EAST WEST UNIVERSITY

CSE477

Section: 02

Lab: 05 Report

Topic: Clustering YouTube Comments — A Case Study Approach

Submitted By:

Name: Md Sifatullah Sheikh

ID: 2022-1-60-029

Submitted To:

Amit Mandal

Lecturer

Department of Computer Science & Engineering

Date: 9 August 2025

Lab Time Questions?

To make DBSCAN look as close as possible to your K-Means result (with k from the elbow), you should (a) run DBSCAN in the **same feature space** as K-Means, (b) **pick eps using a k-distance plot**, (c) **tune min_samples** in a small range, and (d) **select the (eps, min_samples) pair that maximizes agreement with K-Means** while yielding ≈ the same number of clusters.

1) Use the same space as K-Means

- If you ran K-Means on **TruncatedSVD(50)** of TF-IDF with **Euclidean** distance, do the same for DBSCAN.
- If you prefer **cosine** for text, either:
  - Run DBSCAN directly on TF-IDF with `metric="cosine"`; or
  - Still reduce with SVD (50–100 dims) and use `metric="euclidean"`. Pick one and keep it consistent with K-Means.

2) Choose a starting `min_samples`

- Common, effective choices: **5, 10, 15**.
- Rule of thumb: around **D+1** (D = reduced dimension) is conservative; for text, 5–15 usually works.

3) Build a k-distance plot to propose eps

- For each point, find the distance to its **min_samples-th** nearest neighbor.
- Sort those distances; the **elbow** of this curve is a good eps.
- Turn several candidate eps values from that curve

	min_samples	eps	score	ARI	NMI	n_clusters	noise_rate
11	15	0.996071	-1.101	-1	-1	1	0.01
7	10	0.996071	-1.101	-1	-1	1	0.01
0	5	0.904748	-1.102	-1	-1	1	0.02
1	5	0.913817	-1.102	-1	-1	1	0.02
3	5	0.961250	-1.102	-1	-1	1	0.02
2	5	0.928337	-1.102	-1	-1	1	0.02
5	10	0.961043	-1.102	-1	-1	1	0.02
4	10	0.944516	-1.102	-1	-1	1	0.02
6	10	0.970555	-1.102	-1	-1	1	0.02
8	15	0.961043	-1.102	-1	-1	1	0.02

4) Grid-search eps × min_samples to match K-Means

- For each pair, run DBSCAN and compute:
  - `n_clusters` (excluding noise)
  - **ARI/NMI** against the K-Means labels (compare **only non-noise points**)
  - noise rate (fraction of `-1`)
- Pick the pair with the **highest ARI/NMI**, **cluster count close to k**, and **reasonable noise** (e.g., <30%).

Abstract

This report presents an end-to-end unsupervised analysis of YouTube comments using TF–IDF features with two clustering algorithms: K-Means and DBSCAN. I evaluate cluster quality via elbow and internal metrics, visualize cluster structure, inspect top terms and representative comments per cluster, and compare the two methods. The study concludes with reflections on dataset characteristics and practical takeaways for real-world feedback analysis.

1. Introduction

The objective of this lab is to discover meaningful discussion themes in YouTube comments without relying on labels. K-Means partitions the data into a fixed number of clusters (k), while DBSCAN groups points by density and identifies noise. These complementary views help assess both dominant topics and outliers such as spam or off-topic remarks.

2. Data and Environment Setup

• Data file: /content/drive/MyDrive/CSE477/cleaned_comments.csv

• Expected columns: cleaned_tokens (list-like) or cleaned_text (string).

• Notebook environment: Google Colab (Drive mounted).
Code installs (pandas, numpy, matplotlib, seaborn, scikit-learn).

3. Step 1 — Data Validation and Initial Hypothesis

I loaded the dataset and reported its shape. A quick preview verified the presence of the expected text column. Dataset size (number of comments) impacts cluster stability and interpretability.

