DS 203: Final Project Presentation

Pradhicksan C M

Third Year In Engineering Physics

Swayam Saroj Patel

Third Year In Engineering Physics

Aaditya Gupta

Third Year in Electrical Engineering

Contents

Problem Statement

Given a collection of student-submitted lecture summaries that have lost their session-wise labels, perform the following tasks:

1. Cluster the summaries based on content to recover session associations.

2. Rank the summaries within each session by their detail and relevance.

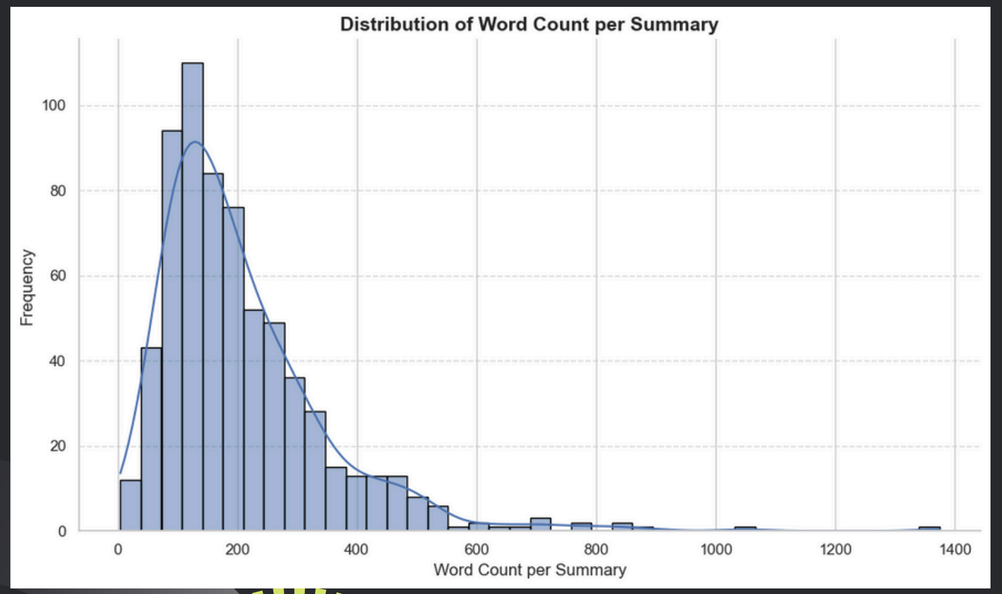
3. Build keyword-based visualizations, such as word clouds and session-level keyword importance.

4. Develop a simple application that takes user-provided keywords and retrieves the most relevant session along with its top summaries.

01. Exploratory Data Analysis

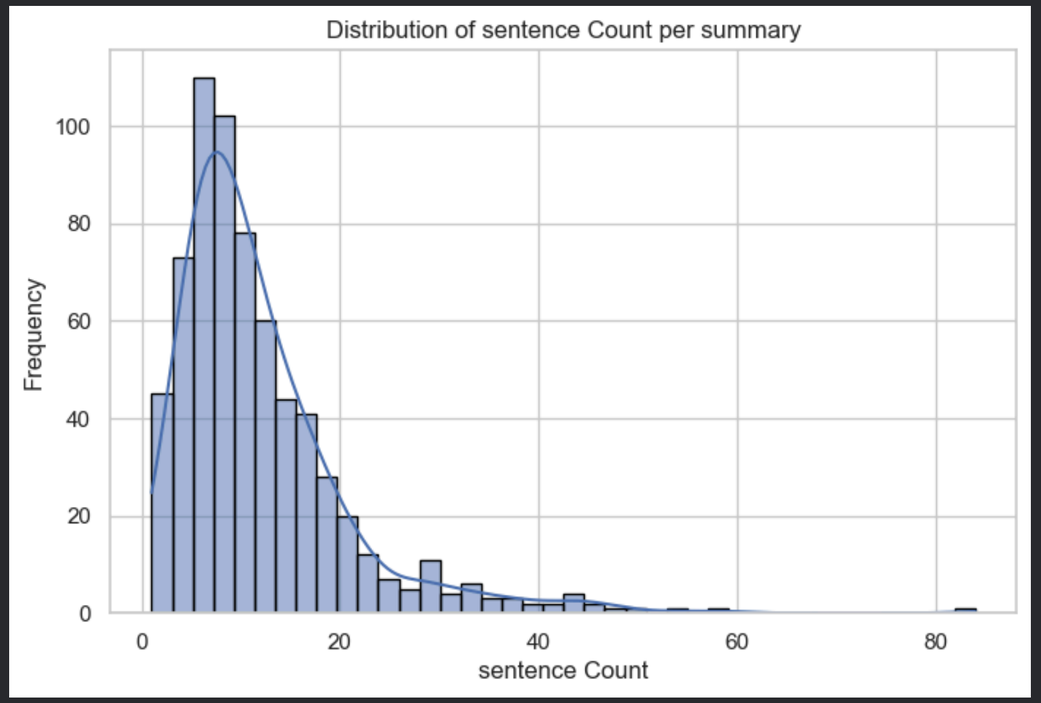
Initial Insights about the Data

Word Count Statistics



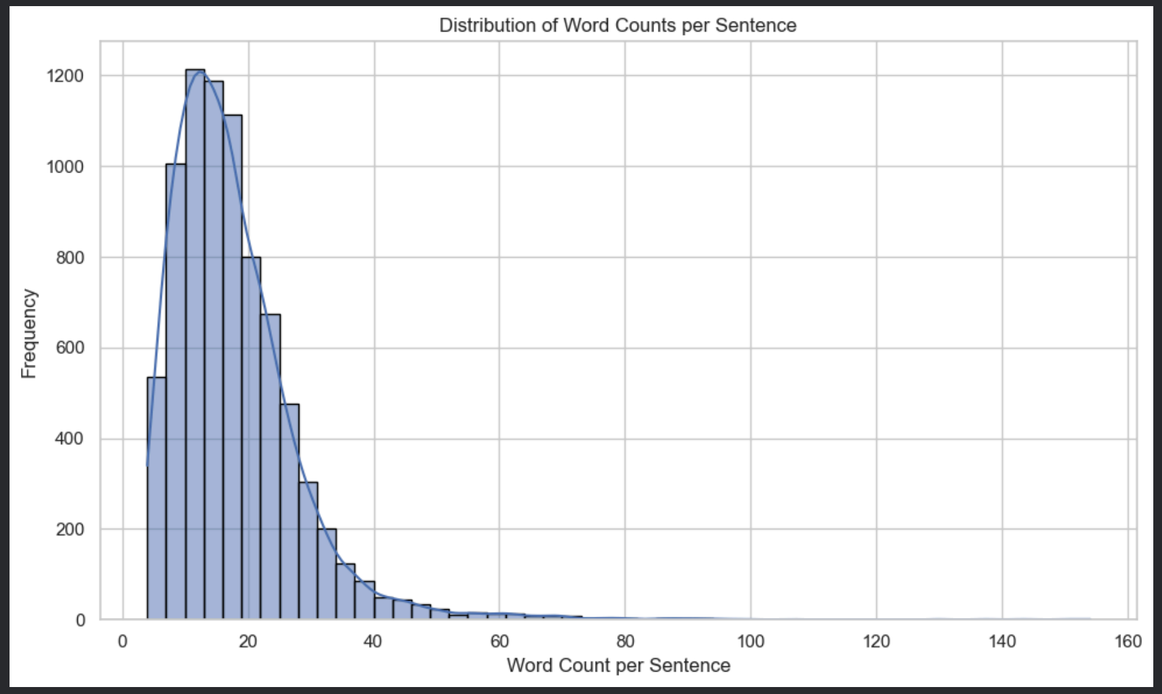
Statistic	Value
Count	667
Mean	206.70
Std	144.01
Min	4.00
25%	111.00
50% (Median)	172.00
75%	262.50
Max	1375.00

Sentence Count Statistics



Statistic	Value
Count	667.00
Mean	11.96
Std	8.90
Min	1.00
25%	6.00
50% (Median)	10.00
75%	15.00
Max	84.00

Words in sentence Statistics



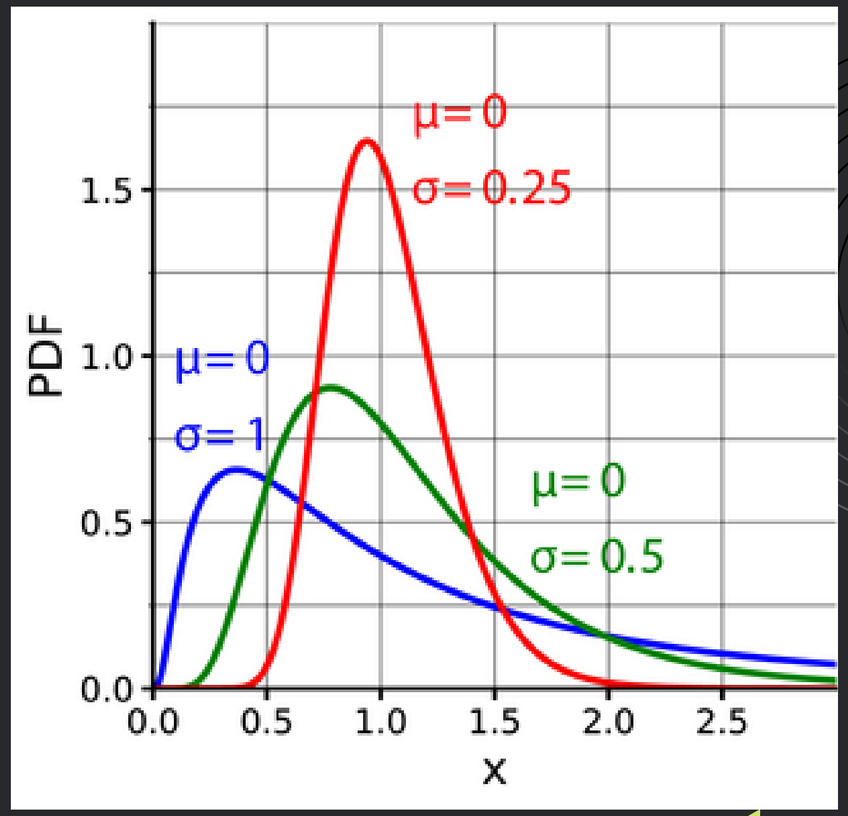
Statistic	Value
Count	7975.00
Mean	17.58
Std	10.52
Min	4.00
25%	11.00
50% (Median)	16.00
75%	22.00
Max	154.00

Key Observations from Data Analysis

1. A total of 668 student summaries are present in the given data.
2. The average word count per summary is 207, with extremes ranging from 4 to 1375 words.
3. The average number of sentences per summary is 12. Some summaries had only one sentence, indicating minimal effort or submissions made purely for compliance.
4. The average sentence length is 18 words, with some reaching up to 154 words in a single sentence.

Key Observations from Data Analysis

5. All three distributions—word count, sentence count, and sentence length—seem to follow a log-normal trend, as shown in the previous plots.



Log Normal Distribution

Key Observations from Data Analysis

6. Academic dishonesty cases identified:
   - Submissions containing only irrelevant links.
   - One case of exact duplication.
   - Several instances of lightly reworded copying, detected using vector similarity scores (rest of them included in the submission).

👉Pair 4: (Index 432, Index 622) — Similarity: 0.9925

📌Summary 432: we learned about crisp-dm (cross industry standard process for data mining), a six-step, cyclical process. it starts with business understanding, where we define the problem and assess relevant statistics. then comes data understanding, where we collect and explore the dataset. the modeling phase involves building and evaluating different models, followed by evaluation, where we assess the results to ensure they align with business needs. finally, in the deployment stage, the model is finalized, and reports are generated. after that, we explored exploratory data analysis (eda), a crucial approach in statistics and data science for investigating datasets. we also looked at outliers and quartiles, understanding how boxplots help visualize variability and detect outliers. additionally, we studied inter-feature relationships using matrix plots to identify correlations between different features. we then learned about three types of missing data: missing completely at random (mcar), where the missing values have no pattern; missing at random (mar), where missing data is related to some observed variables; and missing not at random (mnar), where the missing values are dependent on unobserved factors. we discussed true outliers, which are extreme values in a dataset that are not errors but actual observations.

📌Summary 622: we were taught about crisp-dm (cross industry standard process for data mining), a cyclical six-step process. we begin with business understanding, defining the problem and examining pertinent statistics. next is data understanding, gathering and understanding the dataset. the modeling stage entails constructing and testing various models, followed by evaluation, where we test the outcomes to make sure they meet business requirements. last but not least, in the deployment phase, the model is completed, and reports are produced.then, we learned about exploratory data analysis (eda), an important statistic and data science method for examining datasets. we also considered outliers and quartiles and recognized how boxplots can display variability and identify outliers. we also learned about inter-feature relationships with matrix plots to look for correlations among various features. we then discovered three categories of missing data: missing completely at random (mcar), in which the missing values are not patterned; missing at random (mar), in which missing data is a function of some observed variables; and missing not at random (mnar), in which the missing values are a function of unobserved variables. finally, we talked about true outliers, which are outliers in a dataset that are not errors but real observations.

Data Cleaning and Feature Engineering

1. **Character encoding issues** were identified and corrected (e.g., " replaced with â€™ in some submissions).
2. One submission containing only a **link** was **removed** from the dataset.
3. Sentence tokenization logic was improved to handle special cases (e.g., avoiding splitting at "i.e."). Sentences with fewer than 3 words were excluded to reduce noise.
4. A CSV file was generated with the following features for further processing: **'Session_Summary', 'tokens', 'word_count', 'sentences', 'sentence_count'**.

Session_Summary	tokens	words_count	sentence_count
we started our lecture with a recap of previous le	['we', 'started', 'our', 'lecture', 'with', 'a', 'rec	224	16
in this session, we explored various feature	['in', 'this', 'session', 'we', 'explored', 'various	271	16
population and sample were further discussed	['population', 'and', 'sample', 'were', 'further	336	22
we first looked at all the summaries and	['we', 'first', 'looked', 'at', 'all', 'the', 'summar	219	13
midsem metrics for evaluation and also discussio	['midsem', 'metrics', 'for', 'evaluation', 'and',	13	1

Exported csv file

02. Vectorizing Summaries

The following Vectorization techniques were experimented

A) TF-IDF: Converts text into weighted term-frequency vectors based on word importance across documents.

B) Doc2Vec: Generates fixed-length vector representations that capture the semantic meaning of entire documents.

C) GINA (Transformer-based): Uses a pre-trained transformer model to produce context-aware embeddings for summaries.

A) TF-IDF: Term Frequency - Inverse Document Frequency

The first method that we tried for clustering the summaries was the TF-IDF method:

1. **TF - Term Frequency**: It is defined as the number of times a particular term (in our case, any word) appears in a document compared to the total number of terms.
2. **IDF - Inverse Document Frequency:** It is defined as the log of the fraction of the total number of documents in the given dataset to the number of records that contain a particular term.

Using the product of these two parameters, the combined parameter TF - IDF is calculated, which can then be used to cluster the summaries based on the values for each word.

