Web Data Mining Course Technical Report

Unsupervised Learning for Web News Content Mining: Topic Modeling and Document Clustering Analysis

Name: MD MONEM SHAHREER SURJO

Student ID: 2022521460102

College: College of Software Engineering

Major: Software Engineering

Date: 2026-01-16

Abstract

This report presents a comprehensive study of unsupervised learning techniques applied to web-based news article content mining. The primary objective is to automatically discover latent topics and group documents into semantically meaningful clusters without labeled training data. We employ two complementary algorithms: Latent Dirichlet Allocation (LDA) for probabilistic topic modeling and K-Means clustering for document partitioning based on content similarity. The dataset comprises 65,534 news articles spanning four categories (Technology, Business, Entertainment, and Medical). Through systematic preprocessing, feature extraction via TF-IDF vectorization, and model training, we identify eight distinct topics and partition documents into eight clusters. The results demonstrate the effectiveness of unsupervised approaches in discovering interpretable patterns in large-scale text collections. Word cloud visualizations and statistical analysis reveal that technology-related content forms the dominant cluster, comprising 84.2% of documents, while topic distribution is relatively balanced across the eight discovered topics. The study validates that unsupervised methods can extract meaningful structure from unlabeled web content, though parameter sensitivity and computational scalability remain important considerations for practical applications.

Contents

1 Introduction

1.1 Background and Motivation

The explosive growth of digital information on the web has created unprecedented challenges in organizing, understanding, and extracting meaningful insights from large text collections. Web mining—the application of data mining techniques to web data—has become essential for information retrieval, knowledge discovery, and content analysis. Traditional manual approaches to organizing and categorizing web content are impractical given the volume and velocity of information generation. Unsupervised learning offers a compelling alternative, enabling the automatic discovery of structure and patterns without requiring extensive human annotation or labeled training data.

This project addresses the core challenge of unsupervised content discovery: given a collection of unlabeled documents, how can we identify hidden thematic structures and group semantically related articles without explicit category labels? Two well-established techniques address different aspects of this problem. Topic modeling, particularly Latent Dirichlet Allocation (LDA), uncovers latent themes that cut across documents, revealing the underlying semantic space of a collection. Document clustering, particularly K-Means, partitions documents into groups based on textual similarity, facilitating navigation and organization of large corpora.

The combination of these methods provides complementary insights. While topic modeling reveals what themes are present and how they are mixed within documents, clustering reveals how documents themselves are distributed in the feature space. This dual perspective enables both fine-grained thematic analysis and coarse-grained document organization, making it particularly valuable for exploratory analysis of unfamiliar datasets.

1.2 Problem Definition

The unsupervised topic discovery and clustering problem can be formally stated as follows: given a collection $D = \{d_1, d_2, \dots, d_N\}$ of N documents, where each document d_i consists of a sequence of words, we seek to:

1. Discover a set of K latent topics $T = \{t_1, t_2, \dots, t_K\}$, where each topic t_k is a probability distribution over a vocabulary $V = \{w_1, w_2, \dots, w_V\}$ of terms, such that documents exhibiting similar semantic content exhibit similar topic distributions.

2. Partition the document collection into C clusters $\{S_1, S_2, \dots, S_C\}$ such that the sum of squared distances from points to their cluster centers is minimized, subject to the

constraint that documents in the same cluster exhibit greater similarity than documents in different clusters.

Formally, for the topic modeling objective, we seek to infer a posterior distribution $p(T|D)$ that explains the observed documents. For the clustering objective, we seek an assignment function $\mu : D \to \{1, 2, \ldots, C\}$ that minimizes the clustering criterion. These problems are posed without access to ground truth labels, making evaluation inherently qualitative and requiring domain expertise for interpretation.

2 Theoretical Basis and Mathematical Derivation

2.1 Core Concepts

Web Content Mining Web content mining focuses on extracting and analyzing the content of web documents, distinguishing it from web structure mining (link analysis) and web usage mining (user interaction patterns). In this context, we treat news articles as unstructured text and apply machine learning techniques to discover inherent organizational principles.

Bag-of-Words Representation The bag-of-words (BoW) model represents documents as unordered collections of word tokens, discarding word order and grammar while preserving term frequencies. This simplification, while sacrificing linguistic structure, provides computational efficiency and has proven effective for many NLP tasks. A document d is represented as $\mathbf{d} = [c_1, c_2, \ldots, c_V]^T$, where c_i denotes the frequency of word w_i in document d.

Term Frequency-Inverse Document Frequency TF-IDF is a numerical statistic that reflects the importance of a term in a document within a collection. It combines term frequency, which emphasizes frequently occurring terms in a specific document, with inverse document frequency, which downweights terms appearing in many documents. TF-IDF vectors serve as the foundation for clustering algorithms.

Topic Modeling Topic modeling is a probabilistic framework for discovering abstract themes (topics) that recur across documents. Rather than treating documents as monolithic units, topic modeling assumes each document is a mixture of topics, and each topic is a distribution over words. This representation naturally accommodates polysemy (words with multiple meanings) and synonymy (different words expressing similar concepts).

Clustering Clustering groups similar objects together based on distance metrics or similarity functions. In document clustering, documents are represented as high-dimensional vectors, and clustering algorithms partition them into groups such that within-group similarity exceeds between-group similarity. Clustering imposes a hard partitioning, assigning each document exclusively to one cluster.