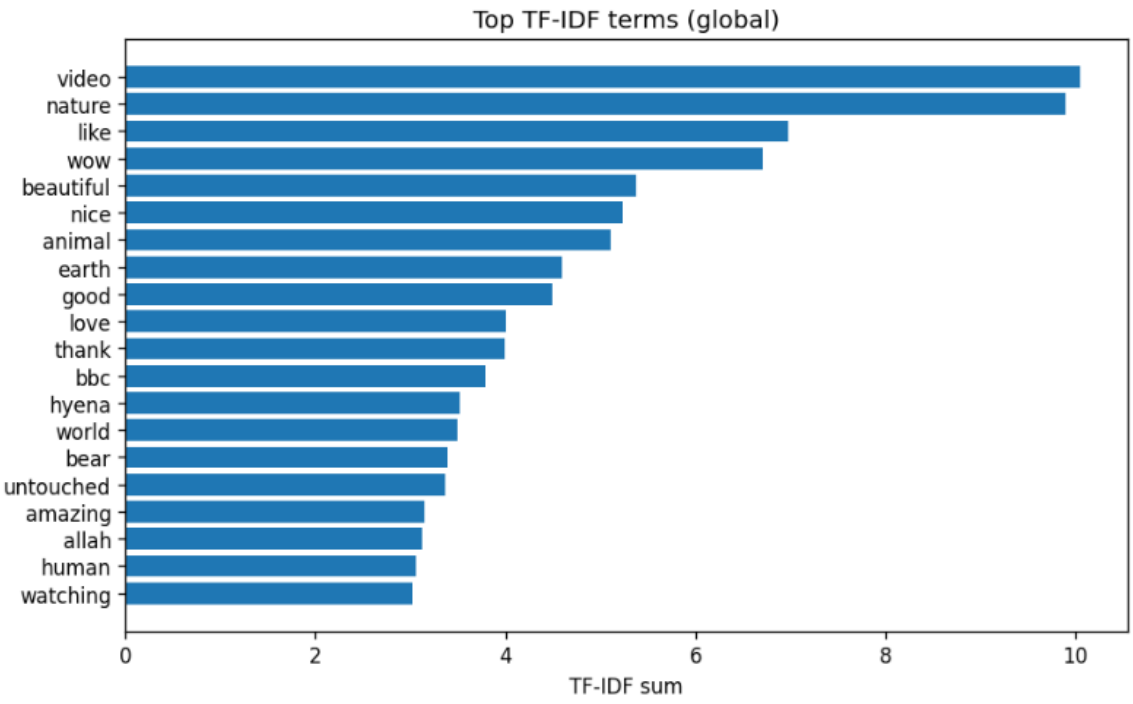
My dataset contains 200 comments and 4 columns.

	username	timestamp_text	comment_text	cleaned_tokens
0	@afzalhossen8454	Unknown	Subhan allah	['subhan', 'allah']
1	@AnimalPrimeHub	Unknown	15:05 My heart absolutely broke for that young...	['heart', 'absolutely', 'broke', 'young', 'mot...
2	@PreemKaSagar	Unknown	Hm ghaar maan thay blaak bicho nay kata hmin t...	['ghaar', 'maan', 'thay', 'blaak', 'bicho', 'n...
3	@Hasnainraza-fm7jp	Unknown	The two truths ▶ sound like sleep and good luck...	['two', 'truth', 'sound', 'like', 'sleep', 'go...
4	@Nimazz_z	Unknown	😊♥	[]

Columns: ['username', 'timestamp_text', 'comment_text', 'cleaned_tokens']

4. Step 2 — Text to Vector (TF–IDF)

I constructed a text corpus from cleaned_tokens or cleaned_text and computed a TF–IDF matrix with unigrams and bigrams. This representation captures important words and short phrases while limiting feature size.