Preprocessing the data

- The dataset achieved after performing EDA needs to be further modified and processed using the TF- IDF method. The stop words need to be removed so that they do not hinder the clustering as they will appear in almost every document, thus misleading the clustering algorithm. To do so, we used the built-in stop word set *'english'* for the TF-IDF vectorizer function of the scikit-learn library.

- We also set the maximum number of features for the vectorizer to 100, as too many features would not yield significant results. The main words were bound to have substantial values and would hence be captured in the top 100.

Preprocessing the data

- We set the vectorizer to ignore any words appearing in fewer than two documents, as there is a very low chance that only one person in the class reported something. In contrast, none of the other summaries from the same session reported that term. Including such terms could mislead our classifier to classify such summaries as completely different summaries despite other similarities.

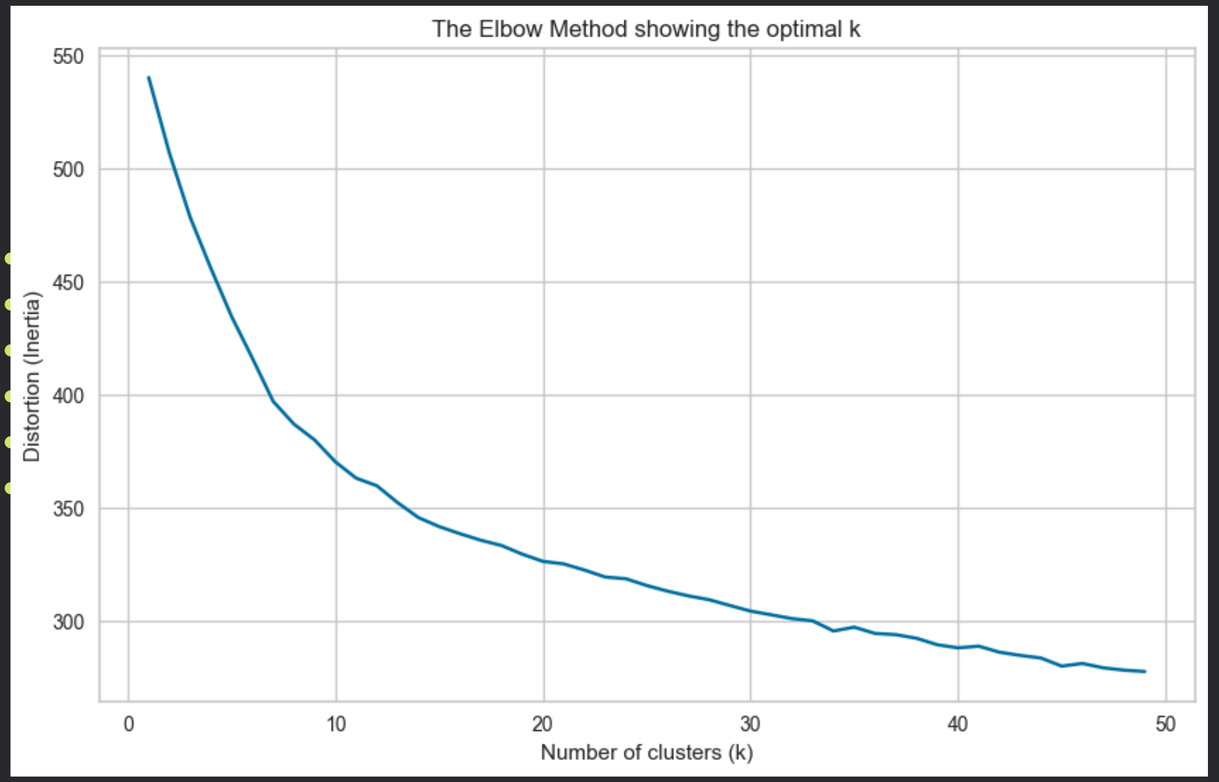
- We set the vectorizer to ignore terms which came in more than 90% of the documents, as they were bound to be terms like ‘data’, ‘science’, ‘machine’, ‘learning’ which would again not yield any beneficial vectorization results as they would be everywhere, thus misleading the classifier to classifyall the summaries into one group.

Results

- The TfidfVectorizer() function from the scikit-learn library was configured with the appropriate parameters, and then the dataset was fit on the created vectoriser. This created a TF-IDF matrix containing 14916 storage elements, where the words from each summary, which appeared in the top 100 words, were assigned TF-IDF values.
- Using this matrix, we used the K-means clustering algorithm and the elbow method to find the appropriate number of clusters that could be used. The range for plotting the elbow curve was from 1 to 50, and a good elbow was seen between 5 and 10.
- However, we thought the summaries would have encompassed at least 20 sessions, so we decided to choose 20 clusters as the optimal number and see the results.

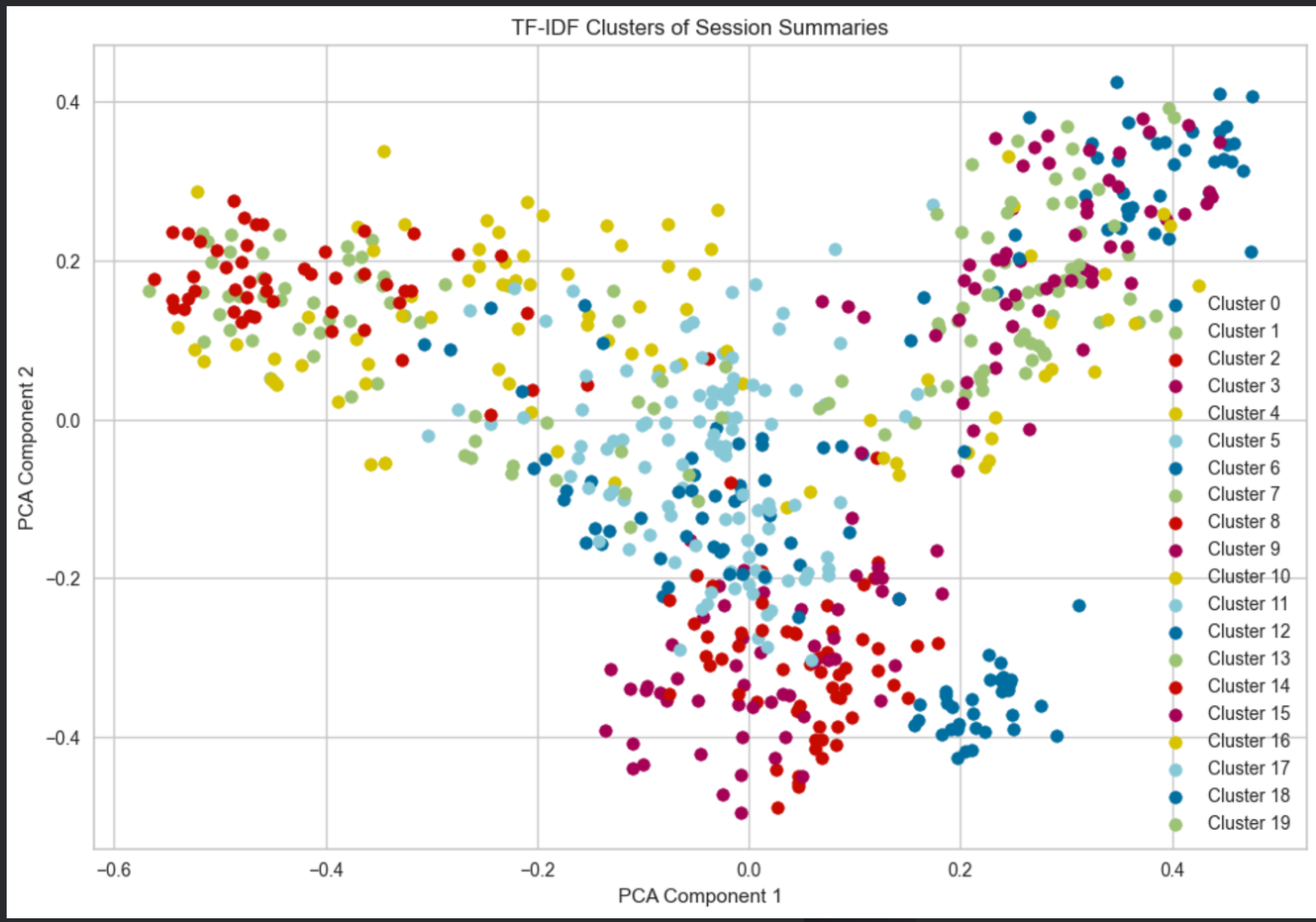
Results

- From the PCA plot, we can see that in 2D, we do not see very well-defined clusters, which is understandable, considering that there are 100 total features. This was our first indication that this method might not work well for our purpose.
- We also obtained the top terms for each cluster to check whether the clustering was working and whether it used appropriate words for clustering.



The Elbow method curve for K-means clustering

2D PCA plot for 20 clusters using the TF-IDF



Following were the top words for each cluster:

Cluster 0:
data, missing, outliers, understanding, eda, values, analysis, mean, plots, random

Cluster 1:
population, sample, parameters, line, sum, mean, simple, confidence, fit, statistics

Cluster 2:
clustering, curve, positive, means, classification, true, points, logistic, model, data

Cluster 3:
learning, nominal, ordinal, ratio, interval, data, regression, clustering, classification, example

Cluster 4:
regression, error, excel, distribution, data, model, errors, linear, plot, line

Cluster 5:
function, probability, logistic, class, matrix, metrics, regression, discussed, using, model

Cluster 6:
encoding, feature, data, values, dimensionality, variable, discussed, variables, problem, class

Cluster 7:
pca, vif, data, variance, analysis, features, dimensionality, distribution, number, used

Cluster 8:
distribution, sample, mean, population, standard, confidence, normal, samples, regression, means

Cluster 9:
data, analysis, eda, session, excel, using, dataset, learnt, different, outliers

Cluster 10:
vif, data, dimensionality, features, discussion, feature, variance, model, missing, discussed

Cluster 11:
training, test, data, model, multiple, value, linear, matrix, use, sample

Cluster 12:
regression, models, model, features, linear, feature, method, best, methods, data

Cluster 13:
plots, data, outliers, analysis, like, eda, discussed, saw, dataset, excel

Cluster 14:
positive, true, curve, class, model, logistic, matrix, clustering, different, line

Cluster 15:
data, class, outliers, points, problems, discussed, methods, missing, analysis, values

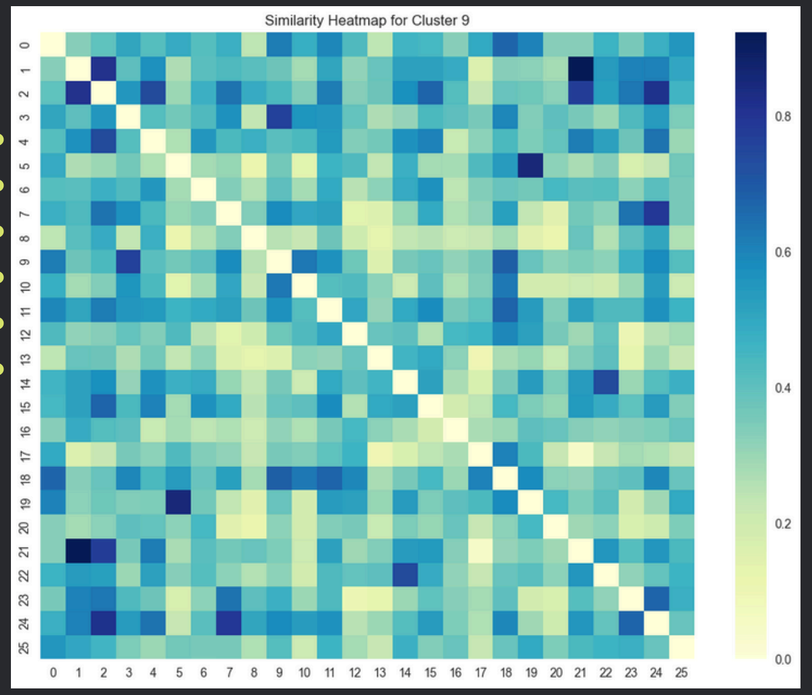
Cluster 16:
interval, confidence, value, population, sample, zero, regression, error, variable, linear

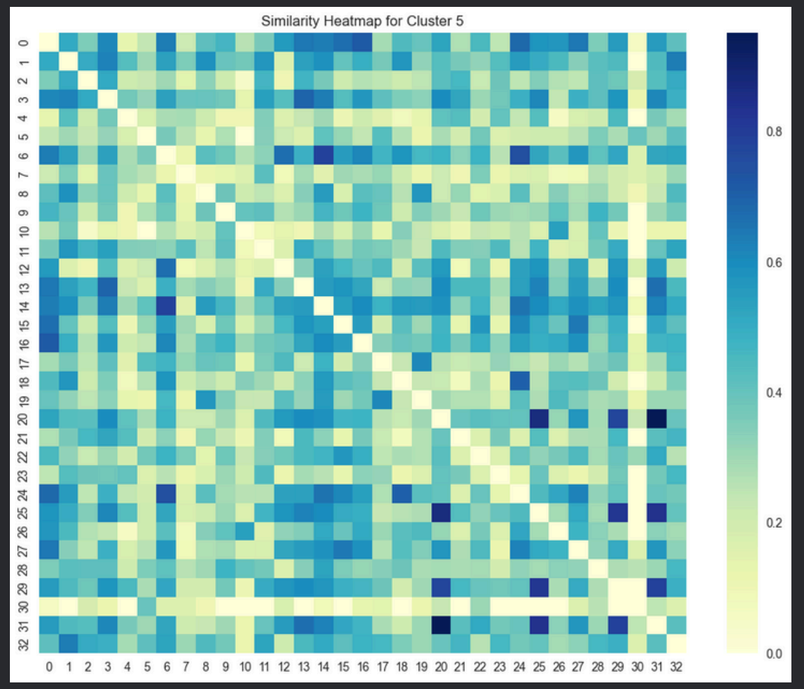
Cluster 17:
value, feature, features, model, multiple, mlr, values, better, regression, linear

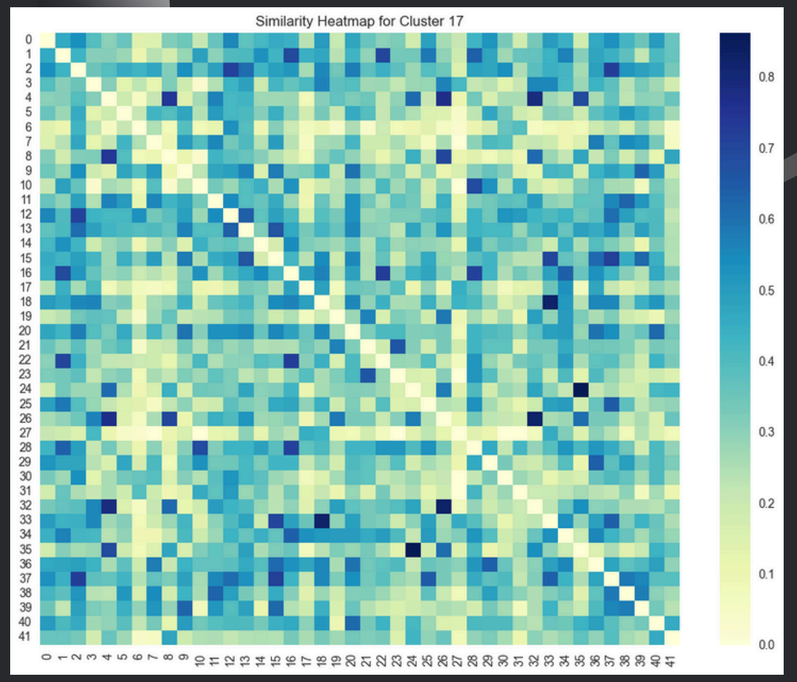
Cluster 18:
sir, learned, data, class, like, values, started, explained, model, regression

Cluster 19:
multiple, independent, variable, variables, regression, linear, data, mlr, model, discussion

Following are some correlation matrices for summaries within a cluster:







From these matrices, we can observe that for the given clusters, the similarity between the summaries is too low (less than 0.5), which means that summaries within the same cluster are also very different. Such low correlations were observed for 12 out of the 19 clusters created. This indicates a failure of this vectorisation method, so we chose not to use this method.

