

AOML

# Movie Similarity Prediction

J013: Khushi Daga

J005: Siddh Baldha



# Learning Goals

1

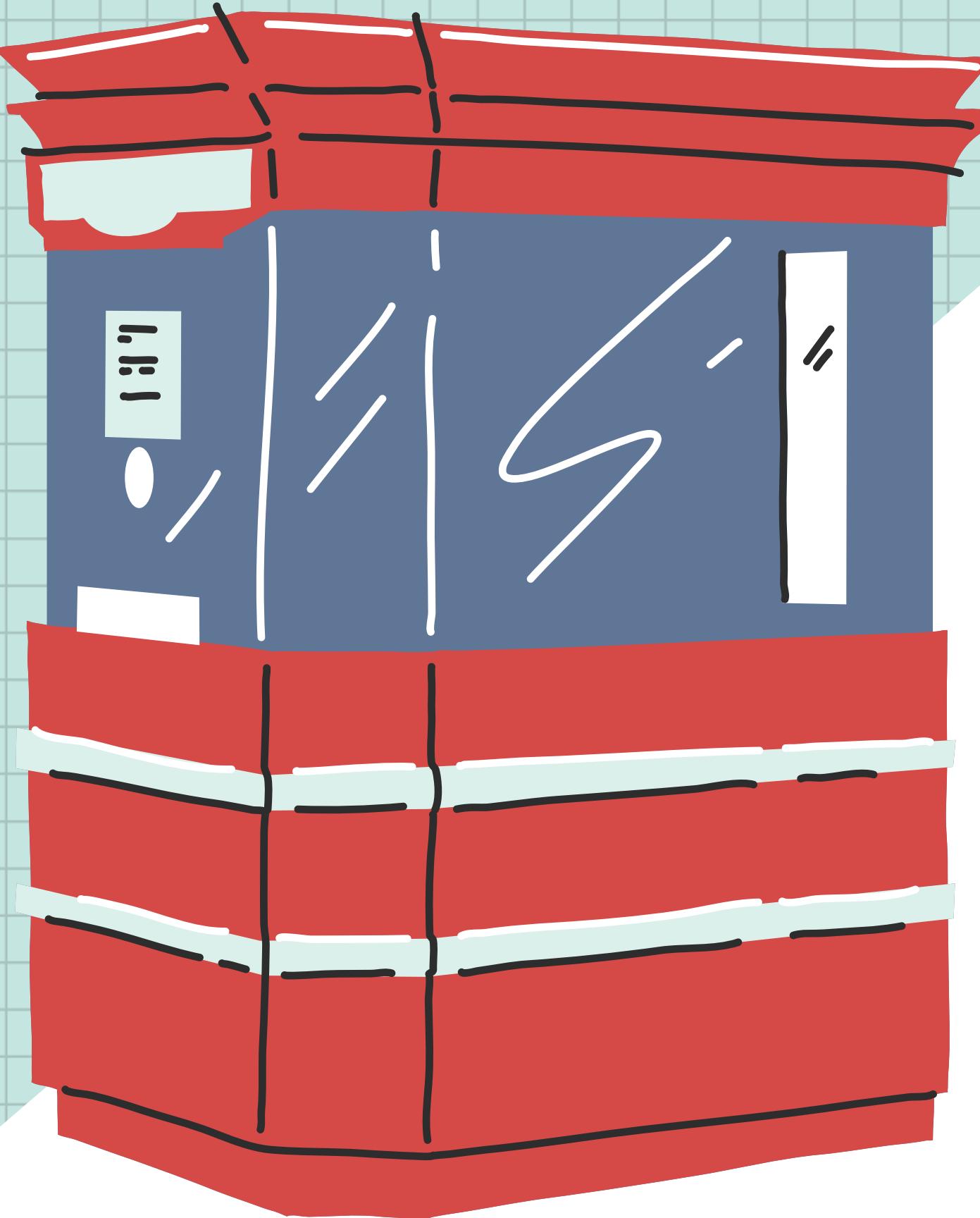
We will quantify the similarity of movies based on their plot summaries available on IMDb and Wikipedia.