5. Step 3 — K-Means Clustering and Interpretation

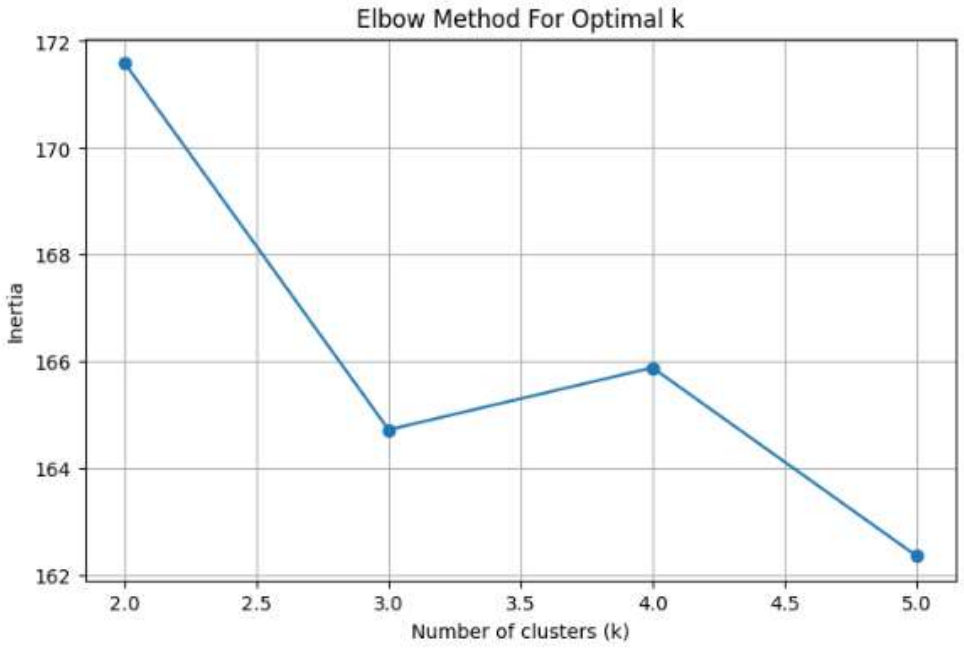
5.1 Elbow and Internal Metrics

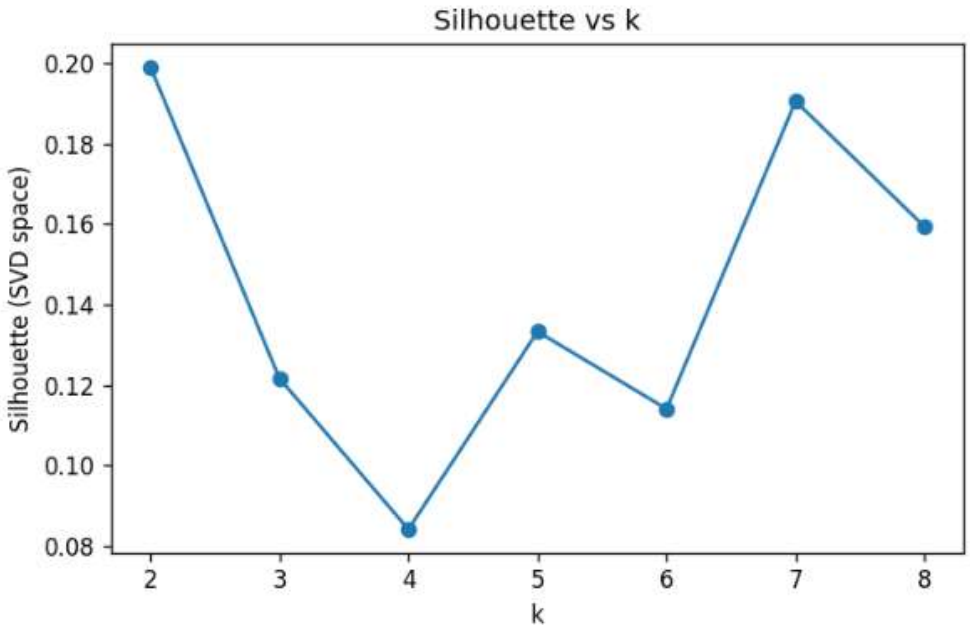
To choose k, I plotted inertia (elbow method) and computed internal validation metrics: Silhouette (higher is better), Calinski–Harabasz (higher is better), and Davies–Bouldin (lower is better). When the elbow is ambiguous, these metrics help triangulate a sensible k.