B) Doc2Vec

1. **Captures document-level meaning:** Learns embeddings that represent the entire summary, not just individual words.

2. **Two model variants:** Uses Distributed Memory (DM) and Distributed Bag of Words (DBOW) architectures to capture context.

3. **Unsupervised approach:** No labels needed — learns patterns directly from the text data.

4. **Enables semantic comparison:** Similar summaries have similar vectors, useful for clustering and ranking.

Doc2Vec – Key Parameters

- **vector_size:** Dimensionality of the feature vectors (embedding size).
- **window**: Maximum distance between the current and predicted word within a sentence.
- **min_count**: Ignores words with total frequency lower than this.
- **epochs**: Number of training iterations over the corpus.
- **dm**: Defines the training algorithm — 1 for Distributed Memory (DM), 0 for DBOW.
- **negative**: Number of negative samples used (if hs=0).

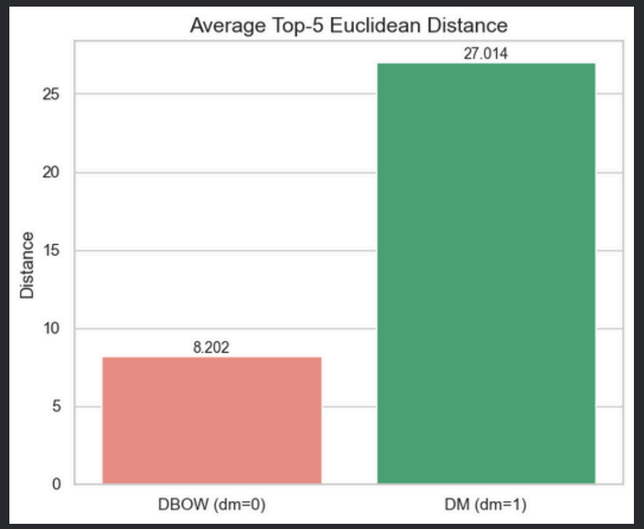
Embedding Quality Evaluation – Loss Function Experiments:

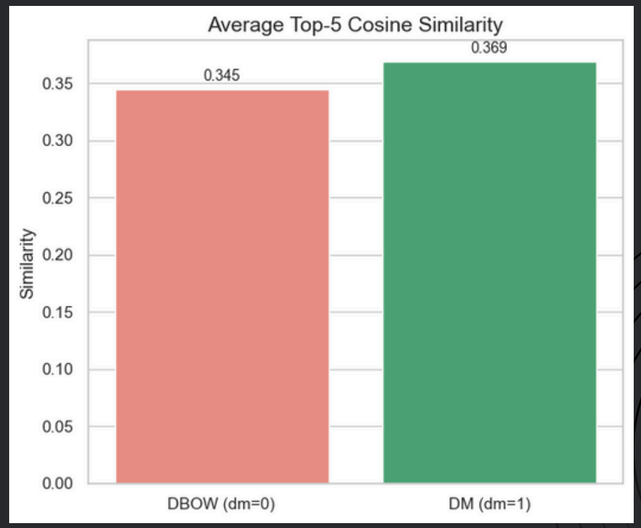
We analyzed embedding behavior by plotting the following metrics for 40 random vectors, while varying one and keeping the other parameters (mentioned in prev. slide) constant:

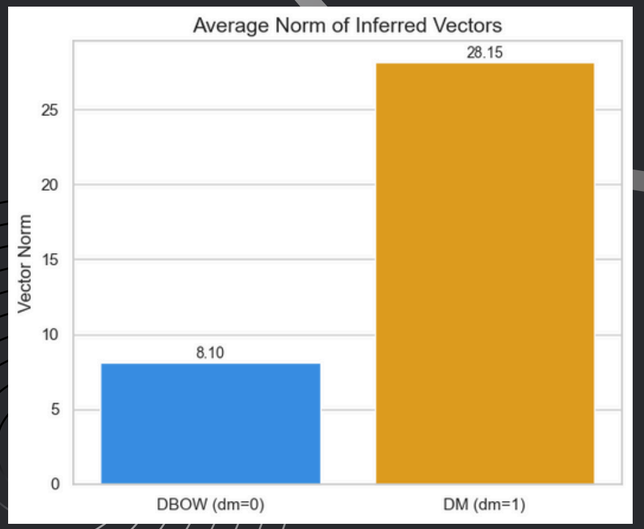
- Norm of each vector
- Average cosine similarity with its top 5 nearest neighbors
- Average norm of its top 5 nearest neighbors

These plots helped evaluate how embedding parameters impact vector quality, clustering behavior, and semantic coherence.

Finding Optimal dm value (1 or 0)

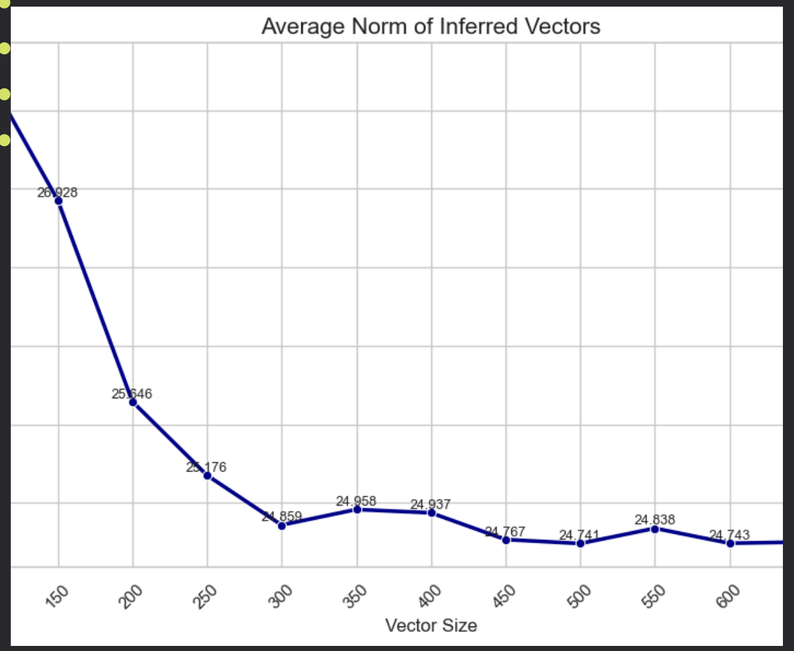


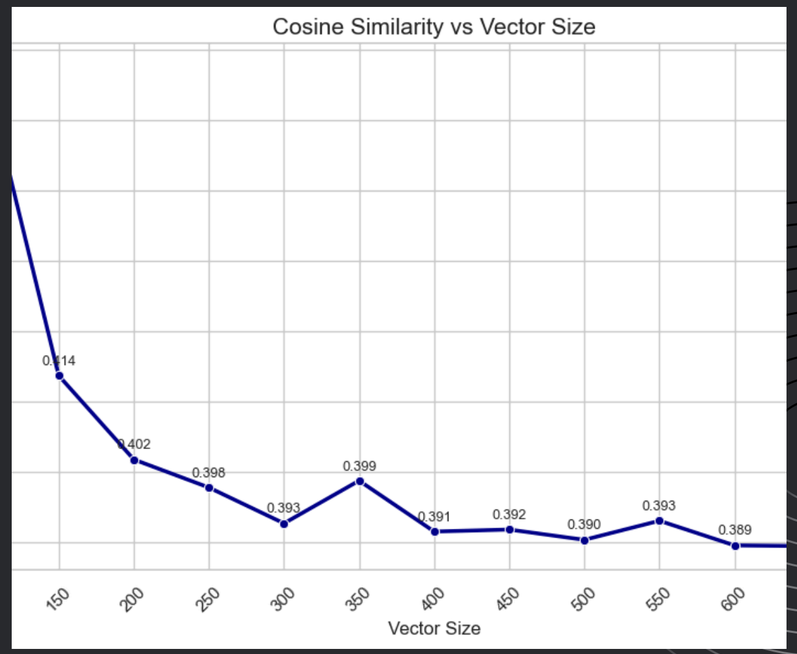


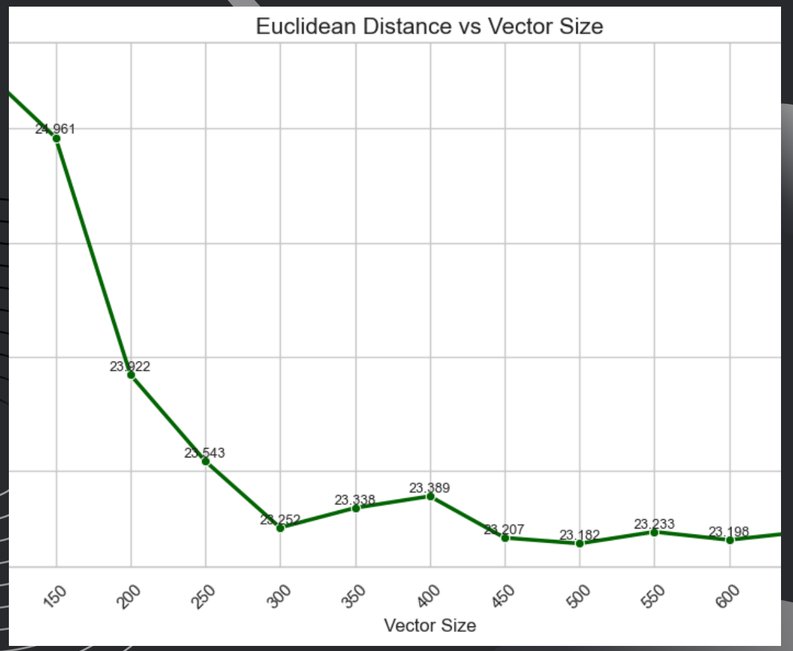


Based on the above analysis, we concluded that dm=1 (Distributed Memory) performs better for our use case

Finding Optimal vector size

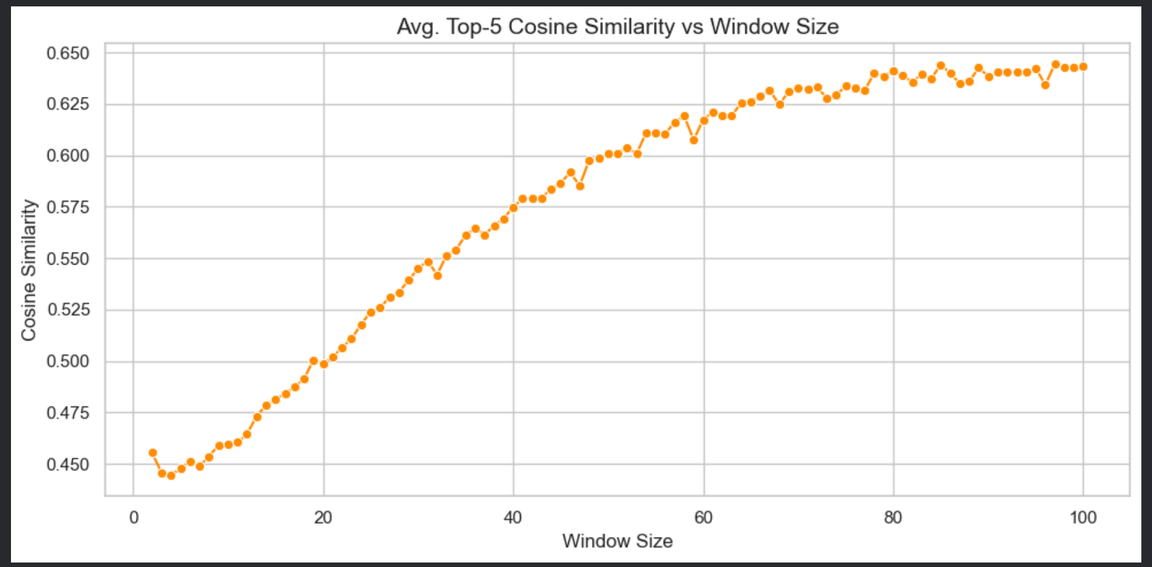


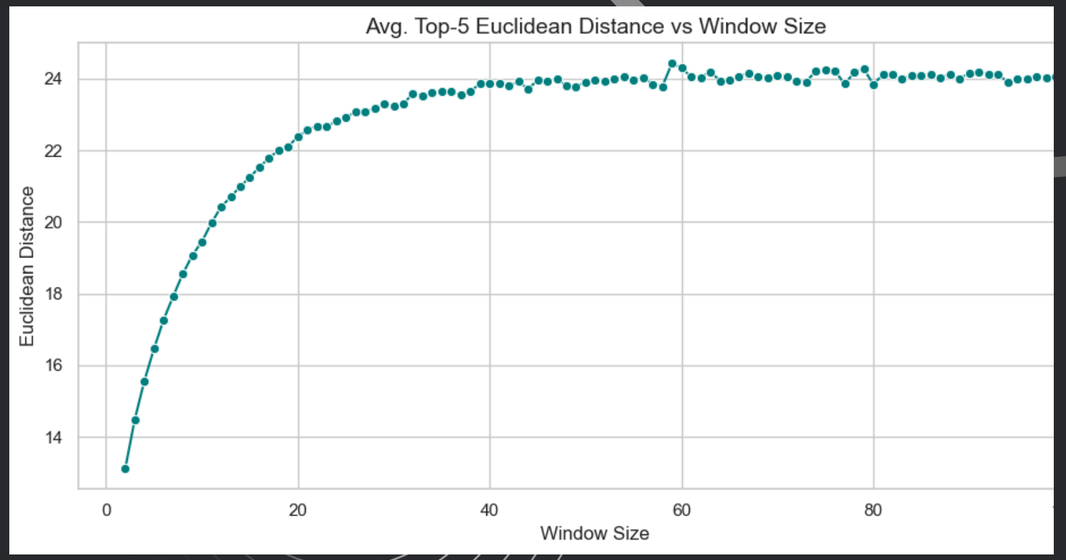




Based on the above analysis, we concluded that a vector size of approximately 300 yields the best performance for our dataset, as the metrics saturate after that.