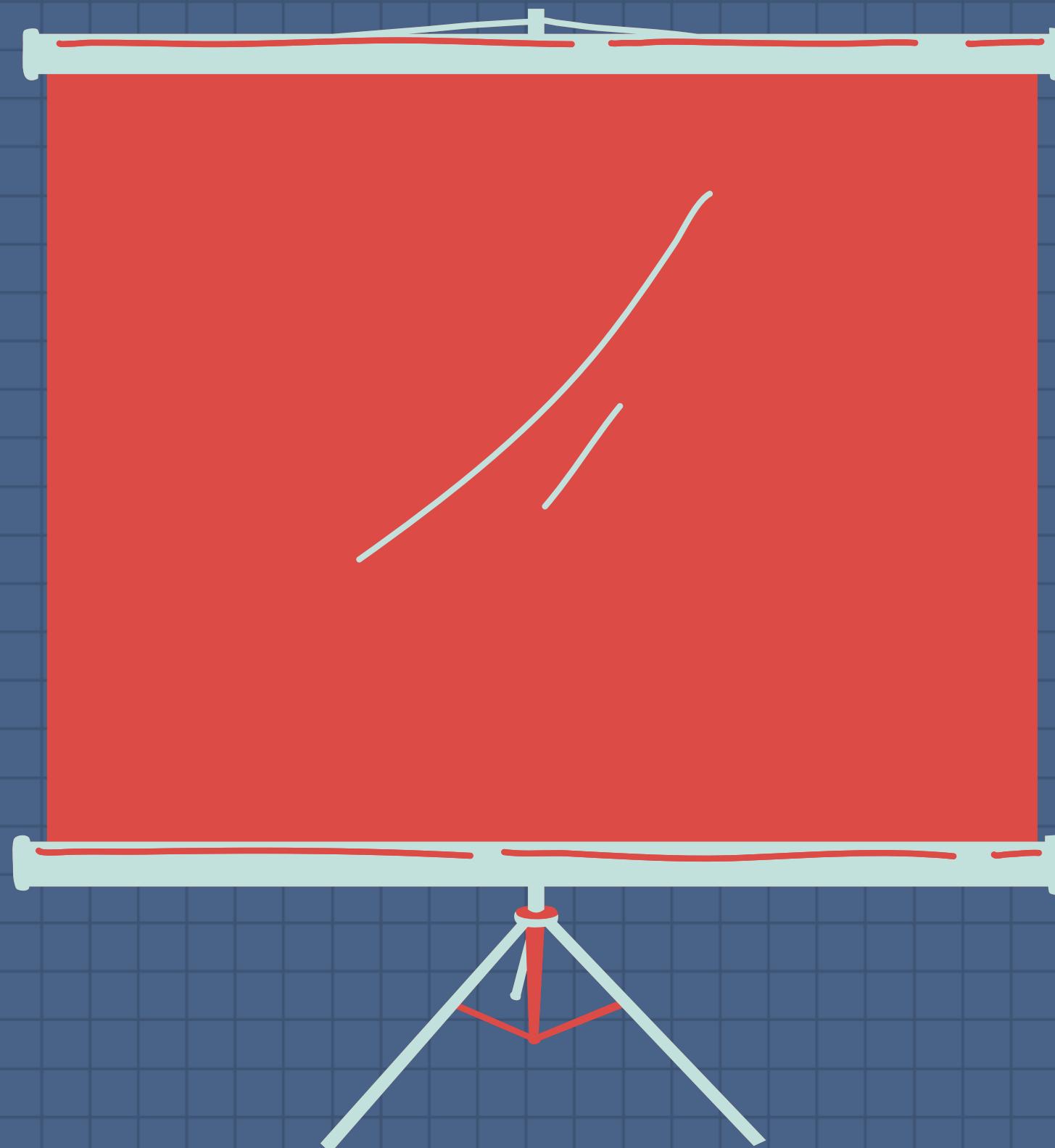
2

We will then separate them into groups and we'll create a dendrogram to represent how closely the movies are related to each other.