	k	inertia	silhouette (SVD)	calinski_harabasz (SVD)	davies_bouldin (SVD)
0	2	150.089363	0.199133	10.441302	1.067064
1	3	146.985020	0.121681	8.276469	2.739650
2	4	144.211747	0.084115	7.390208	1.655327
3	5	142.099068	0.133302	6.582851	1.243239
4	6	139.625409	0.114079	6.424243	2.869122
5	7	136.308338	0.190573	6.702123	1.145634
6	8	134.054979	0.159349	6.405714	1.259693

5.2 Final k, Clusters, and Top Terms

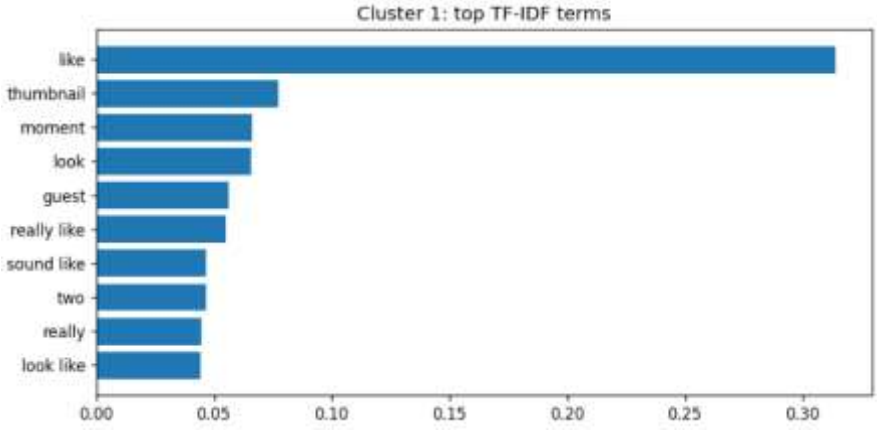
I selected k = [fill value] based on the elbow and supporting metrics. After fitting K-Means, I extracted the top TF–IDF terms for each cluster and reviewed representative comments.

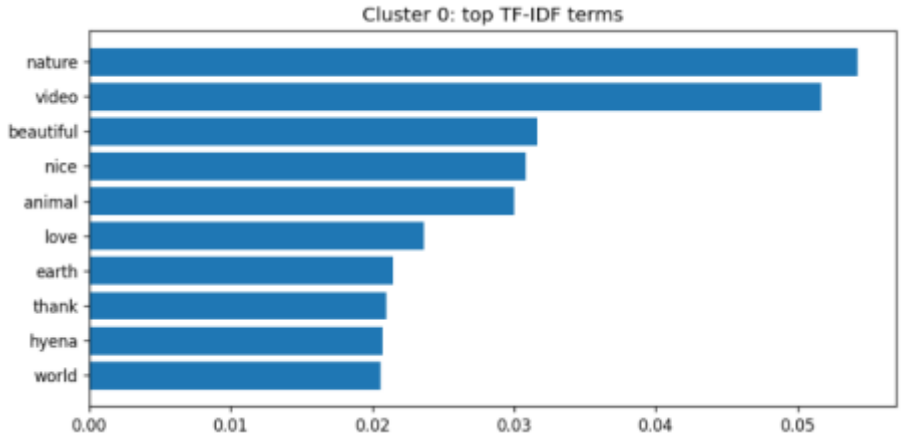


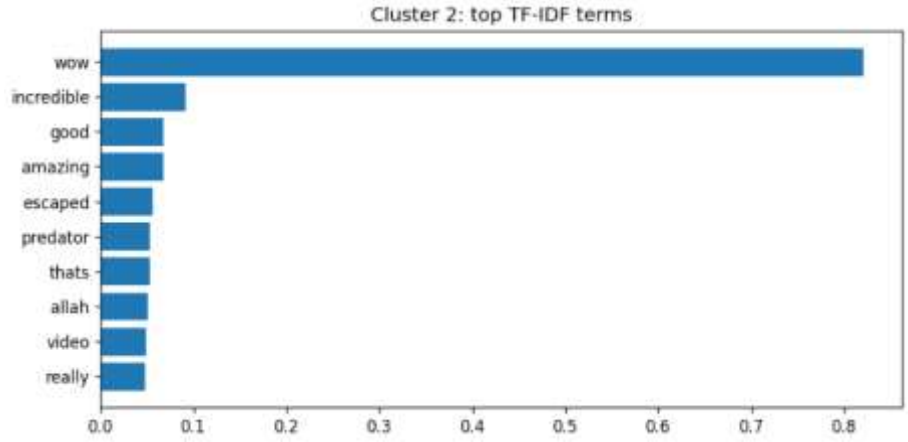


5.3 2D Visualization

To visualize structure, I projected TF–IDF to 2D (TruncatedSVD). The scatter plot shows how comments group under K-Means labels.