Finding Optimal Window size





Based on the above analysis, we concluded that a window size of around 40 would be optimal.

Realizations from Initial Clustering Attempts

After estimating optimal parameters and performing clustering using Doc2Vec, manual analysis revealed poor clustering quality, with unrelated sessions grouped together.

Key Learnings:

1. Training models to capture contextual meaning from scratch is challenging.
2. Limited training data leads to poor-quality embeddings.
3. For tasks like ours, pretrained models offer significantly better performance, especially when fine-tuned for specific needs.

Conclusion:

We shifted to using Jina, a transformer-based pretrained embedding model, for improved semantic understanding and clustering accuracy.

C) JINA (Transformer-based)

1 .**Transformer-based architecture:** Built on state-of-the-art transformer models to capture deep contextual meaning in text.

2. **Pretrained on diverse English data:** Offers robust generalization even on unseen or domain-specific text.

3. **Generates dense, semantic embeddings:** Encodes entire documents or summaries into high-quality vector representations.

4. **Ready for downstream tasks:** Realized it would be ideal for clustering, ranking, semantic search, and similarity comparisons tasks.

Analysing the embedding of Jina

Random summaries and text were embedded using this technique and anlayzed:

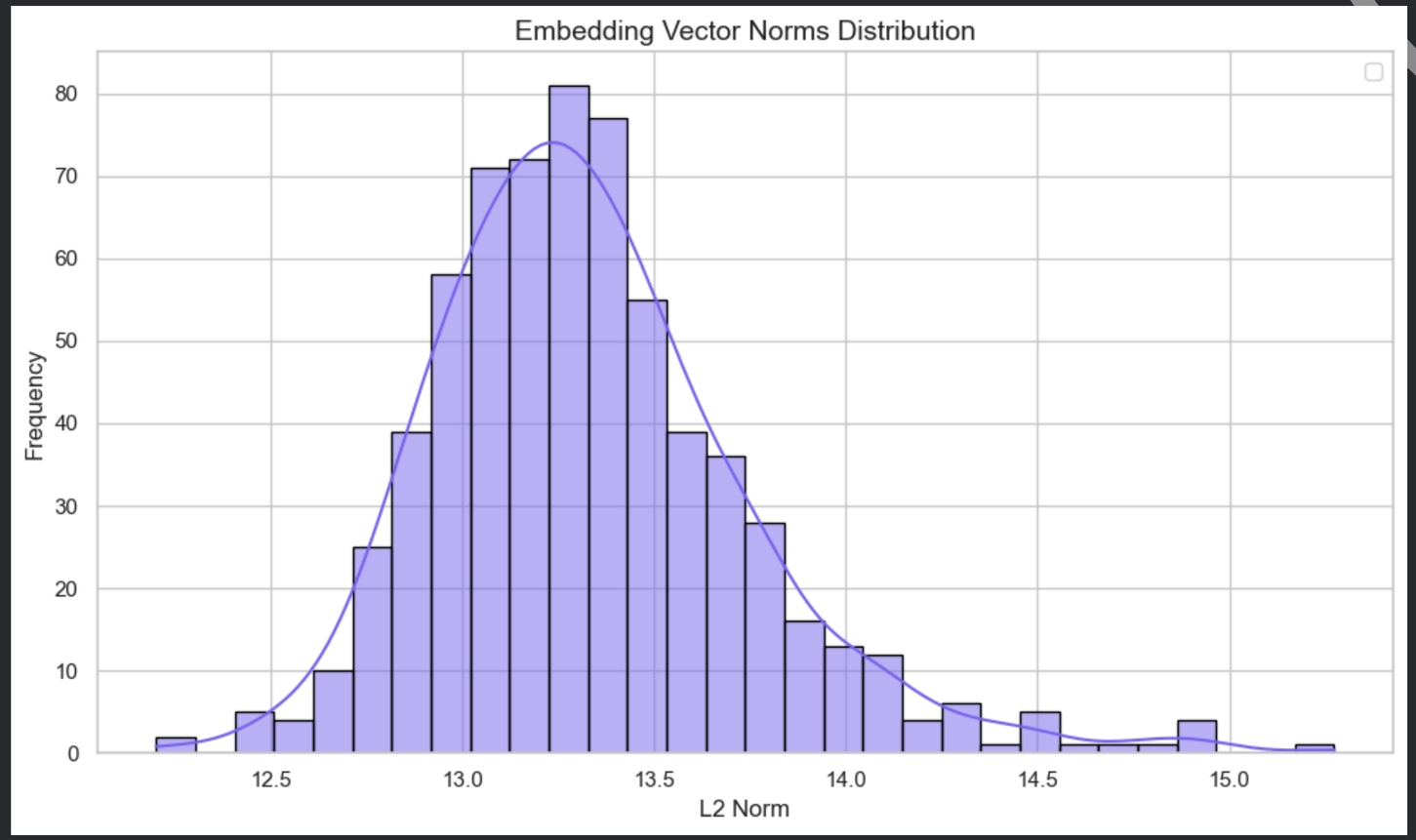
Text-1	Text-2	Cosine Similarity
Summary-1	Summary-2 belonging to same session as Summary-1 (picked through manual analysis)	0.9325
Summary-1	Summary-2 belonging to a different session as Summary-1	0.7700
Summary-1	Some random text about sports from the internet	0.6796
Summary-1	Some random text about Economics from the internet	0.6871

Embedding Quality – Key Observations:

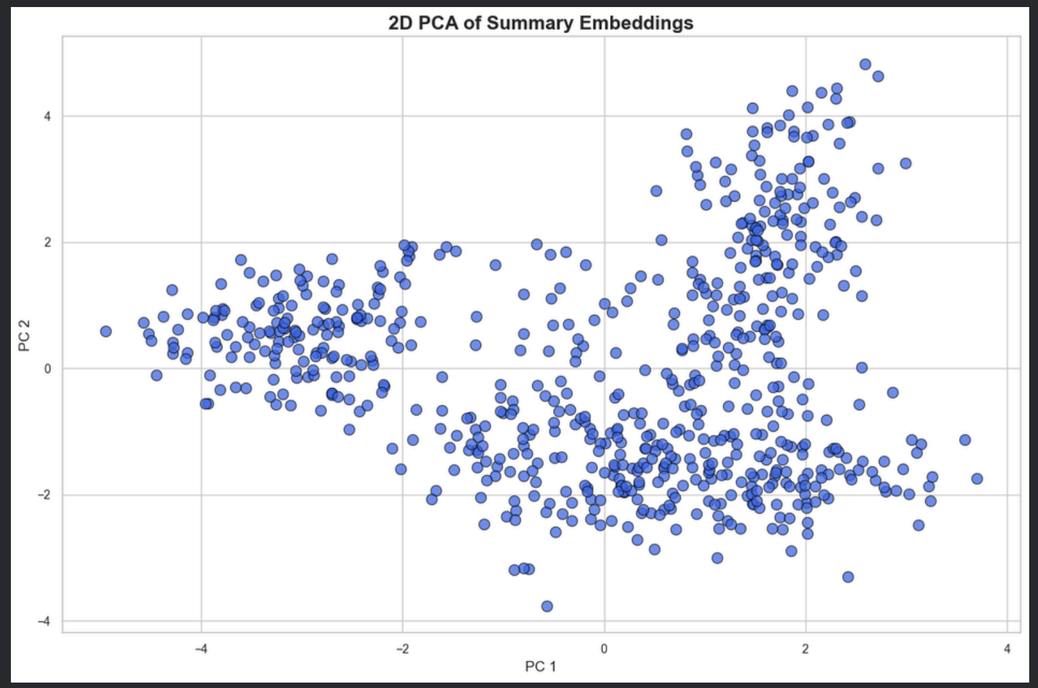
1. **Same session texts show very high cosine similarity (typically > 0.9).**
2. **Different sessions within the same domain (e.g., ML/DS topics) show moderately high similarity (~0.8).**
3. **Texts from different domains exhibit lower similarity (~0.65).**

These patterns indicate that the Jina embedding model captures semantic relationships effectively, making it highly suitable for context-aware clustering.

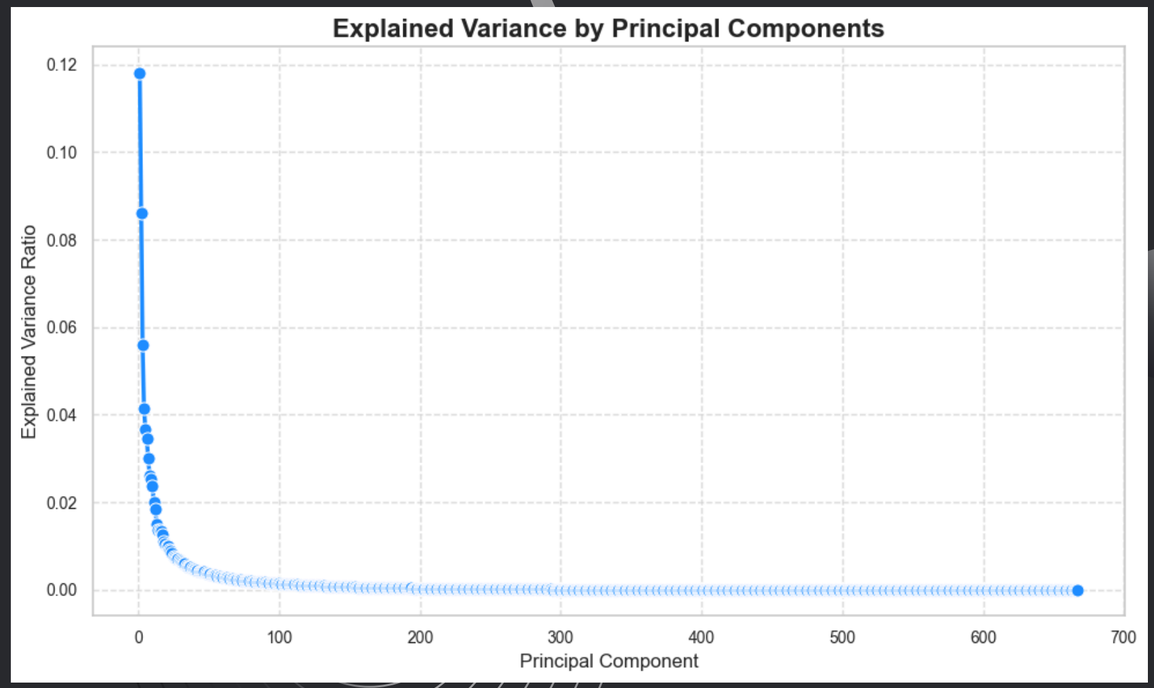
Analysing the embeddings of Jina



PCA Analysis of embeddings of Jina



The figure shows well-separated clusters using only the first two principal components (PCs), indicating strong semantic separation in the embedding space.



Decay of Variance explained by Principal Componensts

03. Clustering Summaries

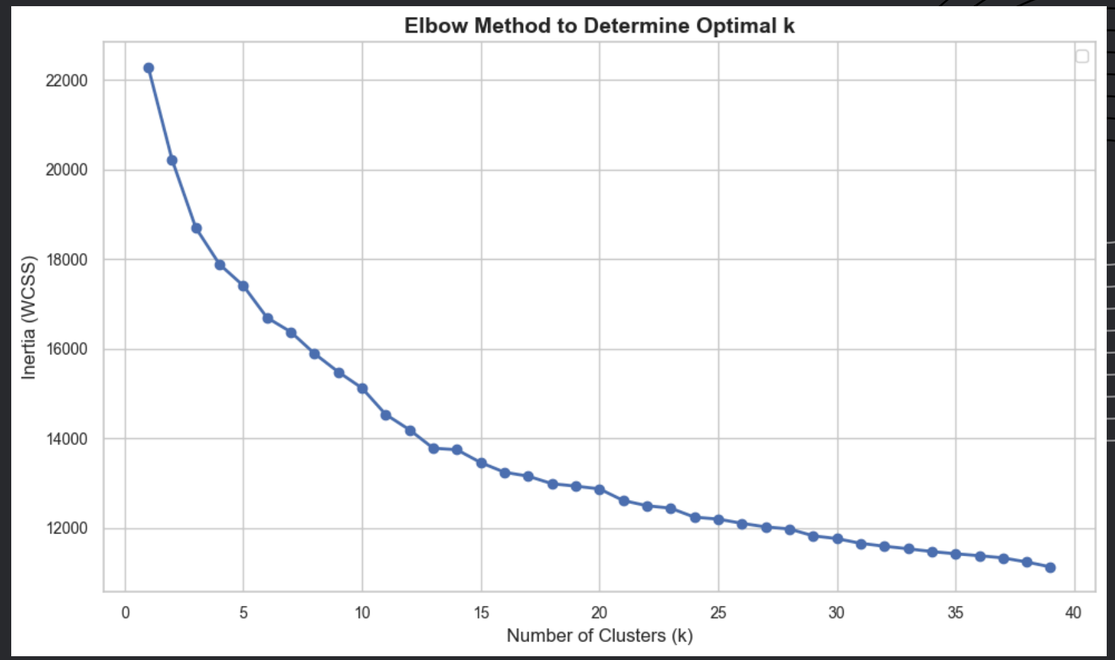
The embeddings generated by the Jina Transformer model were selected for downstream tasks such as clustering, ranking, and visualization.