# Contents

- 1 Import, Observe & Combine Dataset
- 2 Tokenisation
- 3 Stemming
- 4 Create and Fit TfIdfVectorizer
- 3 Create Cluster
- 4 Calculate Similarity Distance
- 4 Conclusion and Visualization





# Import, Observe & Combine Dataset



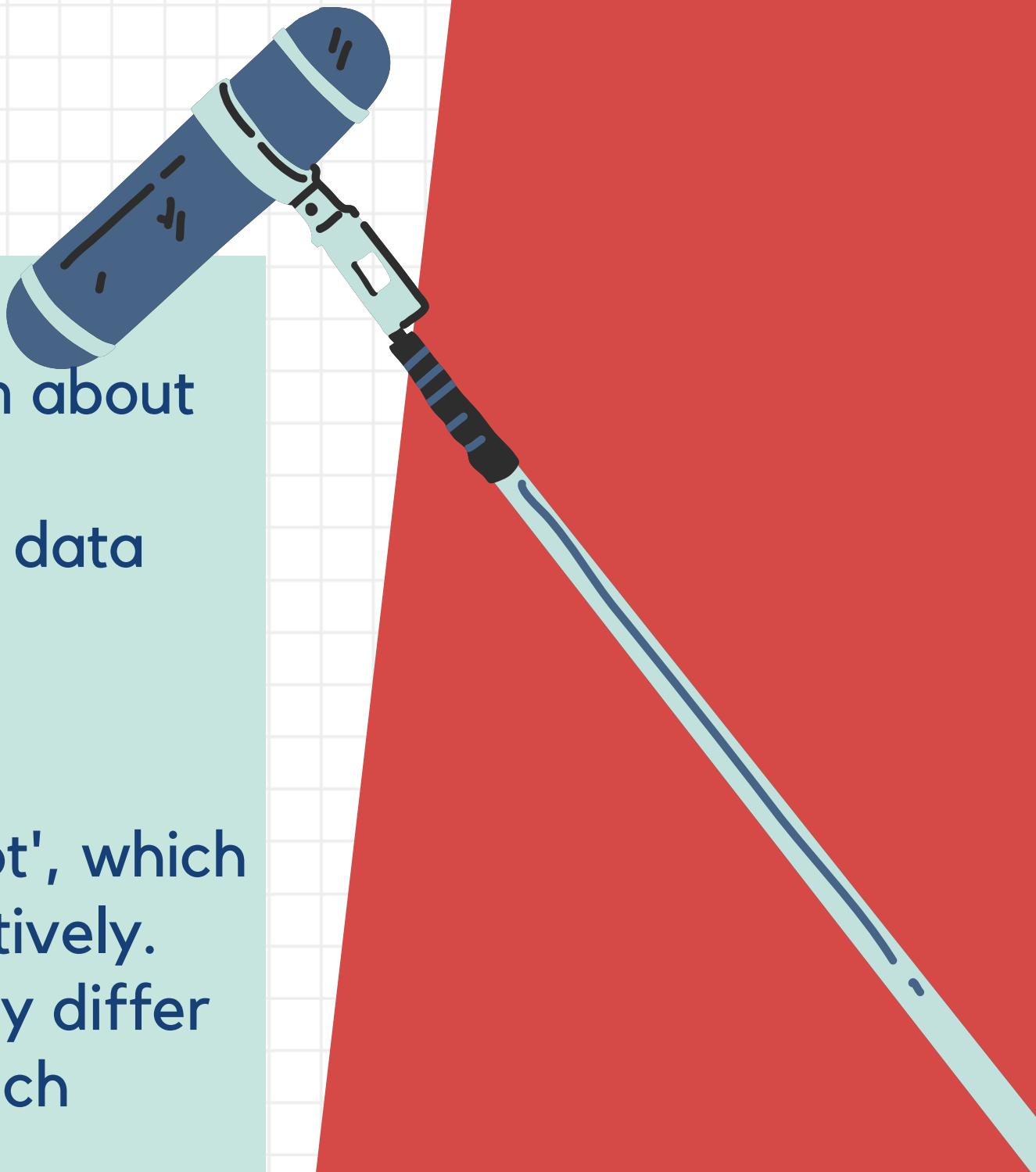