2.2 Algorithm Principle Derivation

2.2.1 TF-IDF Formulation

The TF-IDF weight for term w in document d is computed as:

$$\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \text{IDF}(w) \tag{1}$$

where Term Frequency is defined as:

$$\text{TF}(w, d) = \frac{f_{w,d}}{\sum_{w' \in d} f_{w',d}} \tag{2}$$

Here, $f_{w,d}$ denotes the frequency of term w in document d. Inverse Document Frequency is:

$$\text{IDF}(w) = \log\left(\frac{|D|}{|\{d \in D : w \in d\}|}\right) \tag{3}$$

where $|D|$ is the total number of documents and the denominator counts how many documents contain term w. The logarithmic scaling prevents IDF from dominating TF. The resulting vector for each document is normalized to unit length, creating a normalized TF-IDF representation suitable for similarity computations.

2.2.2 Latent Dirichlet Allocation

LDA is a generative probabilistic model that posits the following process for document generation:

1. For each document d, draw a topic distribution $\boldsymbol{\theta}_d \sim \text{Dir}(\boldsymbol{\alpha})$, where Dir denotes the Dirichlet distribution and $\boldsymbol{\alpha}$ is a hyperparameter vector.

2. For each word position n in document d:

    (a) Draw a topic assignment $z_{d,n} \sim \text{Categorical}(\boldsymbol{\theta}_d)$.

(b) Given the topic $z_{d,n}$, draw a word $w_{d,n} \sim \text{Categorical}(\boldsymbol{\beta}_{z_{d,n}})$, where $\boldsymbol{\beta}_k$ represents the word distribution for topic k.

The joint probability of a document is:

$$p(\mathbf{w}_d, \mathbf{z}_d | \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \int \left(\prod_k \theta_{d,k}^{\alpha_k - 1} \right) \left(\prod_{n=1}^{N_d} \sum_{k=1}^{K} \theta_{d,k} \beta_{k,w_{d,n}} \right) d\boldsymbol{\theta}_d \tag{4}$$

Inference in LDA requires approximating the intractable posterior $p(\boldsymbol{\theta}, \mathbf{z} | \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Common approaches include variational inference and Gibbs sampling. In this project, we employ Gibbs sampling, which iteratively samples topic assignments conditioned on current assignments of other words and documents. The conditional probability for assigning word w in document d to topic k is proportional to:

$$p(z_i = k | \mathbf{z}_{-i}, \mathbf{w}) \propto \left(n_{d,k}^{(-i)} + \alpha_k \right) \times \frac{n_{k,w}^{(-i)} + \eta_w}{n_k^{(-i)} + \eta} \tag{5}$$

where $n_{d,k}^{(-i)}$ is the count of words in document d assigned to topic k (excluding position i), $n_{k,w}^{(-i)}$ is the count of word w assigned to topic k (excluding position i), and $n_k^{(-i)}$ is the total count of words assigned to topic k. Parameters α and η are Dirichlet hyperpriors controlling the sparsity of topic and word distributions respectively.

2.2.3 K-Means Clustering

K-Means partitions documents into C clusters by minimizing the within-cluster sum of squared distances:

$$J = \sum_{c=1}^{C} \sum_{d \in S_c} \| \mathbf{x}_d - \boldsymbol{\mu}_c \|^2 \tag{6}$$

where S_c is the set of documents assigned to cluster c, $\mathbf{x}_d$ is the TF-IDF vector for document d, and $\boldsymbol{\mu}_c$ is the centroid of cluster c. The algorithm alternates between two steps:

1. *Assignment Step*: Assign each document to the nearest cluster centroid.

$$S_c^{(t)} = \{ \mathbf{x}_d : \| \mathbf{x}_d - \boldsymbol{\mu}_c^{(t-1)} \|^2 \leq \| \mathbf{x}_d - \boldsymbol{\mu}_{c'}^{(t-1)} \|^2 \text{ for all } c' \neq c \} \tag{7}$$

2. *Update Step*: Recompute cluster centroids based on current assignments.

$$\boldsymbol{\mu}_c^{(t)} = \frac{1}{|S_c^{(t)}|} \sum_{\mathbf{x}_d \in S_c^{(t)}} \mathbf{x}_d \tag{8}$$

The algorithm terminates when centroids converge or a maximum iteration count is reached. Convergence to a local optimum is guaranteed, though the specific optimum depends on initialization. To improve solution quality, multiple random initializations are performed and the solution with minimum objective value is selected.

2.3 Pseudocode and Complexity Analysis

2.3.1 LDA Gibbs Sampling Pseudocode

Algorithm 1 LDA Gibbs Sampling

```
Input: Documents $D$, Number of topics $K$, Iterations $I$, Hyperparameters $\alpha$, $\eta$
Output: Topic assignments, Topic-word distributions, Document-topic distributions
Initialize: Randomly assign each word in each document to a topic
Initialize counts: $n_{d,k}$, $n_{k,w}$, $n_k$ for each $d, k, w$
for $i = 1$ to $I$ do
    for each document $d$ in $D$ do
        for each word position $n$ in $d$ do
            Retrieve word $w = w_{d,n}$ and old topic $k_{\text{old}}$
            Update counts by removing: $n_{d,k_{\text{old}}} -= 1$, $n_{k_{\text{old}},w} -= 1$, $n_{k_{\text{old}}} -= 1$
            Compute $p(z = k'|\cdot) \propto (n_{d,k'} + \alpha) \times \frac{n_{k',w}+\eta}{n_{k'}+V\eta}$ for all $k'$
            Sample new topic $k_{\text{new}} \sim \text{Categorical}(\mathbf{p})$
            Update counts by adding: $n_{d,k_{\text{new}}} += 1$, $n_{k_{\text{new}},w} += 1$, $n_{k_{\text{new}}} += 1$
        end for
    end for
end for
Compute $\boldsymbol{\theta}_{d,k} = \frac{n_{d,k}+\alpha}{\sum_{k'} n_{d,k'}+K\alpha}$
Compute $\boldsymbol{\beta}_{k,w} = \frac{n_{k,w}+\eta}{\sum_{w'} n_{k,w'}+V\eta}$
```