Initial Thought Process Before Clustering

1. **DBSCAN** was initially tried but resulted in **very few clusters.** Switched to K-Means for **control over the number of clusters.**

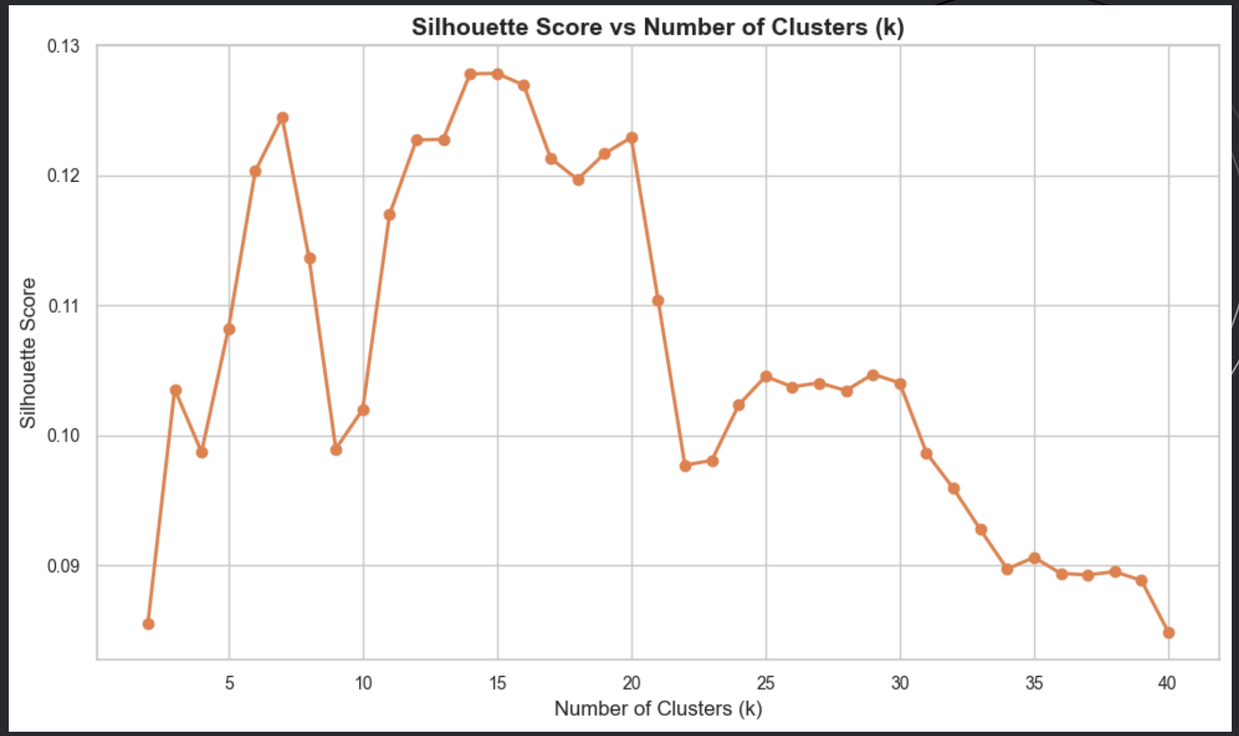
Initial Thought Process Before Clustering

2. **Elbow** plot was generated, but **no clear elbow point** was observed (see plot below), making it unsuitable for choosing optimal K.



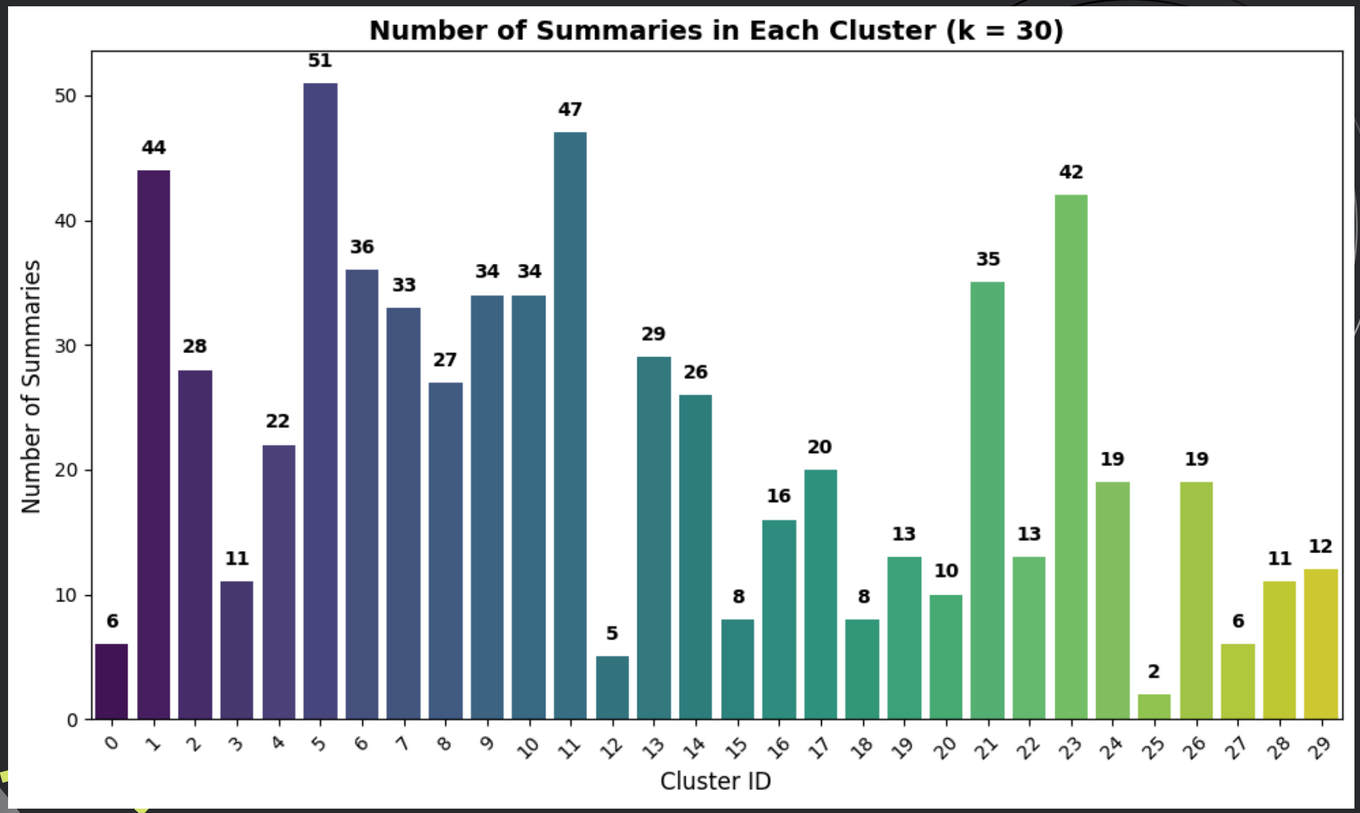
Initial Thought Process Before Clustering

3. Silhouette scores were plotted for various K values, but no significantly high score emerged (see plot below). So decided to manually select the value of K.



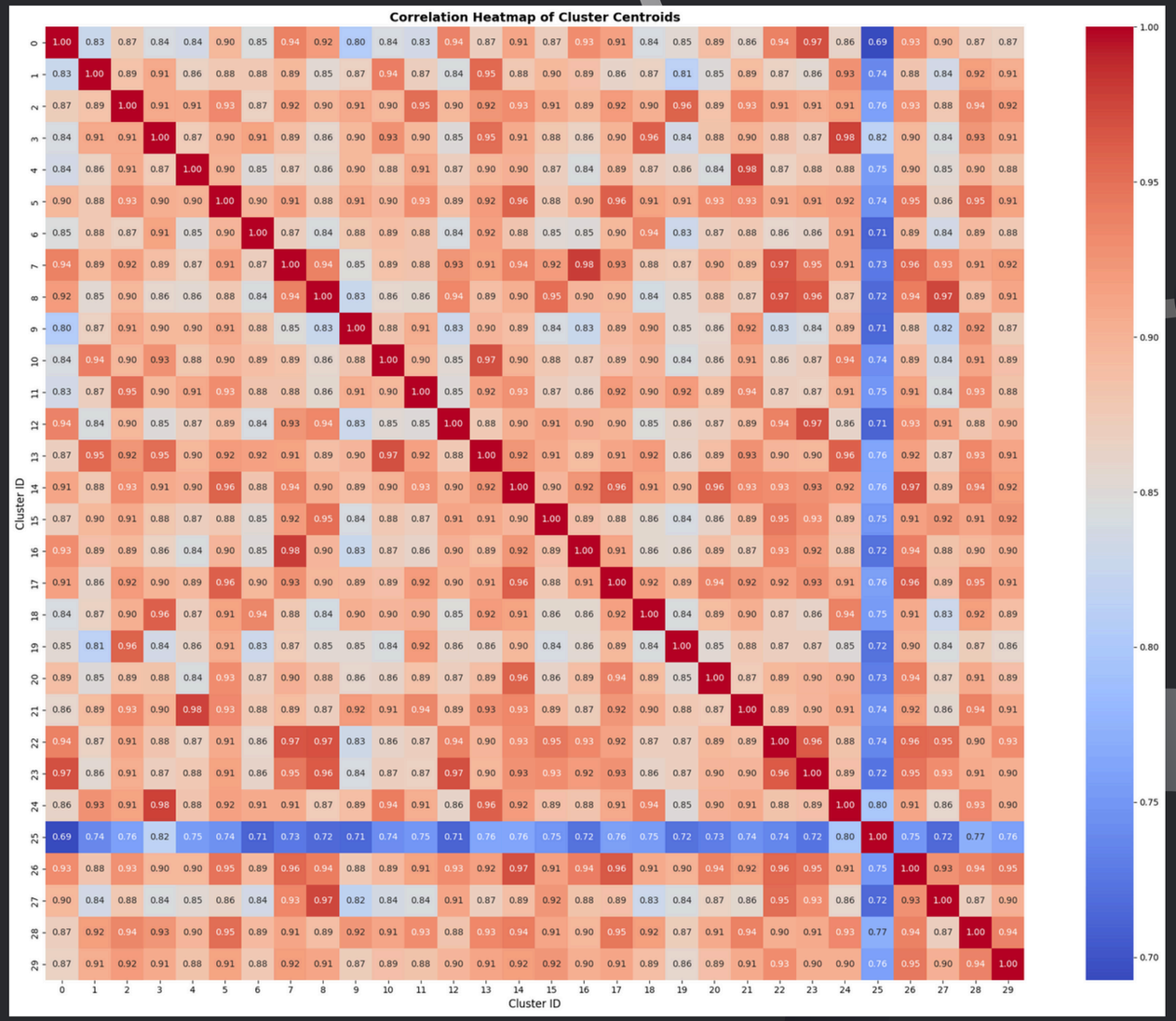
Initial Thought Process Before Clustering

4. Based on domain knowledge (of the classes), the number of sessions was estimated to be between 10 and 30. Choosing a higher K is safer, as similar clusters can be merged later, while a low K risks merging distinct sessions. Final decision: Chose K = 30 for clustering. Obtained clusters as below.



Refining and Updating Clusters

1. **Keyword Analysis**: Each cluster was manually reviewed by examining key terms to understand its core theme and compare across clusters.
2. **Inter-Cluster Similarity**: Cosine similarity between cluster representatives was computed. Pairs with high similarity were manually reviewed, and if their content matched, they were merged.



Refining and Updating Clusters

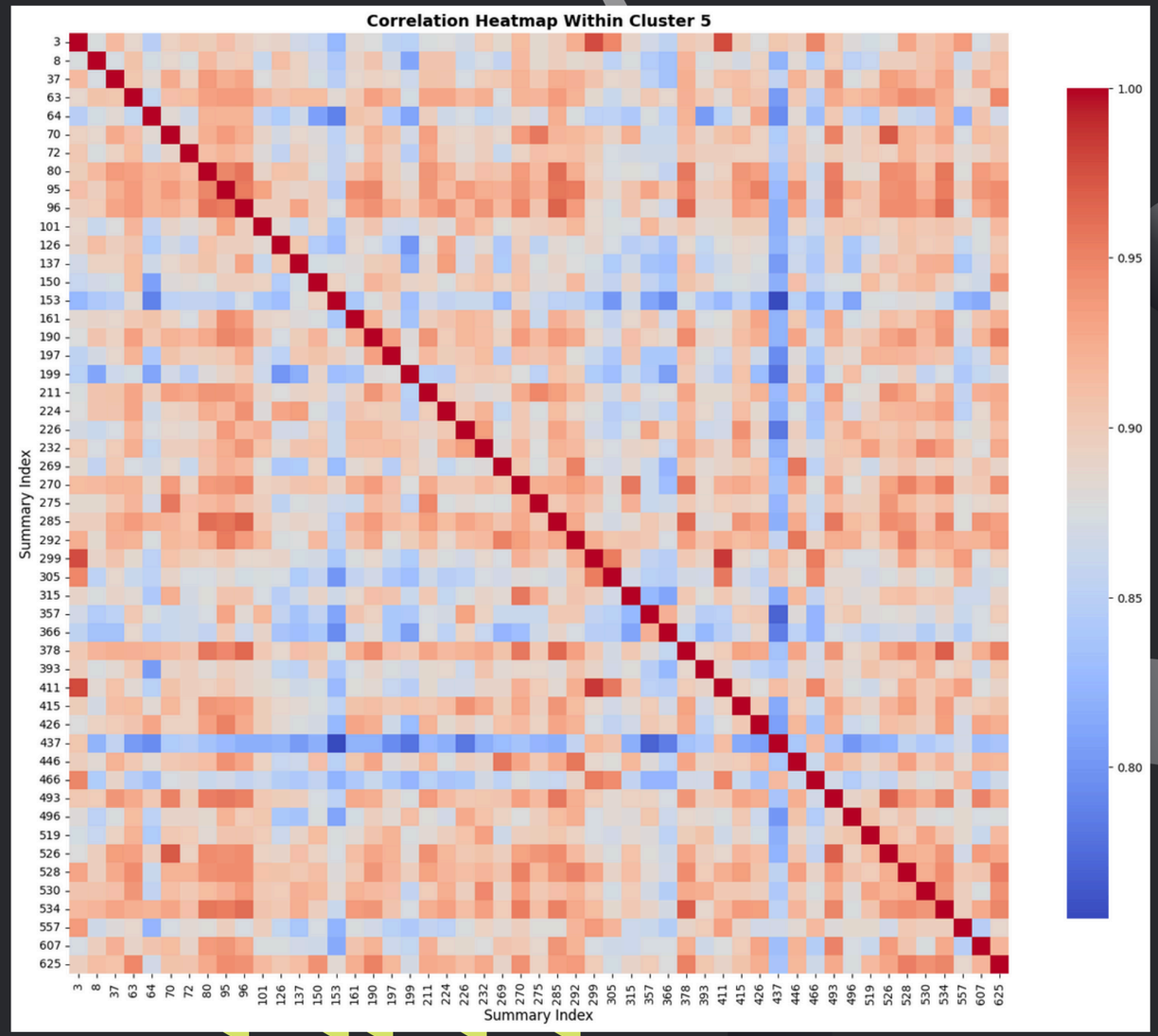
3. **Small Cluster Evaluation**: Clusters with very few summaries were analyzed. If they showed high similarity to a larger cluster and shared content themes, they were merged accordingly.
4. **Reducing Misclustering**: To identify misclustered summaries:
   - Intra-cluster similarity heatmaps were plotted.
   - Summaries with consistently low similarity to others (visibly distinct on the heatmap) were flagged.
   - These were manually reassigned to more appropriate clusters.