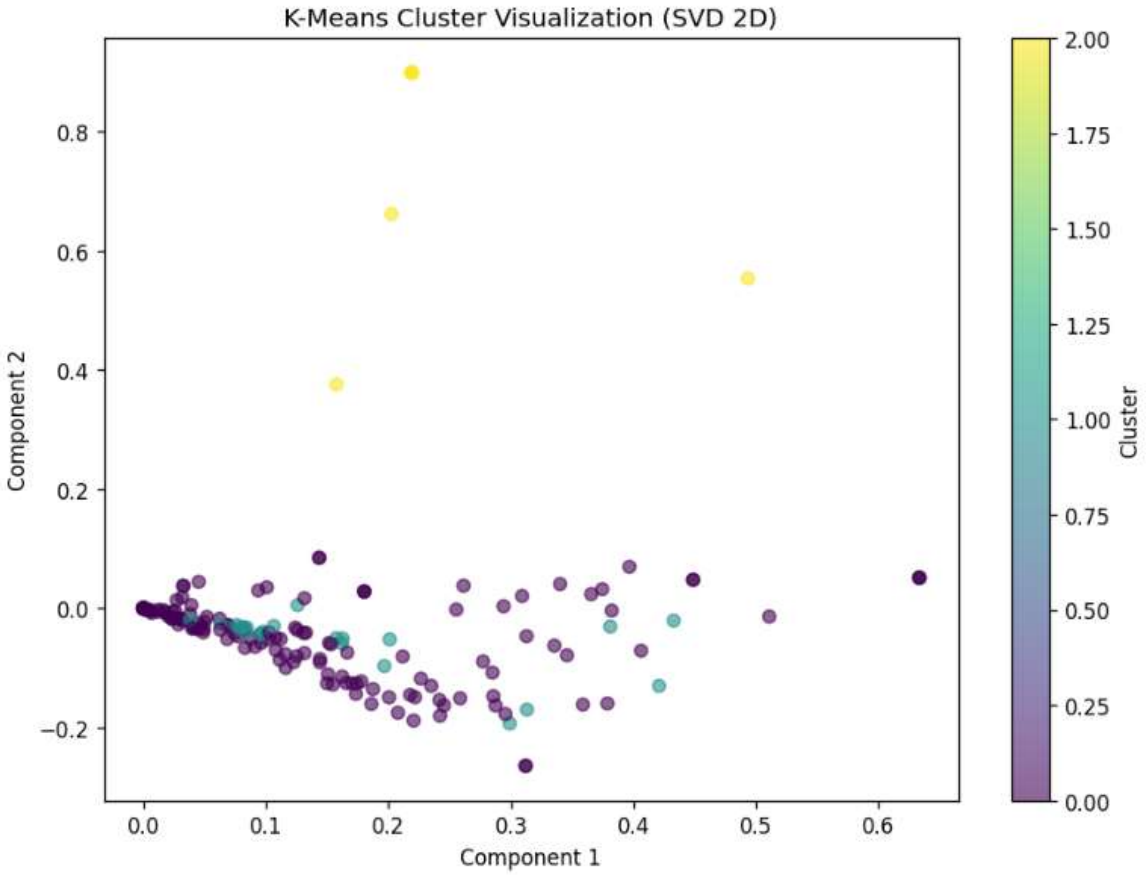




6. Step 4 — DBSCAN (Comparative Analysis)

6.1 Scaling and Parameter Tuning

I scaled the sparse TF–IDF matrix using StandardScaler(with_mean=False) and swept multiple eps values with min_samples=5. I recorded the number of clusters, noise points, and (when applicable) the silhouette score to pick a reasonable eps.