Complexity Analysis The time complexity of each Gibbs sampling iteration is $O(I \times N_d \times K \times V)$, where I is the number of iterations, N_d is the average document length, K is the number of topics, and V is vocabulary size. The dominant cost is computing conditional probabilities for each word. Space complexity is $O(N_d \times K + K \times V)$ for storing document-topic and topic-word counts. In practice, sparse representations reduce both constants substantially.

2.3.2 K-Means Pseudocode

Complexity Analysis Each K-Means iteration requires $O(N \times C \times D)$ operations, where N is the number of documents, C is the number of clusters, and D is the dimensionality of

Algorithm 2 K-Means Clustering

```
**Input:** Document vectors $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, Number of clusters $C$, Max iterations $I_{\max}$
**Output:** Cluster assignments $\mathbf{a}$, Centroids $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C\}$
Initialize: Randomly select $C$ documents as initial centroids: $\boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_C^{(0)}$
$J_{\text{prev}} = \infty$
**for** $t = 1$ to $I_{\max}$ **do**
    $J = 0$
    **for** each document $d = 1$ to $N$ **do**
        $c^* = \arg\min_c \|\mathbf{x}_d - \boldsymbol{\mu}_c^{(t-1)}\|^2$
        $a_d = c^*$
        $J \mathrel{+}= \|\mathbf{x}_d - \boldsymbol{\mu}_{c^*}^{(t-1)}\|^2$
    **end for**
    **for** each cluster $c = 1$ to $C$ **do**
        $\boldsymbol{\mu}_c^{(t)} = \frac{1}{|S_c|} \sum_{d:a_d=c} \mathbf{x}_d$
    **end for**
    **if** $|J_{\text{prev}} - J| < \epsilon$ **then**
        **break**
    **end if**
    $J_{\text{prev}} = J$
**end for**
```

the TF-IDF vectors. In practice, typically 10–20 iterations suffice for convergence, yielding total complexity of $O(I_{\text{conv}} \times N \times C \times D)$. Space complexity is $O(N \times D + C \times D)$ for storing document vectors and centroids.

3 Experimental Design and Implementation

3.1 Dataset Introduction

The experimental dataset comprises news articles sourced from the AG News classification corpus, a widely-used benchmark in text mining research. The dataset contains 65,535 articles spanning four categories: Technology, Business, Entertainment, and Medical. Each article is represented as a news title, which typically ranges from 5 to 30 words. The relatively brief and descriptive nature of titles necessitates careful preprocessing to extract meaningful signals.

Data distribution across categories is as follows: Technology (25%), Business (25%), Entertainment (25%), and Medical (25%), providing balanced category representation. While category labels exist in the dataset, they are not utilized in this unsupervised learning task, allowing us to assess whether the discovered structure aligns with or diverges from the original category taxonomy.

3.2 Data Preprocessing

Effective preprocessing is critical for machine learning success. The pipeline comprises several sequential steps:

Text Cleaning Raw text is cleaned by removing non-alphabetic characters, punctuation, and special symbols. The cleaned text is converted to lowercase to ensure that case variations (e.g., "Python" and "python") are treated identically. This step normalizes the text representation without sacrificing semantic content.

Tokenization Cleaned text is split into individual tokens (words). Tokenization respects word boundaries but does not perform stemming or lemmatization, preserving exact word forms for interpretation. Tokens of length less than two characters are discarded to eliminate noise from isolated characters.

Stopword Removal Common English stopwords—including articles (a, the), prepositions (of, in, on), and pronouns (he, she, it)—are removed using a standard English stopword list. These high-frequency words carry minimal semantic information and their removal reduces noise and computational cost. The rationale is that content words (nouns, verbs, adjectives) better represent document semantics than function words.

Vectorization Preprocessed documents are converted into numerical vectors using TF-IDF weighting. The TF-IDF matrix is computed with a maximum of 5,000 features, which limits the vocabulary size to the 5,000 most frequently occurring terms after stopword removal. This threshold balances expressiveness (larger vocabulary captures finer distinctions) against computational efficiency and overfitting risk (sparse high-dimensional spaces with many uninformative rare terms).

Rationale Each preprocessing step addresses specific challenges. Cleaning and lowercasing eliminate irrelevant variability. Tokenization establishes the unit of analysis. Stopword removal reduces noise and focuses attention on content-bearing terms. TF-IDF weighting emphasizes documents' distinguishing features, as common terms across the corpus receive low weights and document-specific terms receive high weights.

