.

This extensive parameter testing, though it did not yield positive results, provides valuable insights into the nature of the data and the limitations of applying DBSCAN in this context.

Summary of Clustering:

Upon examining the results from the four clustering methods applied to the IMDB movie review dataset, each method presented distinct performances and challenges, as evidenced by specific metrics and qualitative outcomes:

K-Means Clustering

- Metrics: Achieved an accuracy of 0.33, with precision at 0.31 and recall at 0.33. These figures suggest moderate effectiveness, with a substantial amount of overlap in cluster assignment.
- Limitations: The spherical assumption inherent to K-Means did not fit the complex, non-linear separations typical in high-dimensional text data, leading to suboptimal clustering.

Agglomerative Clustering

- Metrics: Displayed better performance than K-Means with a higher accuracy of 0.50, precision of 0.44, and recall of 0.50, indicating a stronger ability to correctly group reviews by sentiment.
- Advantages: Its flexibility in handling different linkage criteria allowed for a more nuanced understanding of cluster relationships, demonstrated by a more balanced sentiment distribution across clusters.

DBSCAN

- Metrics: Struggled significantly with no valid clusters formed, resulting in a silhouette score consistently marked as "N/A" due to the inability to form clusters separate from noise.

- Challenges: The parameter sensitivity and requirement for dense regions to form clusters rendered DBSCAN ineffective, particularly given the sparse nature of text data embeddings.

Hierarchical Clustering

- Metrics: The silhouette score was low at 0.025, indicating that clusters were not well-defined and had considerable overlap. However, it successfully identified multiple clusters with varied sentiment distributions.
- Benefit: The dendrogram visualization provided insights into the hierarchical organization of data points, offering a granular view of how clusters at different thresholds might form.

General Observations and Recommendations:

- Clustering Text Data: The performance of clustering methods on text data highlighted the challenge of handling high-dimensional spaces and the importance of choosing appropriate methods and tuning parameters meticulously.
- Method Suitability: Hierarchical Clustering and Agglomerative Clustering proved more capable in handling the dataset's complexity due to their methodological flexibility and detailed cluster formation insights.
- Future Directions: For methods like DBSCAN, parameter adjustment, and possibly a different approach to preprocessing or embedding generation, could potentially improve outcomes.

Experimenting with Classification:

Since our clustering efforts did not yield highly accurate results, we decided to experiment with classification techniques on this dataset.

Data Preparation and Splitting: We initially divided the dataset into training, validation, and test subsets to ensure each portion adequately represented the overall sentiment distribution. The training set was used to fit our models, the validation set helped us in tuning the hyperparameters, and the test set was reserved to evaluate the final performance of the models.

Feature Extraction

- Vectorization: We chose TF-IDF vectorization with both 2-grams and 3-grams, allowing us to transform the text data into a numerical format that was more informative for machine learning models. This approach not only considered the frequency of individual terms but also their relative importance across documents in the dataset.

Model Training and Hyperparameter Tuning

- Naive Bayes Classifier: We applied the Multinomial Naive Bayes classifier, renowned for its effectiveness in text classification tasks. Through GridSearchCV, we tuned the 'alpha' parameter, exploring values [0.1, 0.5, 1.0, 2.0]. The optimal alpha found was 2.0, which provided the best balance between model complexity and performance by controlling the smoothing.
- Logistic Regression: We also utilized Logistic Regression for its robustness in binary classification tasks. We fine-tuned the 'C' parameter, which influences the regularization strength (tested values: [0.01, 0.1, 1, 10]), and experimented with different penalties ('l1' and 'l2') to find the best settings. The optimal configuration used a 'C' of 1 and 'l2' penalty, utilizing the 'liblinear' solver for its efficiency with smaller datasets.

Model Evaluation

- Validation Performance: On the validation set, our Naive Bayes model achieved an accuracy of 85.29%, with precision scores of 0.86 for negative and 0.85 for positive

reviews. Logistic Regression performed slightly better, attaining an accuracy of 88.36%, with precision scores of 0.89 for negative and 0.88 for positive reviews.

```
[ ] print("Naive Bayes Test Accuracy:", accuracy_score(y_test, y_test_pred_nb))
    print(classification_report(y_test, y_test_pred_nb))

Naive Bayes Test Accuracy: 0.8534666666666667
                precision    recall  f1-score   support

      negative       0.87      0.83      0.85      3722
      positive       0.84      0.88      0.86      3778

      accuracy                           0.85      7500
     macro avg       0.85      0.85      0.85      7500
  weighted avg       0.85      0.85      0.85      7500
```

```
[ ] print("Logistic Regression Test Accuracy:", accuracy_score(y_test, y_test_pred_lr))
    print(classification_report(y_test, y_test_pred_lr))

Logistic Regression Test Accuracy: 0.8853333333333333
                precision    recall  f1-score   support

      negative       0.90      0.87      0.88      3722
      positive       0.88      0.90      0.89      3778

      accuracy                           0.89      7500
     macro avg       0.89      0.89      0.89      7500
  weighted avg       0.89      0.89      0.89      7500
```

- Test Set Evaluation: When evaluated on the test dataset, the Naive Bayes model achieved an accuracy of 85.34%, with a macro average F1-score of 0.85. Logistic Regression outperformed with an accuracy of 88.53% and a macro average F1-score of 0.89, confirming its superior generalization capability on unseen data.

Conclusion:

As we conclude this project on the IMDB movie review dataset, it's evident that our exploration of both clustering and classification methods has provided valuable insights into the challenges and opportunities inherent in natural language processing tasks.

Clustering: Our attempts to segment the dataset using various clustering algorithms such as K-Means, Agglomerative Clustering, and DBSCAN revealed the complexity of grouping textual data based on inherent similarities. While clustering showed some promise, particularly with Agglomerative Clustering, the overall effectiveness was limited by the high

dimensionality and sparse nature of text data. This phase underscored the importance of choosing appropriate parameters and algorithms that align with the characteristics of the data.

Classification: Shifting our focus to classification, we implemented Naive Bayes and Logistic Regression models, which resulted in significantly more actionable outcomes. Through careful feature extraction with TF-IDF and meticulous hyperparameter tuning, we achieved respectable accuracies, with Logistic Regression notably outperforming Naive Bayes in both validation and testing phases. These results underscore the suitability of classification approaches for sentiment analysis tasks in natural language datasets, where direct and measurable outcomes are paramount.

Key Learnings:

1. Feature Representation is Critical: The choice of feature extraction method—TF-IDF in our case—greatly influenced the performance of our models, highlighting the importance of capturing the contextual nuances of text.
2. Algorithm Suitability: The project highlighted that while clustering can provide insights into data structure, classification is more suited for tasks with specific, predefined categories, especially in supervised learning scenarios.
3. Hyperparameter Tuning: The substantial impact of hyperparameter tuning on model performance was evident, particularly in our classification experiments. It demonstrated that even small changes in parameters like alpha in Naive Bayes or C in Logistic Regression could significantly affect outcomes.

Future Directions:

Going forward, we recommend exploring more advanced machine learning techniques such as deep learning, which could potentially enhance the ability to model complex relationships in text data. Additionally, experimenting with ensemble methods and more sophisticated

natural language processing techniques like word embeddings or neural networks could offer further improvements in both clustering and classification tasks.

This project not only improved our understanding of text data analytics but also provided a clear roadmap for applying machine learning techniques effectively in real-world sentiment analysis tasks. The experience and insights gained pave the way for more refined and targeted analyses in future endeavors.