6.2 Final DBSCAN Result and Visualization

With the chosen eps, I re-ran DBSCAN and visualized cluster assignments on the same 2D projection used for K-Means. Label -1 indicates noise points.

	eps	min_samples	clusters	noise_points	silhouette (SVD)
0	0.300000	5	2	155	0.006019
1	0.500000	5	2	155	0.006019
2	0.700000	5	2	155	0.006019
3	0.900000	5	2	155	0.006019
4	1.200000	5	2	155	0.006019

Cluster size comparison:

• K-Means counts: [paste printed counts]

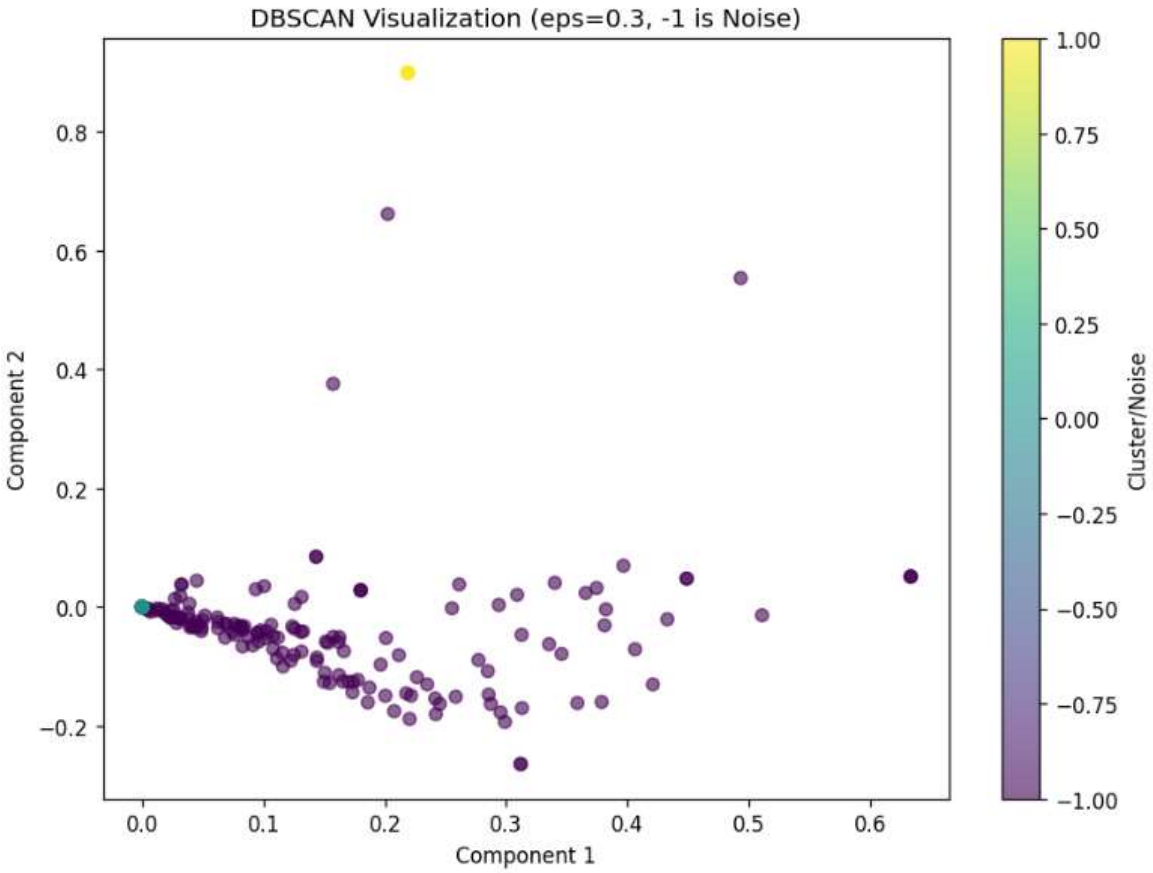
• DBSCAN counts: [paste printed counts]

7. Final Reflection and Application

I reflect on how dataset characteristics (size, topic, language style, code-mixed Bangla/English, slang, emojis) influenced the results. I identify limitations (e.g., small sample size, short comments, sarcasm) and suggest directions (domain-specific stopwords, phrase modeling).