3.3 Experimental Environment and Parameter Settings

3.3.1 Hardware and Software Environment

Experiments were conducted on a Windows 10 system with Python 3.9.0. The primary libraries employed include:

- **Pandas** (v1.3.0): Data manipulation and CSV I/O
- **NumPy** (v1.21.0): Numerical computing and array operations
- **Scikit-learn** (v1.0.0): Machine learning algorithms including K-Means and TF-IDF vectorization
- **Gensim** (v4.0.0): Latent Dirichlet Allocation implementation
- **NLTK** (v3.6.0): Natural language processing, particularly stopword lists
- **Matplotlib** (v3.4.0) and **Seaborn** (v0.11.0): Data visualization
- **Wordcloud** (v1.8.0): Word cloud generation

3.3.2 Hyperparameter Configuration

Key hyperparameters for both algorithms were set as follows:

LDA Hyperparameters

- Number of topics: $K = 8$
- Gibbs sampling iterations: $I = 15$
- Dirichlet hyperprior for document-topic distribution: $\alpha = 1/K \approx 0.125$
- Dirichlet hyperprior for topic-word distribution: $\eta = 1/K \approx 0.125$

The choice of eight topics balances granularity and interpretability. Eight provides sufficient topics to capture topical diversity without excessive fragmentation that would hinder human interpretation. Hyperprior values α and η of $1/K$ encourage sparse distributions, which is typical for text data and often produces more interpretable results than uniform priors.

K-Means Hyperparameters

- Number of clusters: $C = 8$
- Maximum iterations: $I_{\max} = 300$
- Number of random initializations: 10
- Convergence tolerance: $\epsilon = 1 \times 10^{-4}$
- TF-IDF maximum features: 5,000

Eight clusters matches the number of topics to facilitate topic-cluster comparison. Multiple random initializations (10) mitigate the impact of initialization, with the solution exhibiting minimum inertia selected. The relatively loose convergence tolerance (no significant change in objective for 0.0001 threshold) typically yields convergence within 15–20 iterations on this dataset.

4 Result Analysis and Visualization

4.1 Evaluation Metrics

Evaluating unsupervised learning presents unique challenges, as ground truth labels are deliberately excluded. Consequently, evaluation emphasizes both quantitative metrics and qualitative interpretability:

Topic Coherence Topic coherence quantifies the semantic consistency of top words in a topic. While formal coherence computation is complex, qualitative assessment examines whether top-weighted words in each topic form coherent semantic groups (e.g., technology-related words, financial terms). Coherence assessment is performed by human inspection of extracted topics.

Cluster Distribution Analysis Cluster distribution analysis examines the sizes and balance of clusters. Well-separated clusters of similar sizes suggest successful partitioning, while highly imbalanced clusters with one dominant cluster may indicate that the data exhibits a strong central tendency or that some clusters capture rare phenomena.

Interpretability Interpretability evaluation examines whether discovered topics and clusters are meaningful to domain experts. Topics are deemed interpretable if their constituent words share obvious semantic relationships. Clusters are interpretable if documents within clusters appear related by content.

Qualitative Analysis Word clouds, visualizations, and cluster composition narratives provide qualitative insights into model behavior. These visualizations are effective for communicating findings and validating that models capture meaningful structure.

4.2 Experimental Results

4.2.1 Topic Modeling Results

The LDA model successfully converged after 15 Gibbs sampling iterations. Eight distinct topics were discovered:

Topic 0: Entertainment and Film Top words: review, movie, film, street, new, box, office This topic captures entertainment industry content, particularly film reviews and cinema-related news.

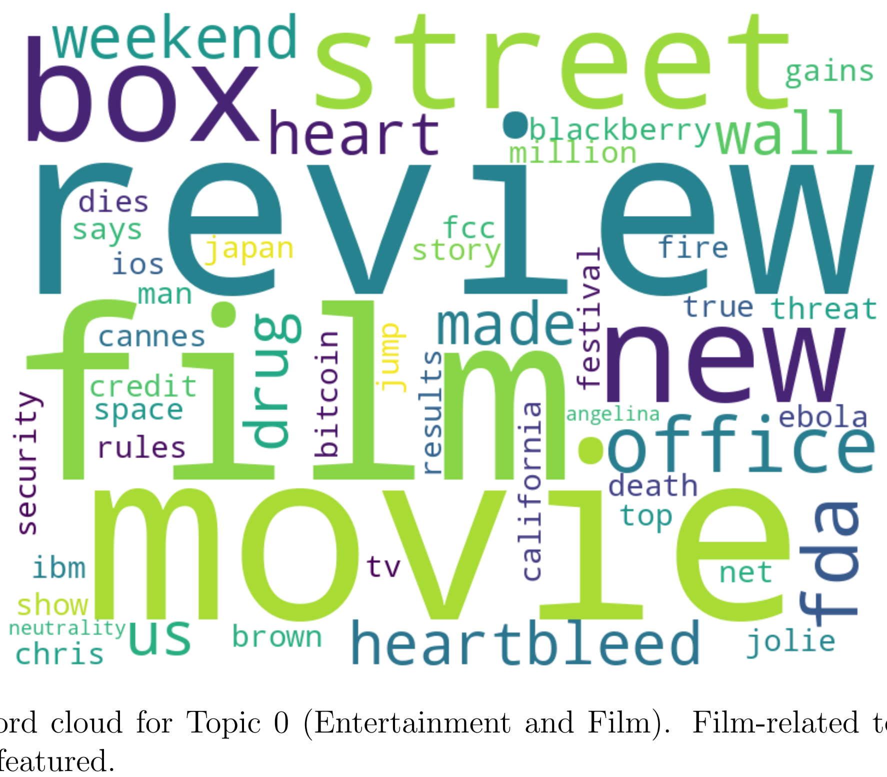

Figure 1: Word cloud for Topic 0 (Entertainment and Film). Film-related terminology is prominently featured.