Refining and Updating Clusters

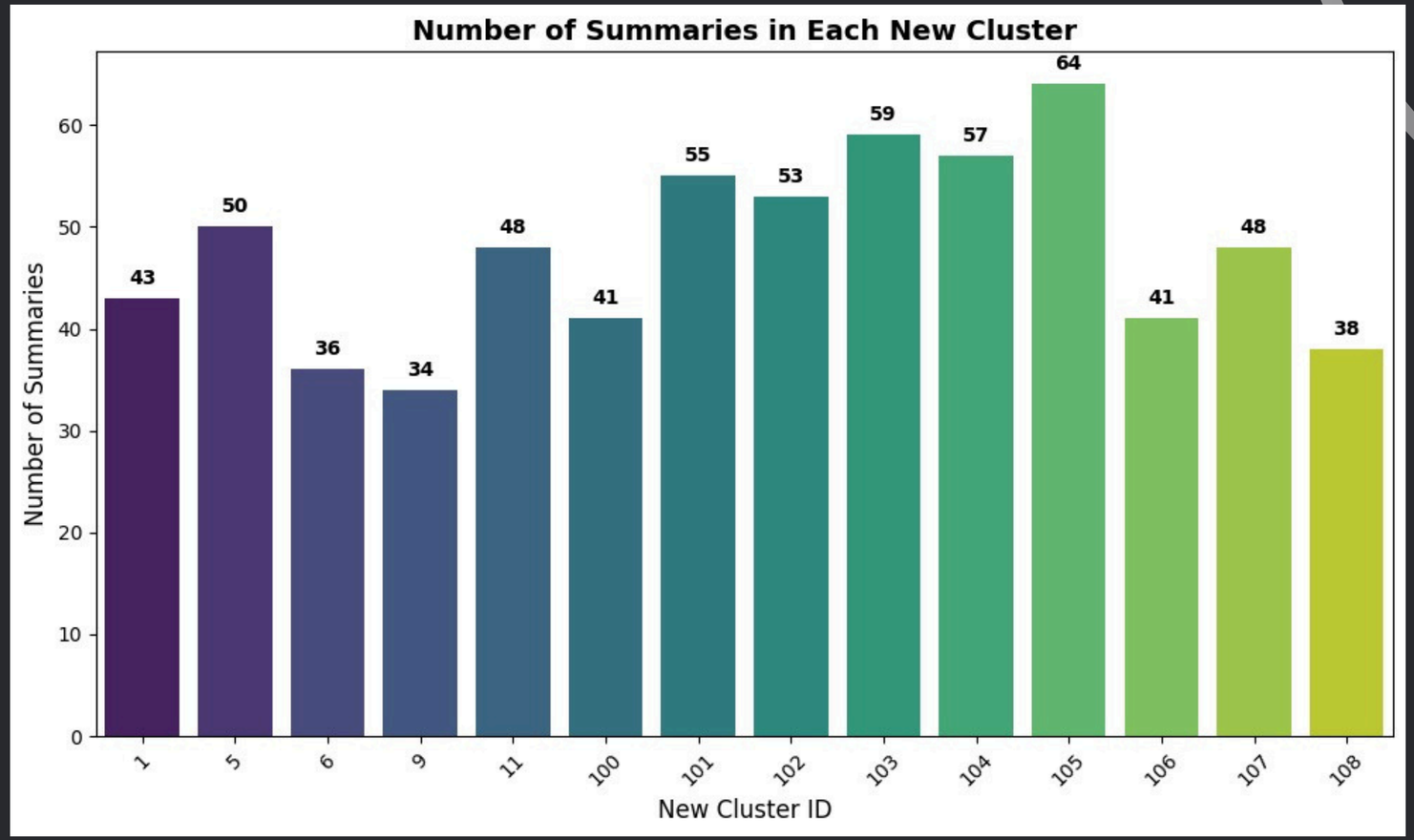
The heatmap shows the cosine similarity between summaries within a cluster(clusterID 5).

Submissions that appear distinctly in blue(eg: id437 here) indicate low similarity with the rest of the summaries in the cluster.

These outliers were flagged for review and manually reassigned wherever necessary to reduce misclustering and improve overall cluster coherence.



After performing the above refinement steps, we obtained the final distribution of summaries across clusters.



These cluster IDs were retained and used consistently in all downstream processes such as ranking, visualization, and search

04. Cluster Representative Summaries

For each session, a single, representative summary with the following characteristics was created :

1. **Complete and comprehensive**: Included all key concepts, explanations, and in-class discussions.
2. **Consistent terminology**: Used the same format throughout (e.g., r_square used everywhere instead of mixing r^2, r^2, etc.).
3. **Ordered structure**: Followed the sequence in which topics were introduced during the lecture, avoiding random or jumbled presentation.
4. **Semantic grouping**: Combined related terms to preserve meaning (e.g., linear-regression instead of linear regression, which could separate related ideas when tokenized; pivot-table instead of pivot table).

Why Representative Summaries?

Creating a single representative summary per session helped solve key issues:

1. **Standardized input for visualization**: Avoided mismatches like r^2 vs r_square and kept related ideas together (eg :Linear-regression) ensuring consistent keyword-based visualizations .
2. **Reliable basis for ranking**: Served as a complete reference to compare and rank student-submitted summaries.
3. **Cross-session analysis**: Enabled meaningful visualization of topic overlaps between sessions due to comprehensive content.
4. **Improved keyword search**: Provided a clean, unified summary for accurate keyword matching in the search application.

Generating Representative Summaries

The following steps were followed to create session representaive summaries

- **Step 1**: Concatenated all summaries within a cluster to form a single block of text.
- **Step 2**: Appended a detailed prompt describing the expected structure, consistency, and completeness.
- **Step 3**: Sent the combined input to the Gemini Flash 1.5 API for summary generation.
- **Step 4**: Collected and stored the generated representative summaries along with their corresponding session IDs.

Prompt used

summarize the following passage in a clear, structured, and comprehensive manner. the passage consists of concatenated summaries written by multiple students, so analyze the text to identify the logical flow of the class content and organize the summary accordingly. retain all key ideas and important keywords. do not omit any crucial information. ensure that each point is mentioned only once, without any repetition, while fully preserving its meaning. the summary must follow these rules:

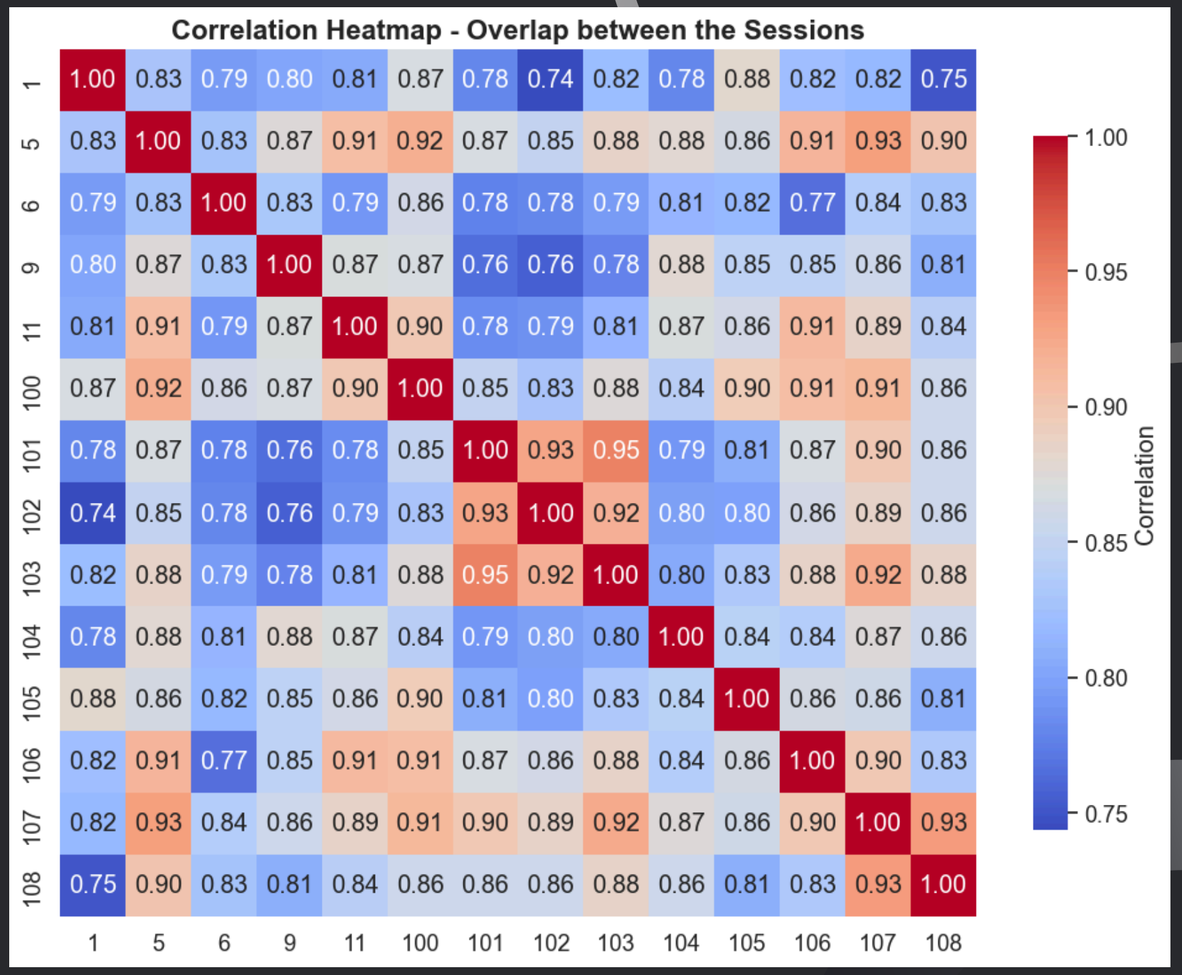
1. write in plain text with correct punctuation
2. no formatting of any kind (no bold, italics, or capital letters)
3. no unnecessary spacing between sentences
4. expand all contractions (for example, write 'it is' instead of 'it's', 'have not' instead of 'haven't', and 'example fully' instead of 'e.g.')
5. do not use subscripts or superscripts; for example, write r_square instead of r^2
6. important: for any group of words that together refer to a single concept, write them as a single hyphenated phrase. for example: linear-regression, multiple-linear-regression, pivot-table, f-statistic, confusion-matrix. this applies not just to the examples above, but to any phrase in the passage where the words together refer to one concept. always think: do these words combine to represent one idea or entity? if yes, then write them with hyphens.

output only the final summarized text, without any introductory or closing statements.

Representative Summaries in csv file

Cluster_ID	Representative_Summary
1	the class focused on exploratory-data-analysis (eda) using excel pivot-tables. pivot-tables were used to
5	the class began by analyzing summary-submission trends, revealing decreasing submissions but increasing
6	heatmaps provide pairwise-analysis of multiple-parameters but lack detailed information. variance-
9	the class covered feature-encoding techniques for preparing categorical and textual data for machine-
11	the class began with a review of gradient descent and an introduction to logistic-regression. a
100	the class began with a review of the mid-semester exam, including a detailed walkthrough of the solution
101	the class covered population-parameter estimation using sample-means, focusing on mean and variance
102	the lecture began by differentiating population-parameters and sample-statistics, emphasizing the
103	the class began with an excel-based session on simple-linear-regression, calculating beta0, beta1, y-cap,
104	the core concept in machine learning is represented by the equation y=f(x), where y represents labels and
105	the lecture covered data-science problem-solving using the crisp-dm-framework, a six-step cyclical
106	the lecture began with a review of expectation algebra, standard-error calculation using a single sample,
107	the lecture began with a review of statistical-significance and the distinction between statistically-
108	multiple-linear-regression (mlr) has a closed-form-solution but is impractical due to matrix-inversion

This heatmap illustrates the overlap of ideas across sessions(of representative vectors genreated). It reveals that some sessions show strong correlations, suggesting they covered broader topics, while others exhibit minimal correlation, indicating they focused on more specific, niche subjects.



05. Cluster Visualizations

Cluster Keyword Visualization leverages TF-IDF to surface the important terms in each cluster summary. We plot every cluster as a bubble whose area is proportional to the count of its high-scoring keywords. Clicking a bubble instantly generates:

- A Session-Local Word Cloud (term size ∝ TF-IDF in that cluster)
- A Global Word Cloud (term size ∝ TF-IDF across all clusters, filtered to that cluster's top words)

Wordcloud of keywords(via TD-IDF) on the whole summary data

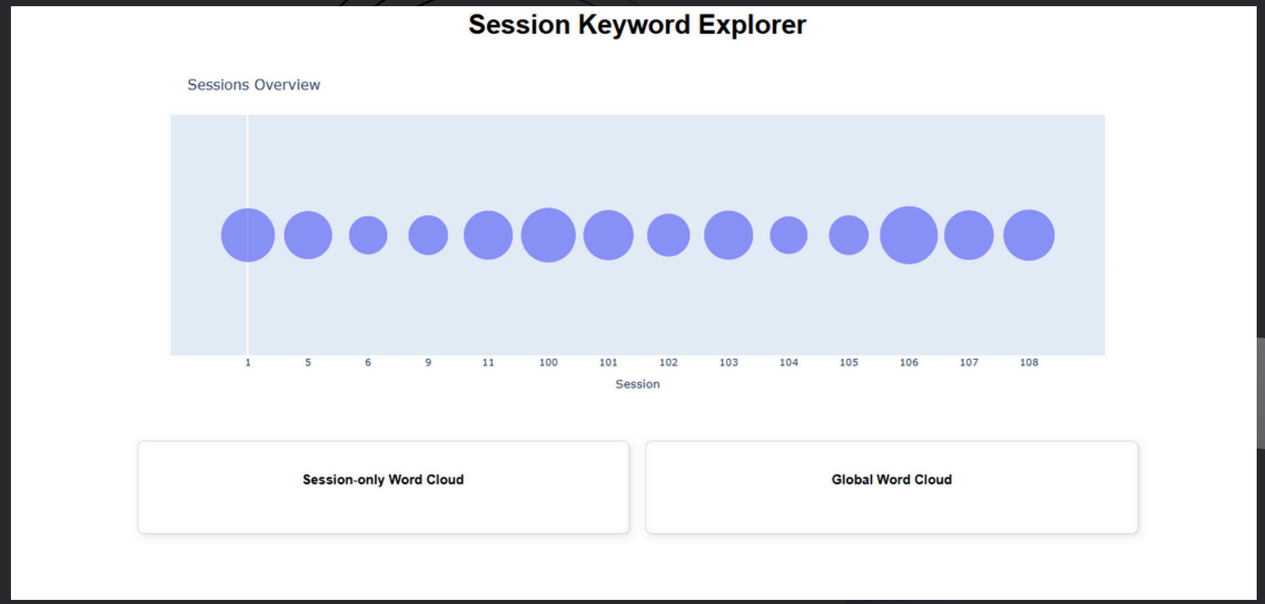
Interactive Interface for visualization

The interface opens up as shown in the figure which has bubbles sized by keyword count let you immediately see which clusters have more keywords.