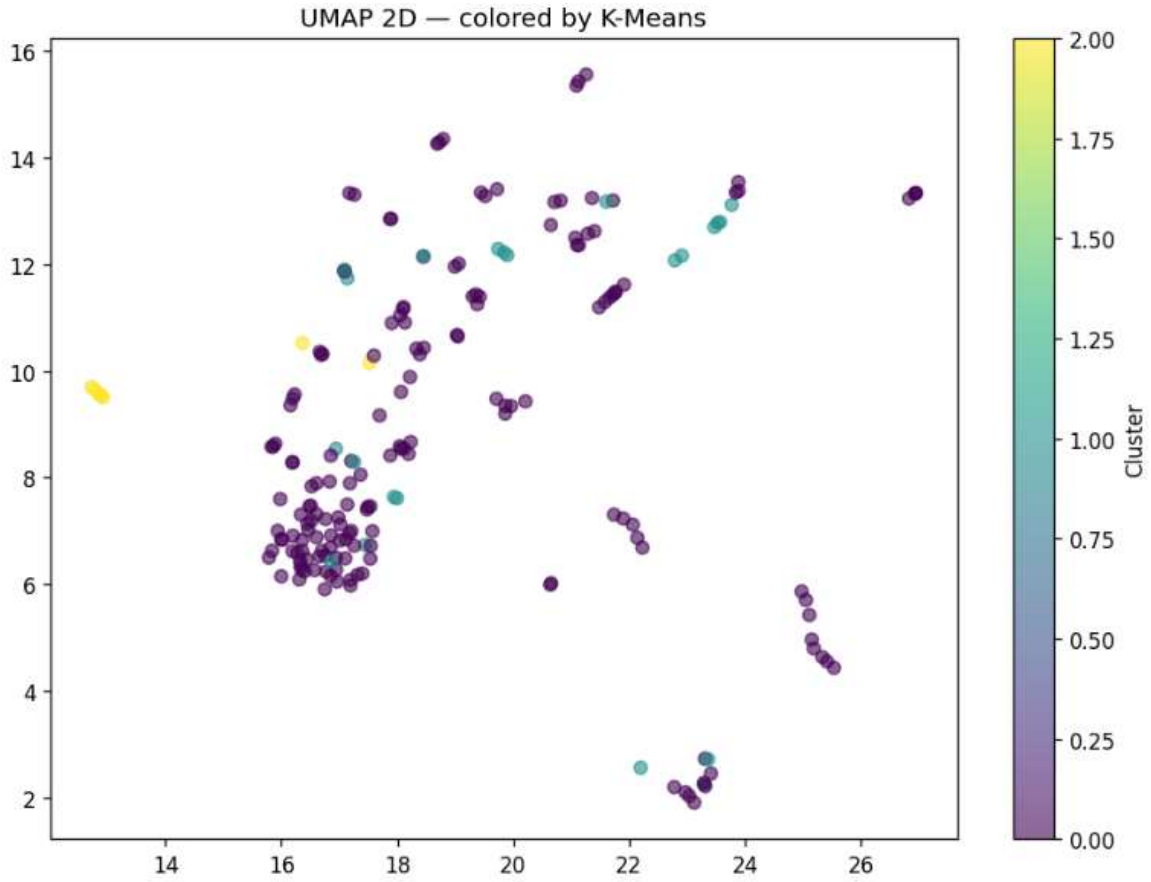
8. Exported Artifacts

I exported a CSV containing the original text and cluster labels for both K-Means and DBSCAN. This artifact supports downstream analysis (e.g., manual labeling, topic summaries).



9. Optional: UMAP 2D Visualization

If UMAP was available, I would have generated an alternative 2D projection that sometimes separates clusters more cleanly. This is presented as a supplementary view.



Appendix A — Reproducibility Notes

• Random seeds fixed where applicable (random_state=42). • Ensure the same TF–IDF parameters and data path when re-running. • If columns differ from expectations, adjust corpus construction accordingly.