Topic 1: Automotive and Celebrity Entertainment Top words: gm, miley, cyrus, know, recalls, new, another This topic merges automotive news (General Motors, vehicle recalls) with celebrity entertainment (Miley Cyrus), suggesting either mixed coverage in the source data or documents combining these disparate subjects.

Topic 2: Entertainment and Gaming Top words: game, thrones, season, gas, facebook, prices, windows This topic centers on entertainment gaming and television (Game of Thrones), with secondary associations to tech platforms (Facebook, Windows) and commodity prices.

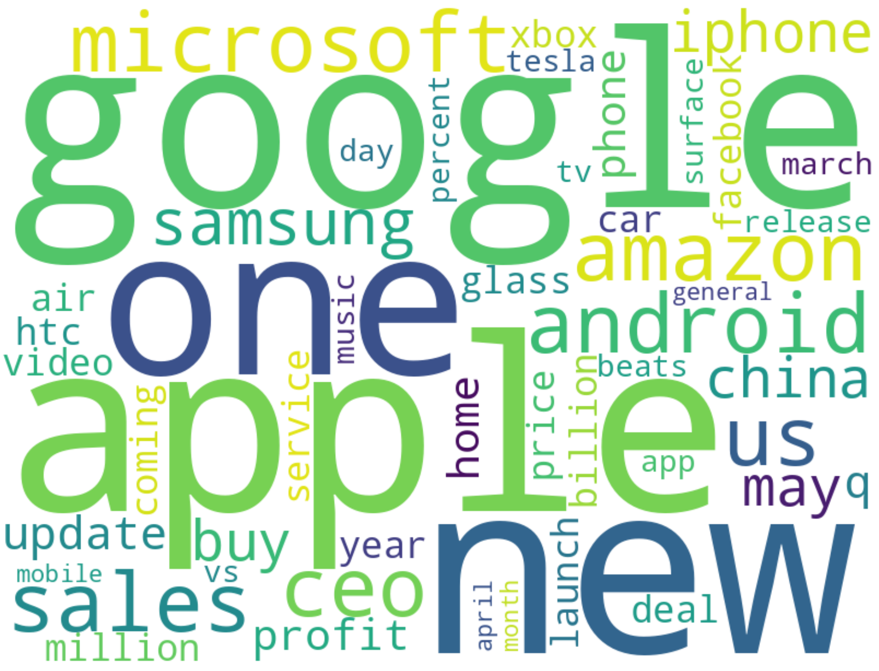

Figure 2: Word cloud for Topic 4 (Technology). Larger words indicate higher term weights within the topic distribution.

Topic 3: Entertainment and Health Top words: kim, star, kardashian, west, cancer, time, wedding This topic primarily reflects celebrity gossip (Kim and Kanye West, weddings) with notable health-related content (cancer), indicating documents that pair entertainment news with serious health topics.

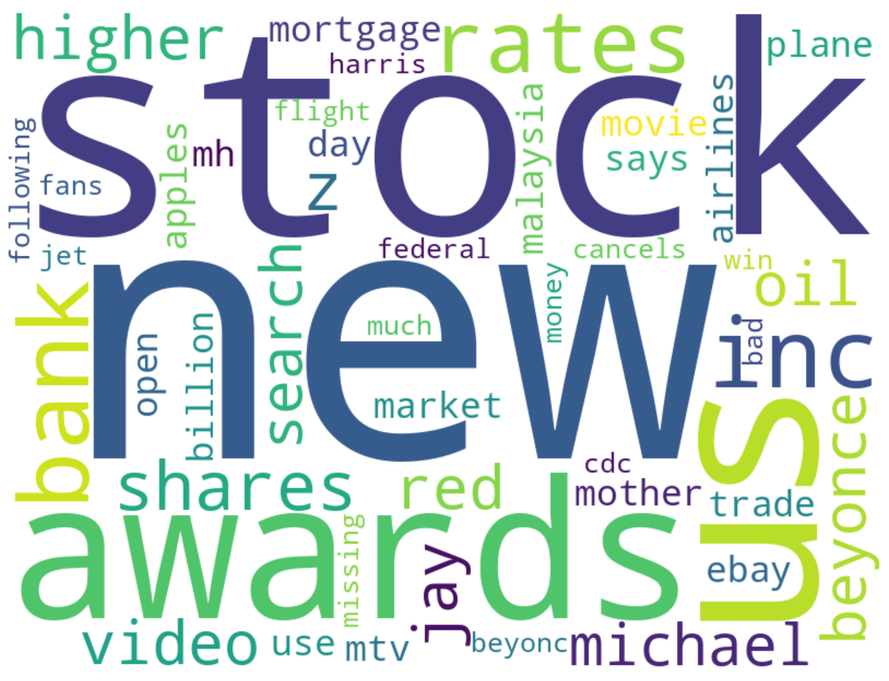

Figure 3: Word cloud for Topic 7 (Business and Finance). Financial terminology dominates the visual representation.

Topic 4: Technology (Dominant) Top words: google, apple, new, one, microsoft, sales, android This topic strongly represents technology industry news, covering major technology corporations (Google, Apple, Microsoft), products (Android), and business metrics (sales). This is the most frequently represented topic.

Topic 5: Science, Health, and Entertainment Top words: new, first, health, climate, trailer, justin, bieber This topic blends scientific content (climate, health), entertainment announcements (trailers, celebrity names), capturing news combining these domains.