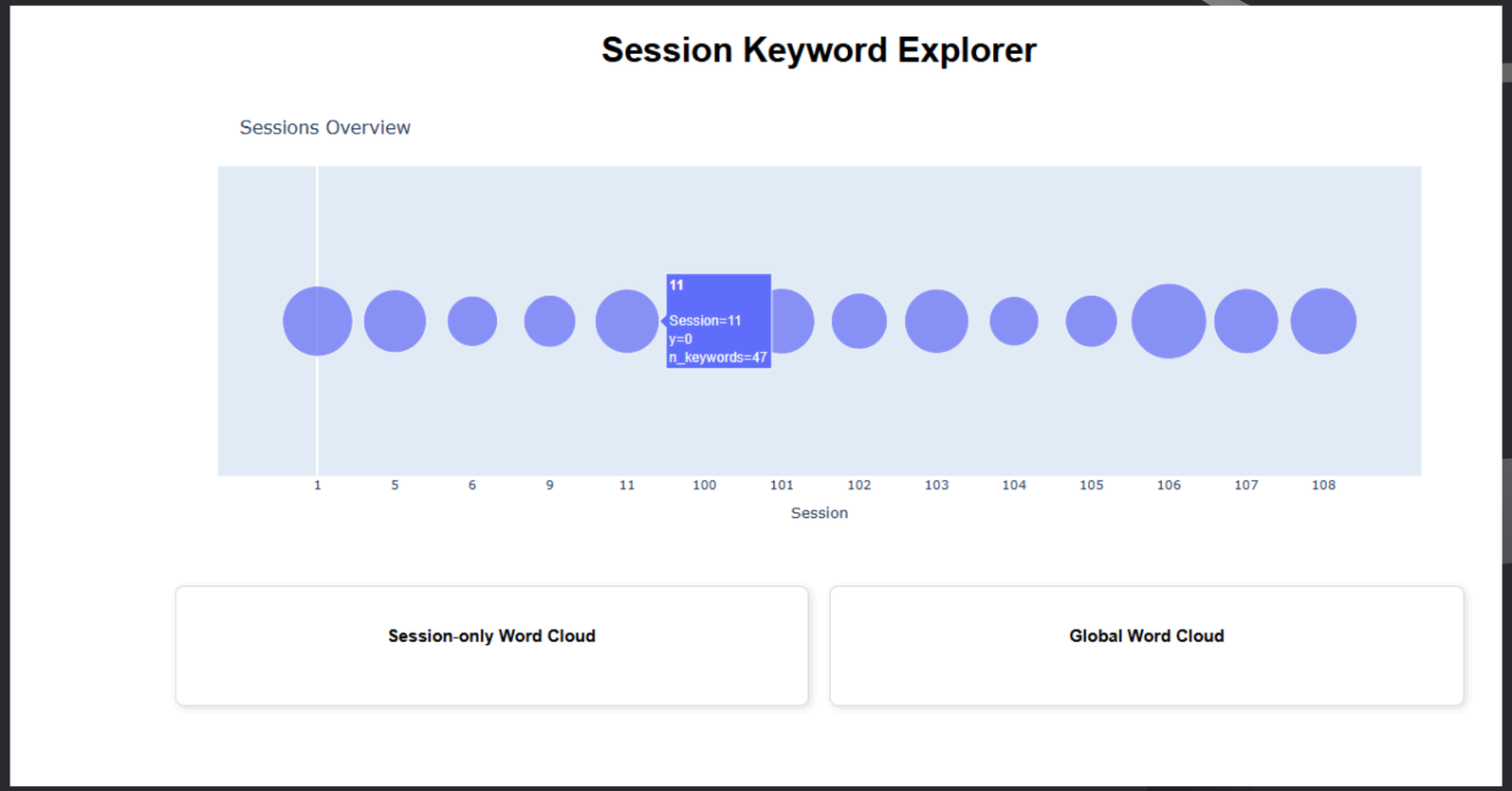
Step-1: Hover your cursor to know the number of keywords.

Step-2: Click the session you want the wordcloud

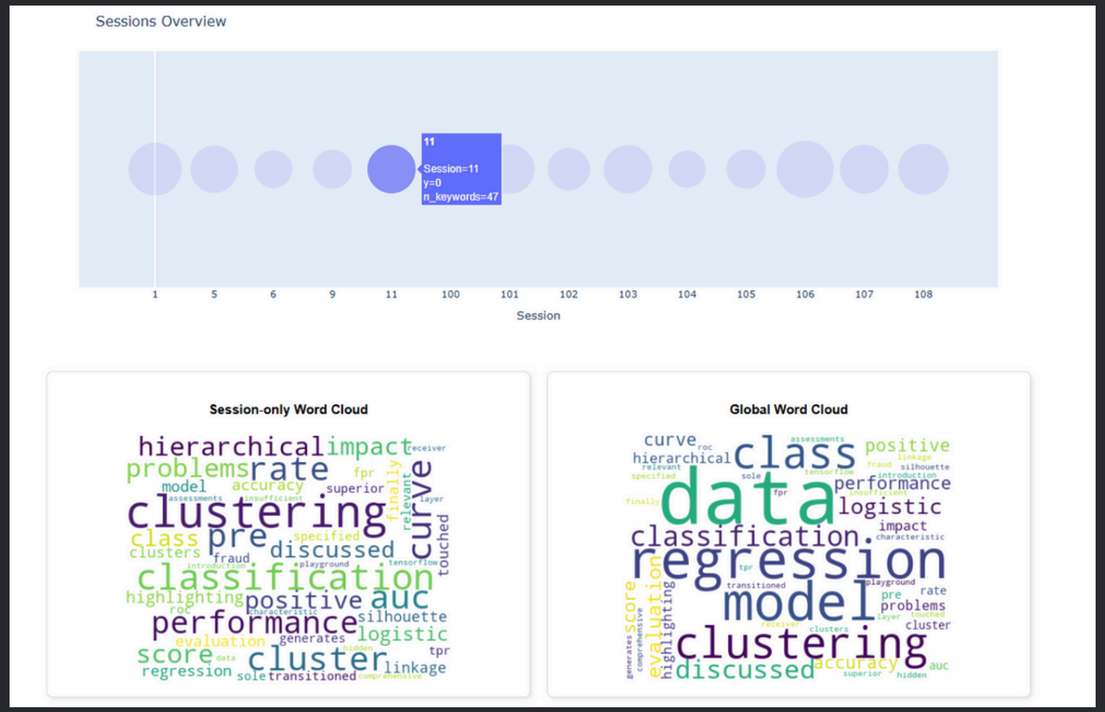
Step-3: Click another one to check the results for it



The UI when the cursor hovers over a circle

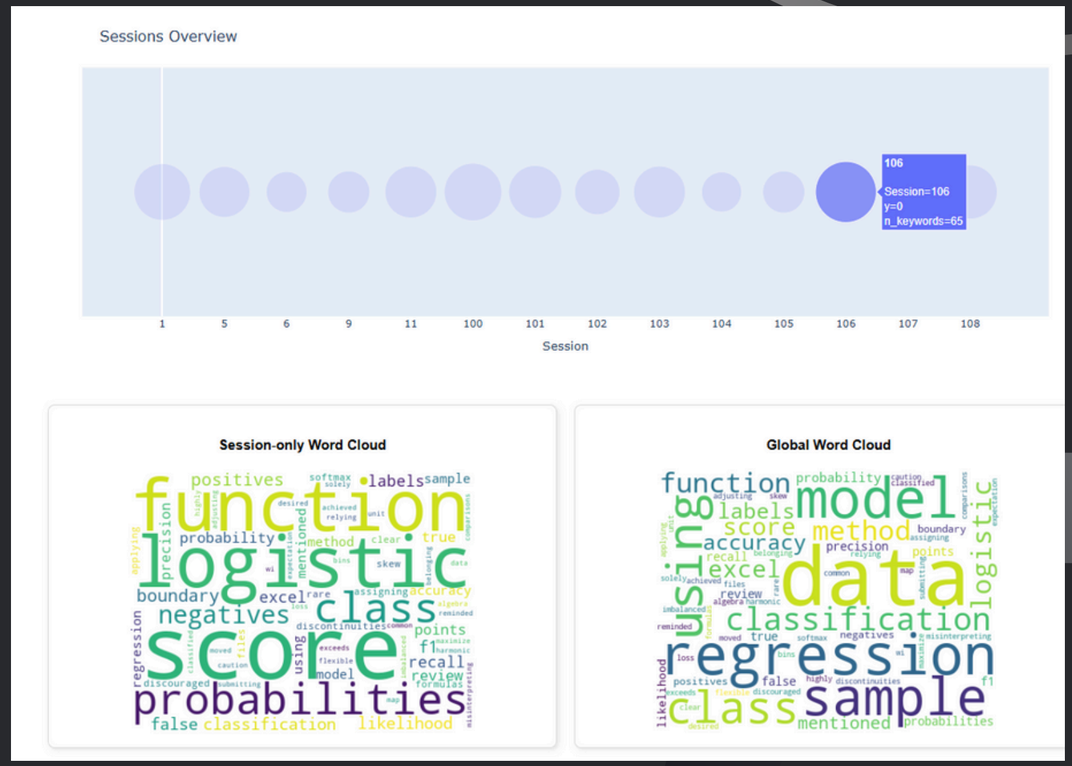


The Results after clicking a circle



Cluster ID 11

Cluster ID 106



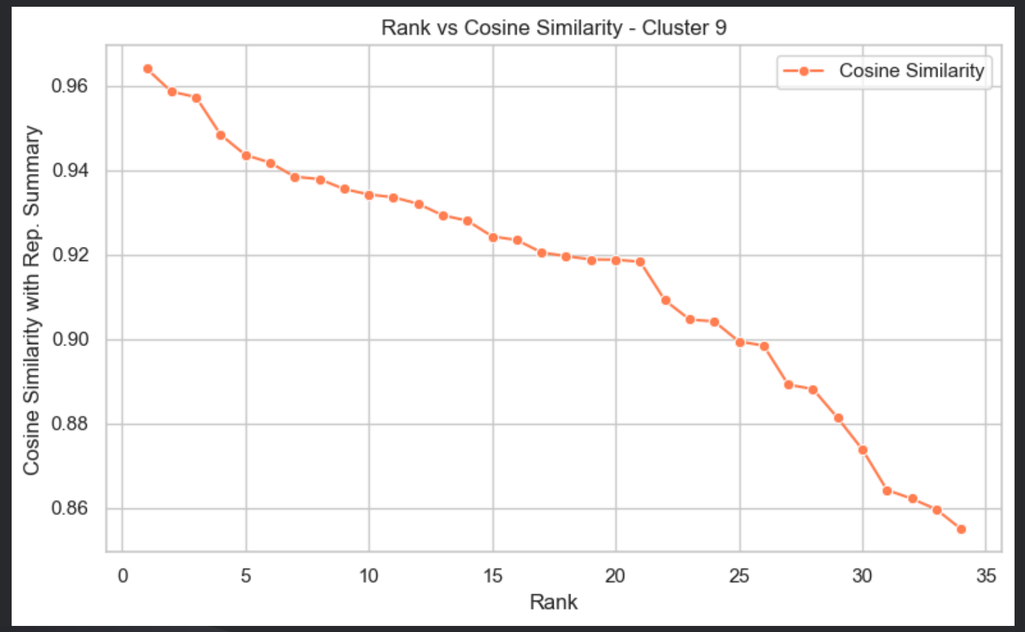
06. Ranking Summaries within a Session

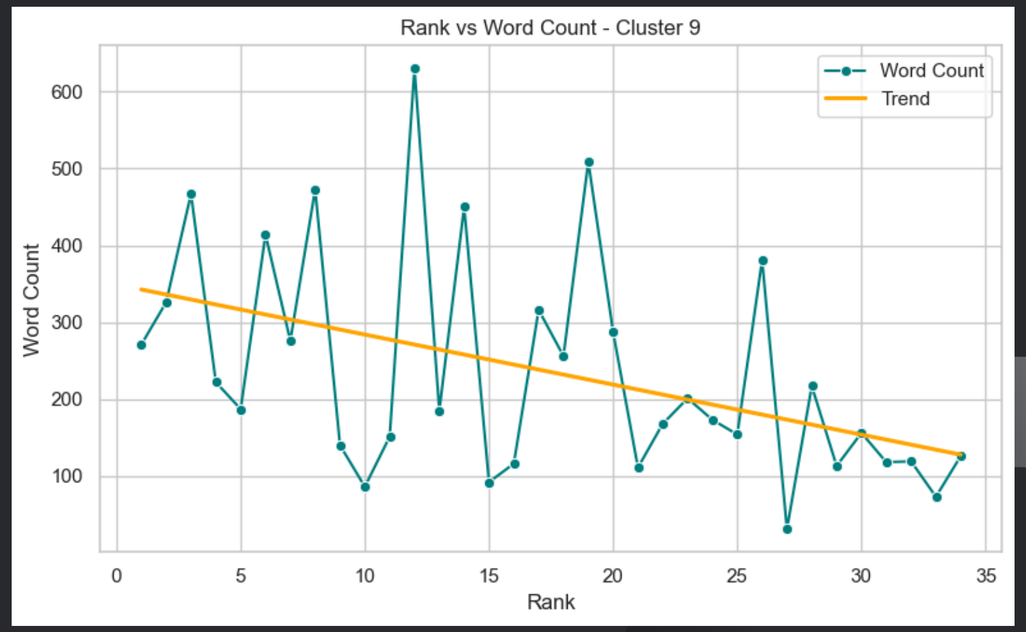
The following steps were carried to rank summaries within a session

- Step 1: Computed vector embeddings of the representative summaries.
- Step 2: Calculated cosine similarity between the representative summary and each student summary in the session.
- Step 3: Ranked the summaries based on similarity scores (higher = more detail and relevance).
- Step 4: Exported the ranked summaries along with scores into a JSON file with all the necessary information like session_id, summary_id, similarity scores and the text.