# Import and Observe

## 1. Importing the Dataset:

- We begin by importing the dataset that contains information about movie plot summaries from both IMDb and Wikipedia.
- This dataset is crucial for our analysis as it provides the raw data we'll be working with, it has data of 100 movies.

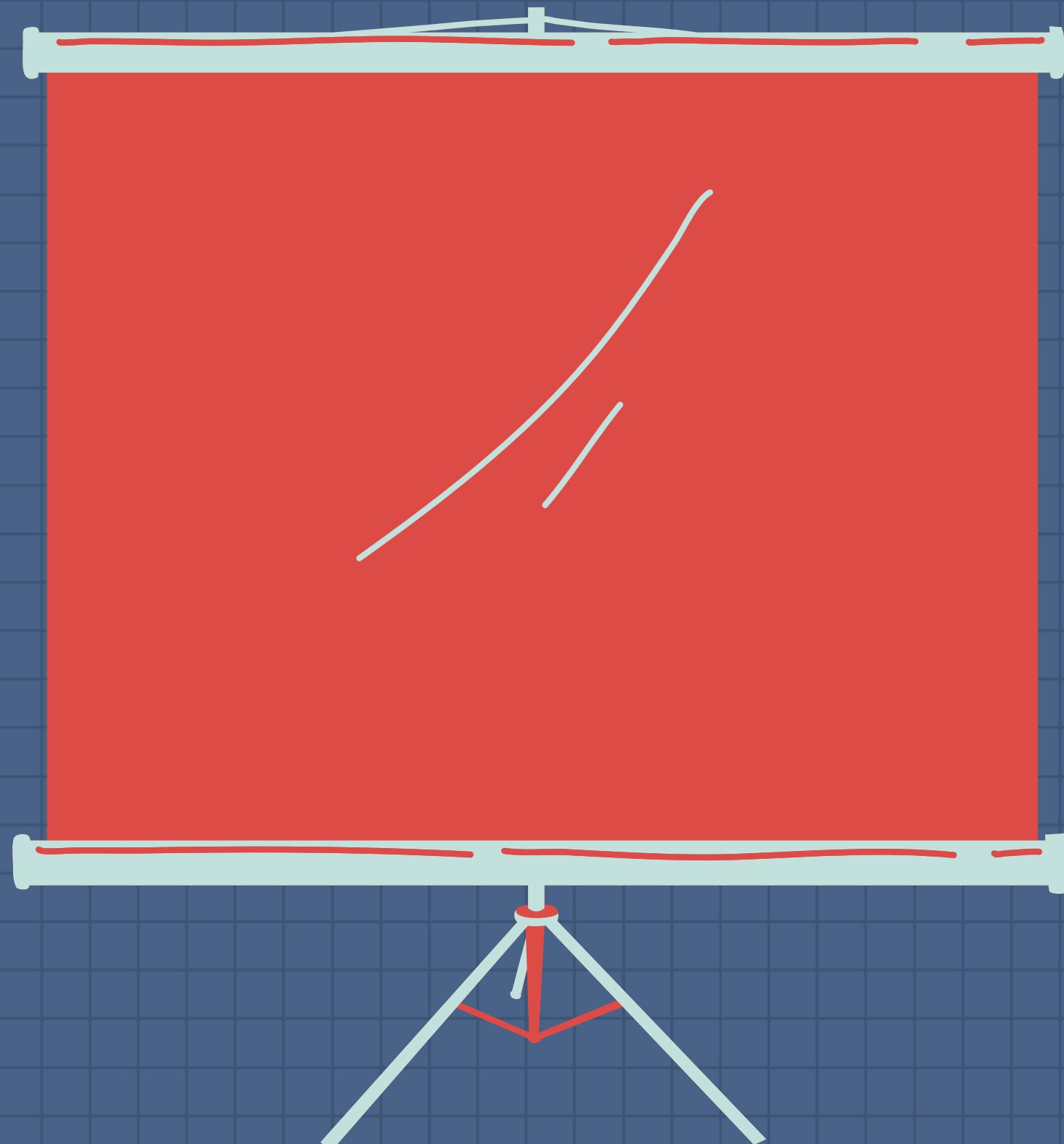
## 2. Observing the Data:

- The dataset contains two columns: 'wiki\_plot' and 'imdb\_plot', which represent plot summaries from Wikipedia and IMDb, respectively.
- Although these columns contain similar information, they may differ in tone and expression, providing unique perspectives on each movie's plot.
- Observing the data allows us to grasp its structure, identify any inconsistencies, & gain insights into potential patterns.



### 3. Combining Plot Summaries:

- To streamline our analysis and avoid redundancy, we'll combine the plot summaries from both 'wiki\_plot' and 'imdb\_plot' into a single column.
- This combined column, let's call it 'combined\_plot', merges information from both sources, providing a comprehensive overview of each movie's plot.
- By consolidating the plot summaries, we eliminate the need to process extra columns, reducing computational overhead and simplifying our analysis process.
- The combined plot summaries serve as the basis for quantifying the similarity between movies and clustering them into groups based on their plot characteristics.



# Tokenization

# Definition and Steps

Tokenization is a crucial step in natural language processing that breaks down text into smaller units for analysis. It enables us to extract meaningful information from the text while considering additional filtration to remove irrelevant tokens.

## Input Text

- Let's consider a small extract from "The Godfather".
- Example: "On the day of his only daughter's wedding, Vito Corleone"

## Narrative Conventions

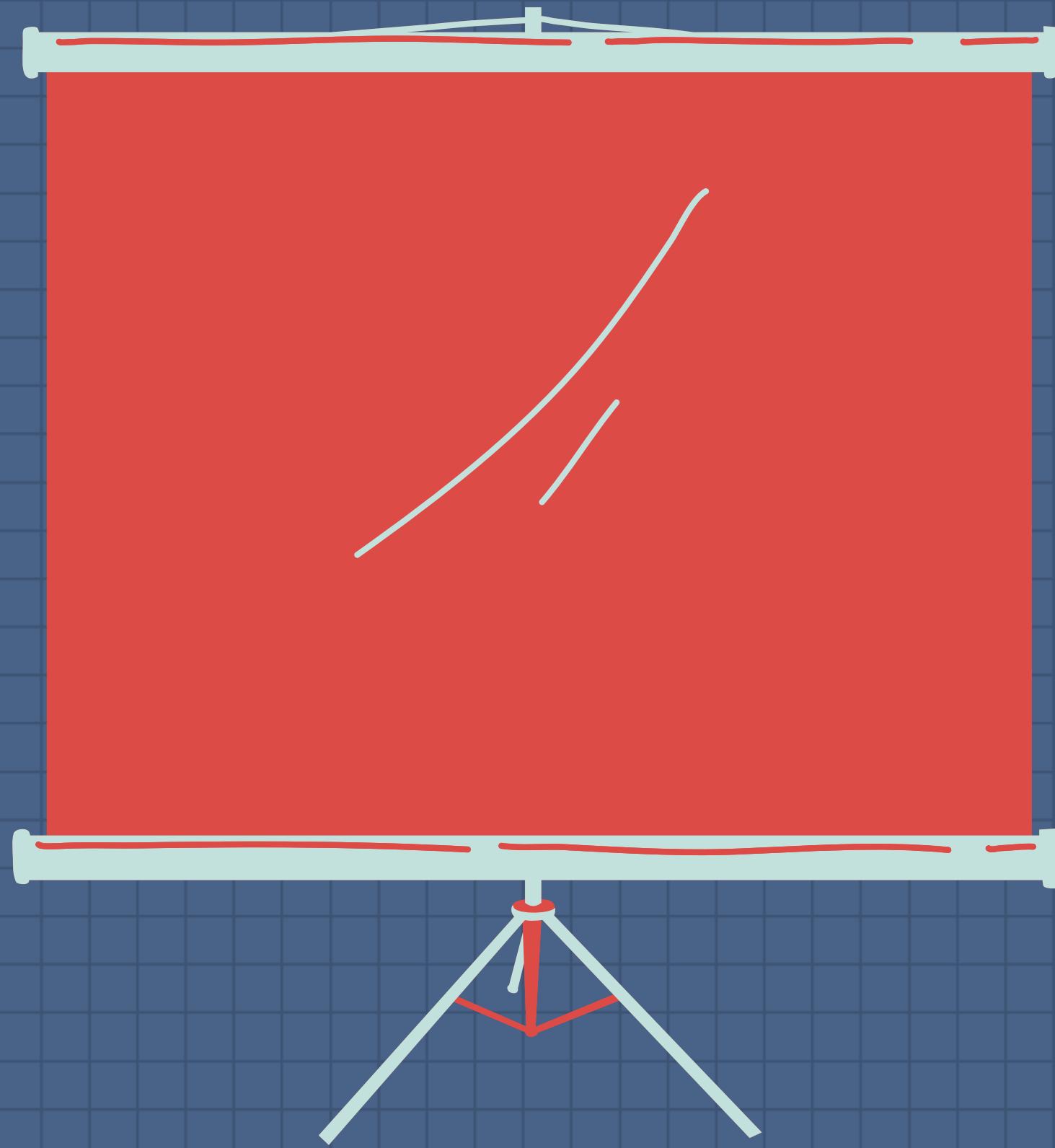
- Text can be tokenized into following words:
- "On", "the", "day", "of", "his", "only", "daughter's", "wedding", ",", "Vito", "Corleone"