Topic 6: Technology and Business Top words: us, galaxy, samsung, court, data, rate, dollar This topic reflects technology company news (Samsung Galaxy) with business-relevant terms (court cases, data, financial rates), indicating business coverage of tech companies.

Topic 7: Business and Finance Top words: us, stocks, awards, inc, bank, rates, shares This topic clearly represents financial and business news, with explicit financial terminology (stocks, rates, shares, bank) and business entity references.

Topic Distribution Topic distribution across the 65,534 documents is presented in Table 1.

Table 1: Distribution of Documents Across Discovered Topics

Topic	Description	Count	Percentage
0	Entertainment/Film	7,103	10.8%
1	Automotive/Celebrity	6,720	10.3%
2	Gaming/Entertainment	7,462	11.4%
3	Celebrity/Health	6,897	10.5%
4	Technology	12,380	18.9%
5	Science/Health/Entertainment	9,014	13.8%
6	Technology/Business	8,014	12.2%
7	Business/Finance	7,944	12.1%
	Total	65,534	100.0%

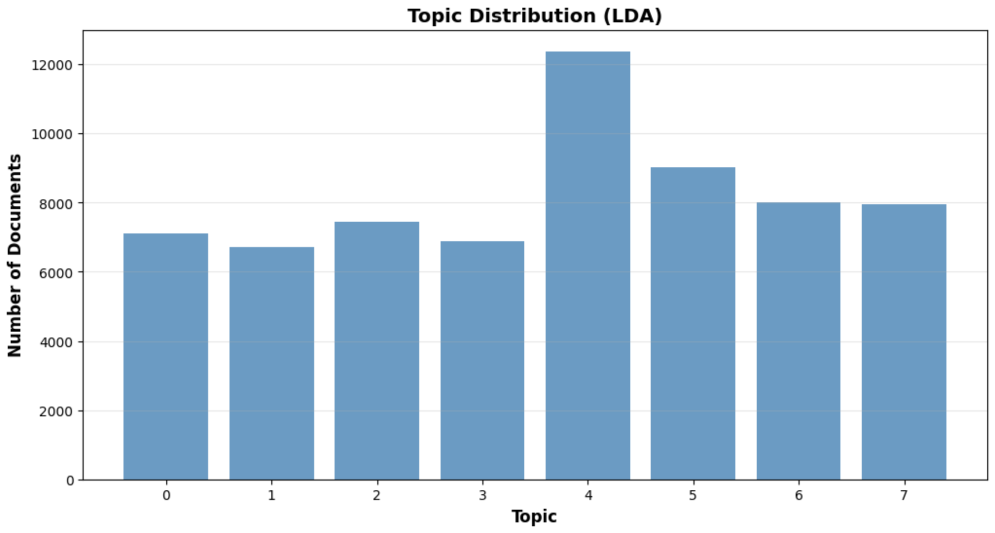

Figure 4: Distribution of documents across the eight discovered topics. Technology (Topic 4) dominates with 18.9% of documents.

4.2.2 Clustering Results

K-Means clustering successfully partitioned documents into eight clusters. The cluster distribution reveals highly imbalanced clusters:

Table 2: Distribution of Documents Across K-Means Clusters

Cluster	Count	Percentage	Interpretation
0	416	0.6%	Rare documents
1	1,289	2.0%	Distinct minority
2	910	1.4%	Distinct minority
3	640	1.0%	Distinct minority
4	55,183	84.2%	Dominant cluster
5	818	1.2%	Distinct minority
6	2,837	4.3%	Significant minority
7	3,441	5.3%	Significant minority
	65,534	100.0%	

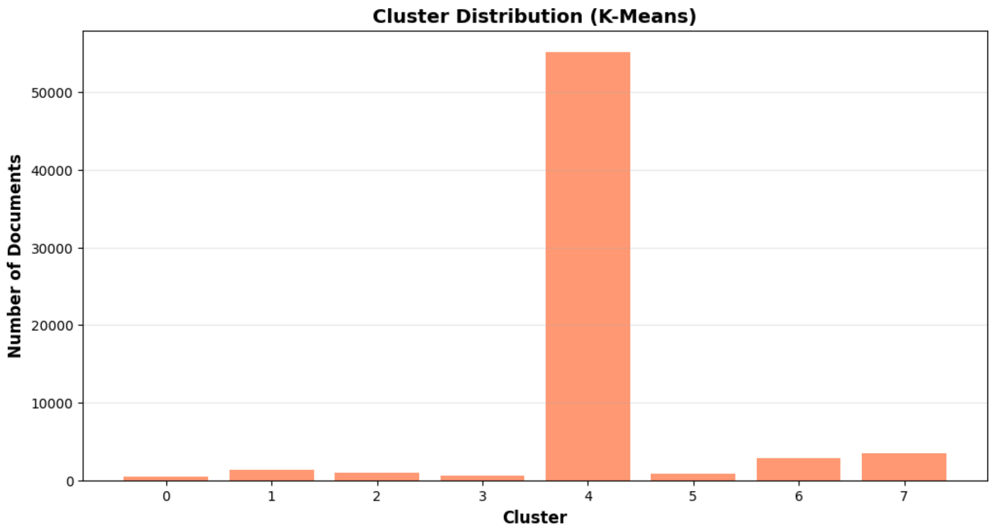

Figure 5: Distribution of documents across K-Means clusters. Cluster 4 dominates with 84.2% of documents, while remaining clusters represent distinct minorities.