Analysis of Summary Ranking

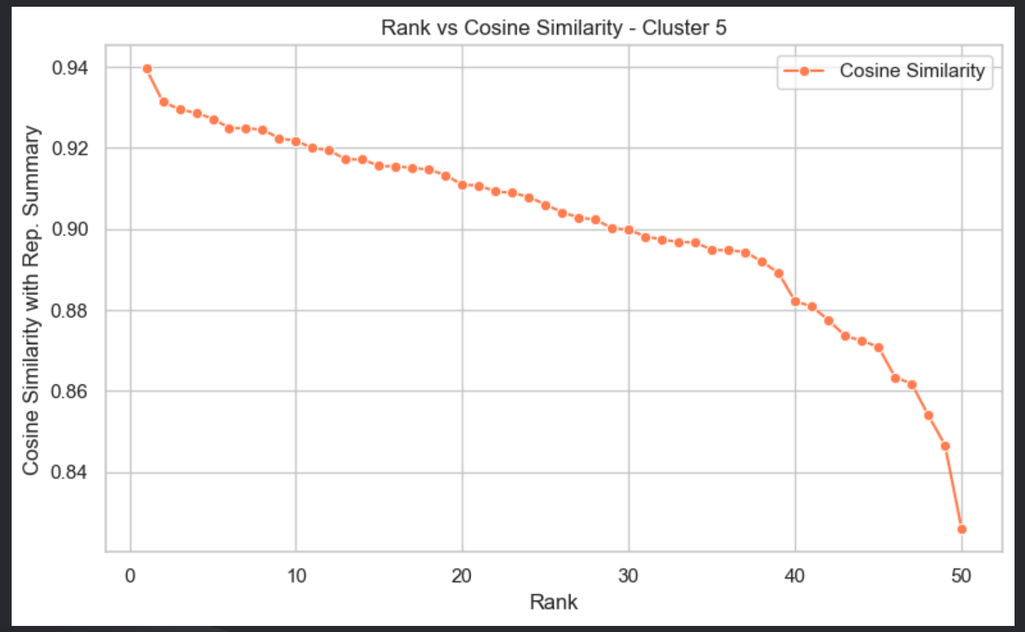
Cluster id-9

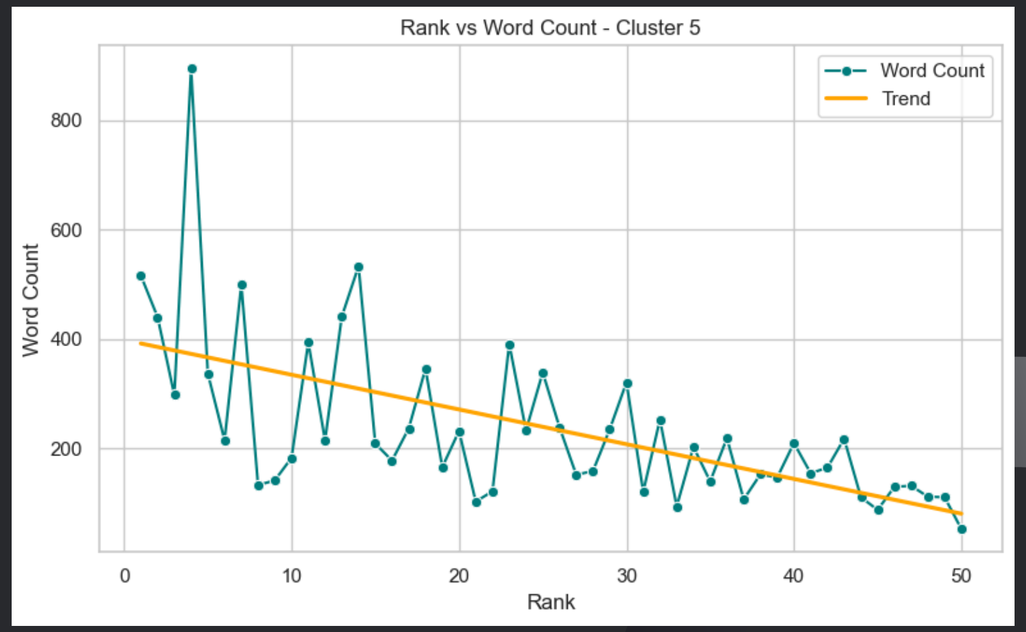




Analysis of Summary Ranking

Cluster ID-5



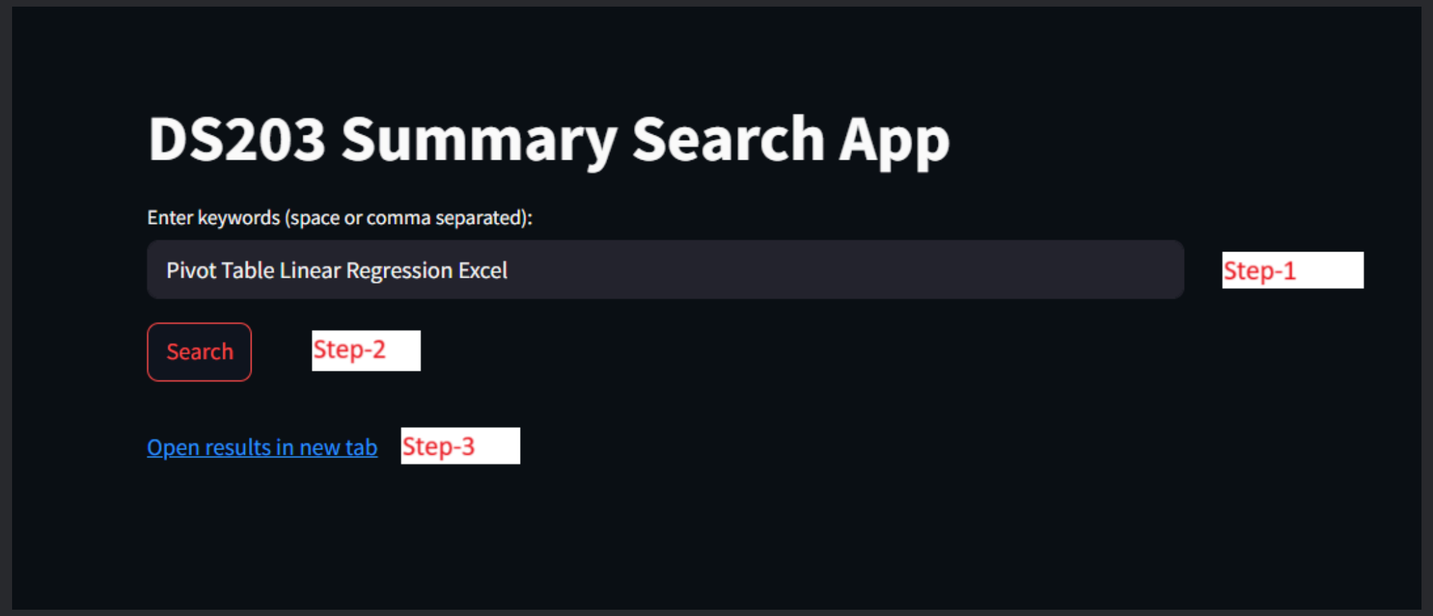


Inferences from Analysis

1. **High-ranked summaries are not always lengthy:** Some top summaries are concise yet capture the session comprehensively.
2. **Low-ranked summaries are not always short:** Longer summaries that lack clear structure and fail to cover key ideas tend to score lower.
3. **General trend:** There is an overall downward trend, indicating that higher-ranked summaries often have more words, as more content generally allows for richer and more complete information.

These observations indicate that the ranking method is robust, effectively **assessing detail and relevance** independent of summary length.

07. Summary Search App

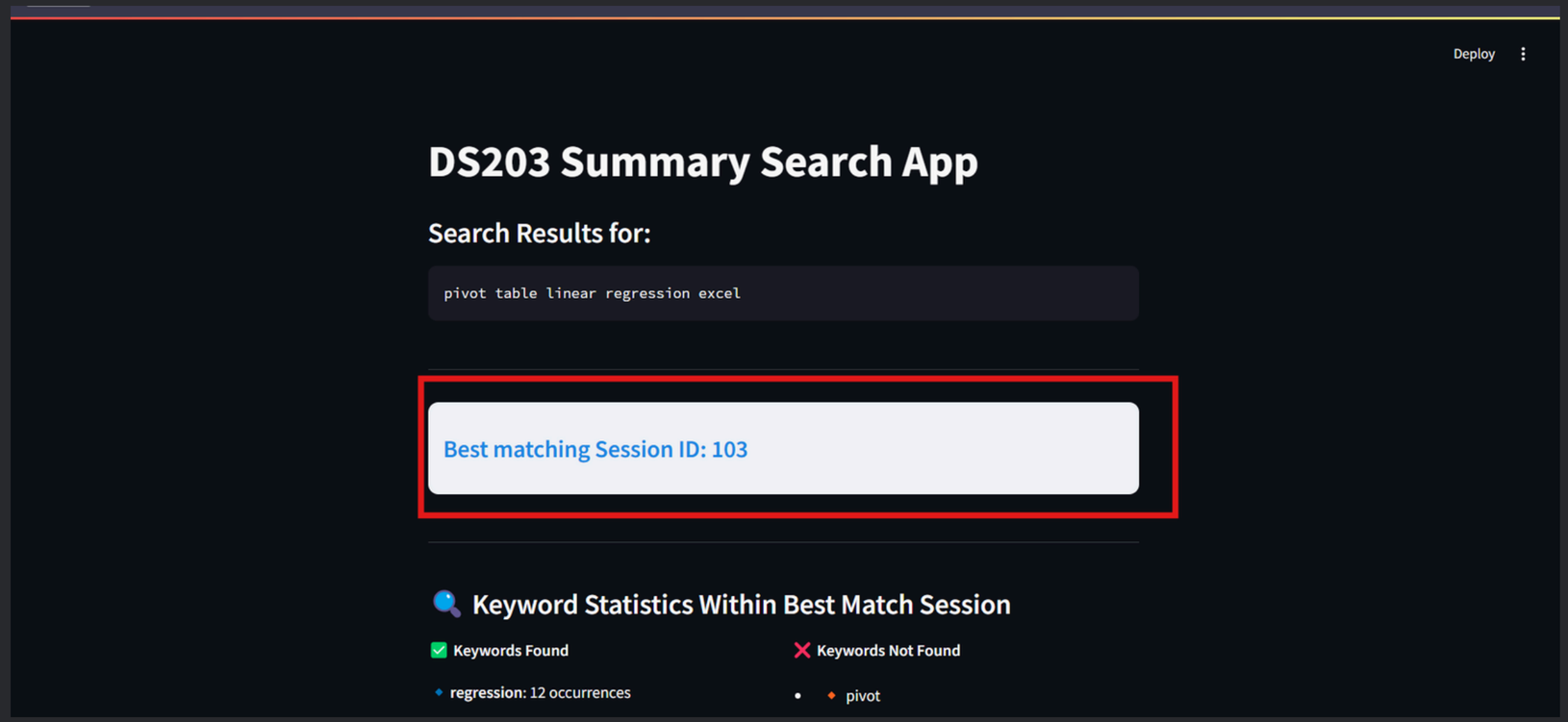


Step-1: Enter the Keywords

Step-2: Click on Search

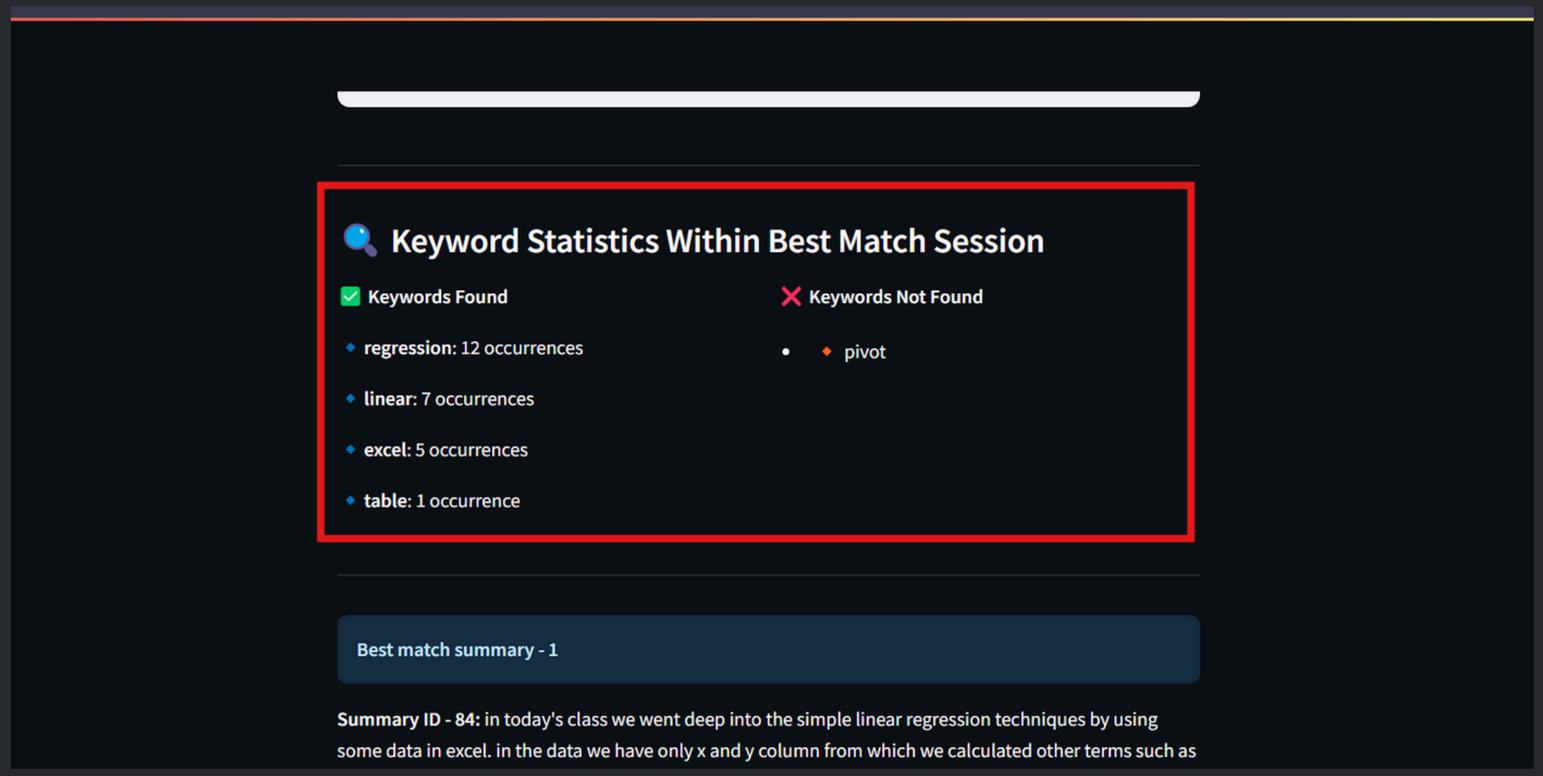
Step-3: Click on Results tab

Result page



Highlighted section shows Session ID of Best Match Session

Result page



Highlighted section shows Keyword Statistics of best match session

Best match summary - 1

Summary ID - 84: in today's class we went deep into the simple linear regression techniques by using some data in excel. in the data we have only x and y column from which we calculated other terms such as x_bar, y_bar, xbar_sq, ybar_sq, error and many more terms. we also create the scatter plot between x and y and realized that it follows linear model. we have also plot the histogram to check what is the nature of distribution of data. if it's a bell shaped curve then it is good. the scatter plot of errors values display a distinct pattern showing that model has failed to pick-up the inherent pattern in the data. next by using the data analysis tool in excel we created the summary output of an linear regression model giving many values related to regression statistics. another interesting thing is that - one that explains most of the variations in the data is 'good model'. further we learnt about some regression statistics short forms like 1)sst = measure of total variation in the given dataset. 2)ssr => total variation explained by the regression model and 3) sse => variation not explained by the model, attributed to random errors. coefficient of determination(r^2) which is the square of the correlation coefficient 'r' between x and y. if the x is increasing and y also increases then it have (+ve) correlation. if the x is increasing and y is decreasing then it have (-ve) correlation. we conduct some 'thought' experiments, related to estimating the population mean from the sample mean: assume that from a population we can take multiple good, representative samples, let's say k samples, each of size n. let's call each sample as s_i. using each s_i, we calculate its mean and call it m_i. for samples are good, representative samples of the population, they will result in means m_i that are close to each other. if we collect all the m_i and create a frequency table and a histogram, it's shape will be bell curved.

Top Ranked summary of the Best match session along with its summary ID

Lessons Learnt:

1. Training context based models from Scratch is difficult, it is rather better to use some pre-trained model and fine tune for the purpose.
2. How powerful transformer based methods are, the quality of embeddings they produce.
3. How easy catching malpractices have become using these modern tools.
4. Essence of domain knowledge, it gave a start pointing for cluster count in this problem.
5. How to use API and get information from other websites using Python.
6. Clearly defining and writing the problem helps a lot in finding solution to those. The case where we defined the problem with the summary text gave us the idea of representative summaries.

THANK YOU