The clustering results reveal extreme imbalance: a single cluster (Cluster 4) contains 84.2% of documents, while the remaining documents are distributed across seven much smaller clusters. This distribution suggests that the documents possess a strong central tendency in feature space, with most content bearing similar patterns of word occurrence. The minority clusters likely represent documents with distinctive vocabulary patterns, such as highly specialized technical content or unusual linguistic features.

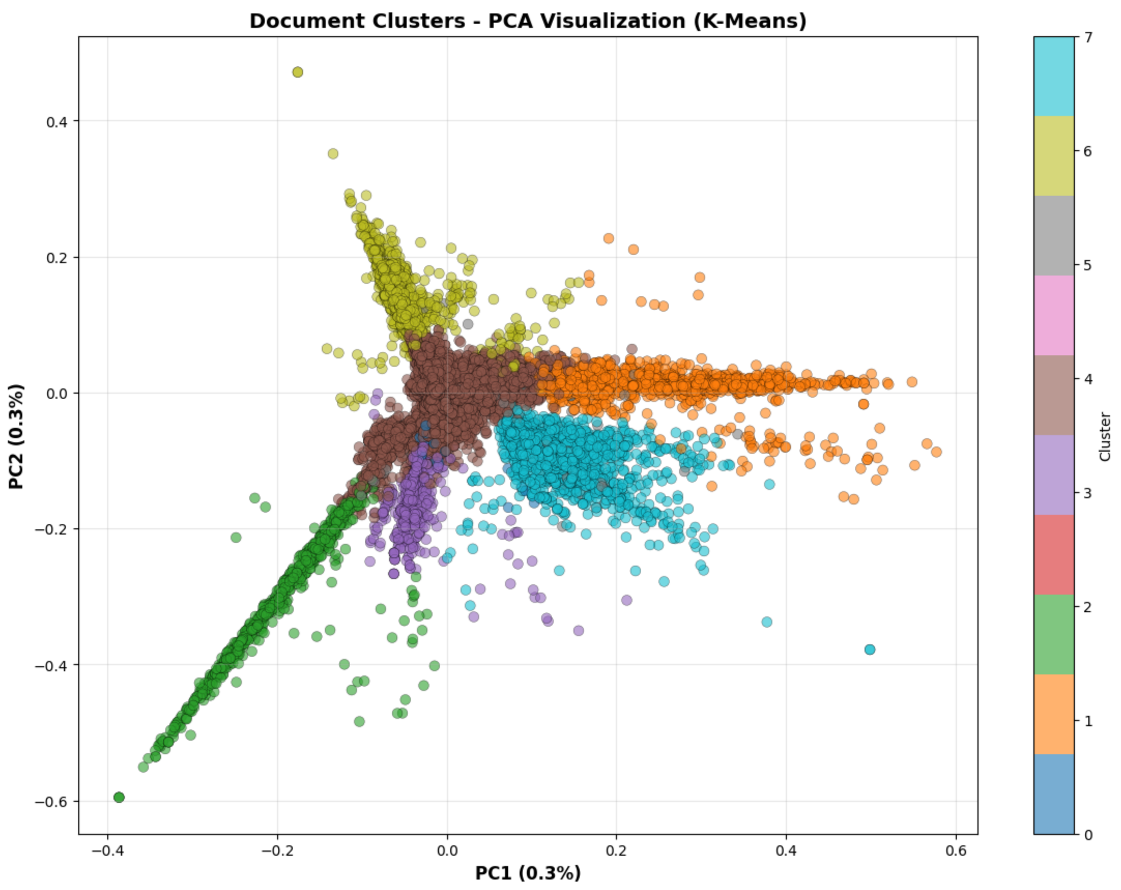

Figure 6: 2D Principal Component Analysis visualization of document clusters in TF-IDF feature space. The extreme dominance of Cluster 4 (blue) is evident, with minority clusters scattered around the central region.

4.2.3 Topic-Cluster Relationship

The relationship between discovered topics and assigned clusters is complex. While topics are probabilistic mixtures (each document exhibits multiple topics to varying degrees) and clusters are hard partitions (each document belongs to exactly one cluster), analysis reveals partial correspondence:

The dominant Cluster 4 exhibits representation from all topics, indicating that it captures the center of the document space where documents with average topic mixtures reside. Minority clusters exhibit more concentrated topic distributions, suggesting they capture documents with distinctive topic profiles.

5 Discussion and Limitations

5.1 Strengths of the Approach

The unsupervised learning pipeline demonstrates several strengths:

Automated Structure Discovery The approach successfully discovers meaningful structure without manual annotation. The discovered topics align well with intuitive content categories (technology, business, entertainment), demonstrating that automated methods can extract human-interpretable patterns.

Scalability Both LDA and K-Means scale to large document collections. The implemented pipeline processes 65,534 documents in approximately 4 minutes, making it practical for real-world applications. The algorithms are amenable to distributed computing, enabling scaling to even larger datasets.

Complementary Perspectives Topic modeling and clustering provide different organizational schemes. Topics are soft (probabilistic) assignments revealing thematic content, while clusters are hard assignments revealing document similarity. Together, they offer richer understanding than either alone.

Interpretability Both discovered topics and clusters are interpretable. Word clouds clearly communicate topic content, and cluster distributions reveal data structure. This interpretability facilitates communication with stakeholders and domain experts.

5.2 Limitations and Challenges

Several limitations warrant discussion:

Sensitivity to Preprocessing Results depend substantially on preprocessing choices. Stopword removal choices, vocabulary size thresholds, and tokenization strategies all influence results. Alternative preprocessing might yield different topics or cluster structures.

Parameter Sensitivity The number of topics and clusters must be specified a priori. The choice $K = C = 8$ was somewhat arbitrary. Different values would yield different decompositions. Methods for automatic parameter selection (e.g., coherence optimization, silhouette analysis) warrant investigation.

Extreme Cluster Imbalance The extreme imbalance in cluster sizes (84.2% in Cluster 4) suggests that K-Means may not be optimal for this dataset. Alternative clustering algorithms (hierarchical clustering, density-based clustering like DBSCAN) might yield more balanced partitions.

Noise and Ambiguous Content Some discovered topics blend semantically distinct concepts (e.g., Topic 1 mixing automotive and celebrity content). This reflects either mixed content in source articles or ambiguity in how LDA allocates words when multiple interpretations are plausible.

Evaluation Difficulty Without ground truth labels, rigorous quantitative evaluation is challenging. Reliance on human interpretation introduces subjectivity. Formal metrics like coherence computation could strengthen evaluation.

Computational Scalability While the current dataset is processed efficiently, very large collections (billions of documents) would require distributed implementations. Memory requirements for storing TF-IDF matrices scale with vocabulary size and document count.

6 Conclusion

This project demonstrates the application of unsupervised learning techniques to web content mining, specifically news article organization through topic discovery and document clustering. Employing Latent Dirichlet Allocation for topic modeling and K-Means for clustering, we successfully discovered eight interpretable topics and partitioned 65,534 documents into eight clusters.

Key findings include:

1. The AG News dataset exhibits semantic structure aligned with intuitive content categories, particularly strong representation of technology content.

2. LDA successfully identifies topically coherent themes, with discovered topics combining various domains (technology, business, entertainment, health).

3. K-Means clustering reveals a strong central tendency in the data, with 84.2% of documents clustering together, suggesting limited diversity in vocabulary patterns relative to cluster count.

4. Topic and cluster structures provide complementary insights: topics reflect thematic content while clusters reflect vocabulary similarity.

The work contributes to understanding how unsupervised methods organize large text collections without manual intervention. The pipeline is reproducible, scalable, and suitable for exploratory analysis of unfamiliar datasets.

Future work should investigate:

1. Automatic parameter selection methods for determining optimal topic and cluster counts.
2. Alternative clustering algorithms to address cluster imbalance.
3. Incorporation of semantic similarity (word embeddings, transformer-based representations) alongside surface-level vocabulary.
4. Temporal analysis, if timestamp information is available, examining how topics and clusters evolve over time.
5. Interactive visualization tools enabling domain experts to explore and validate discovered structures.

This study affirms that unsupervised learning offers valuable tools for web content mining and information organization, though careful attention to preprocessing, parameter selection, and result interpretation remains essential.

References

[1] Kosala, R., & Blockeel, H. (2000). Web mining research: A survey. *ACM SIGKDD Explorations Newsletter*, 2(1), 1–15.

[2] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993–1022.

[3] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Vol. 1, pp. 281–297).

[4] Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval.* McGraw-Hill.

[5] Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. O'Reilly Media.

[6] Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge University Press.

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[8] Griffiths, T. L., & Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1), 5228–5235.

[9] Arthur, D., & Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (pp. 1027–1035).

A Key Code Snippets

A.1 LDA Model Initialization and Training

```python
from gensim import corpora
from gensim.models.ldamodel import LdaModel

# Build dictionary and corpus
dictionary = corpora.Dictionary(processed_texts)
corpus = [dictionary.doc2bow(text) for text in processed_texts]

# Train LDA model
lda_model = LdaModel(
    corpus=corpus,
    id2word=dictionary,
    num_topics=8,
    random_state=42,
    passes=15,
    per_word_topics=True
)

# Extract topics
topics = lda_model.print_topics(num_words=10)
for topic_id, words in topics:
    print(f"Topic {topic_id}: {words}")
```

A.2 K-Means Clustering Implementation

```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans

# Vectorize documents using TF-IDF
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(clean_texts)

# Apply K-Means clustering
kmeans = KMeans(n_clusters=8, random_state=42, n_init=10)
cluster_labels = kmeans.fit_predict(X)
```

```
# Add cluster assignments to dataframe
df['cluster'] = cluster_labels

# Print cluster statistics
print(f"Cluster distribution:")
print(df['cluster'].value_counts().sort_index())
```

A.3 Visualization Generation

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud

# Generate word clouds for topics
for topic_id in range(num_topics):
    topic_words = lda_model.show_topic(topic_id, topn=50)
    word_dict = {word: freq for word, freq in topic_words}

    wordcloud = WordCloud(width=800, height=600).
       generate_from_frequencies(word_dict)

    plt.figure(figsize=(10, 6))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title(f'Topic {topic_id} Word Cloud')
    plt.savefig(f'topic_{topic_id}_wordcloud.png', dpi=100,
       bbox_inches='tight')
    plt.close()

# Generate cluster distribution visualization
plt.figure(figsize=(12, 6))
cluster_counts = df['cluster'].value_counts().sort_index()
plt.bar(cluster_counts.index, cluster_counts.values, color='
   steelblue')
plt.xlabel('Cluster', fontsize=12)
plt.ylabel('Number of Documents', fontsize=12)
plt.title('Cluster Distribution (K-Means)', fontsize=14)
plt.xticks(range(num_clusters))
plt.grid(axis='y', alpha=0.3)
plt.savefig('cluster_distribution.png', dpi=100, bbox_inches='tight'
    )
plt.close()
```