## Additional Filtration

- Additional filtration to remove tokens that are entirely numeric values or punctuation.

## Context Building

- While tokenization breaks down the text into individual units, it may not always capture the context of the entire phrase or sentence.



# Steaming



# Definition



**Stemming involves reducing words to their base or root form, disregarding variations like tense, plurality, or suffixes. For instance, words like 'fishing', 'fished', and 'fisher' all get stemmed to the word 'fish', simplifying the representation of these variations.**

# Process

- Instead of treating 'witnesses' and 'witness' as separate dictionary entries, stemming reduces them both to their root form, which is 'wit'.
- Stemming algorithms analyze the words and remove suffixes or prefixes to find the root form.
- In this case, both 'witnesses' and 'witness' would be stemmed to 'wit', simplifying the representation and enabling easier analysis.

## Stemming Algorithm:

- The Snowball Stemmer, known for its versatility and language support, is chosen for its effectiveness in stemming words across different languages and contexts.

### Example

- "Young William Wallace witnesses the treachery of Longshanks" ~ Gladiator
- "escapes to the city walls only to witness Cicero's death" ~ Braveheart
- In both sentences, the words 'witnesses' and 'witness' convey the same meaning, differing only in grammatical form.

# Club together Tokenize & Stem

**Wrapping tokenization and stemming into a single function and using it as the tokenizer argument in TF-IDF vectorization simplifies the preprocessing pipeline and enhances efficiency and consistency in text data processing tasks.**

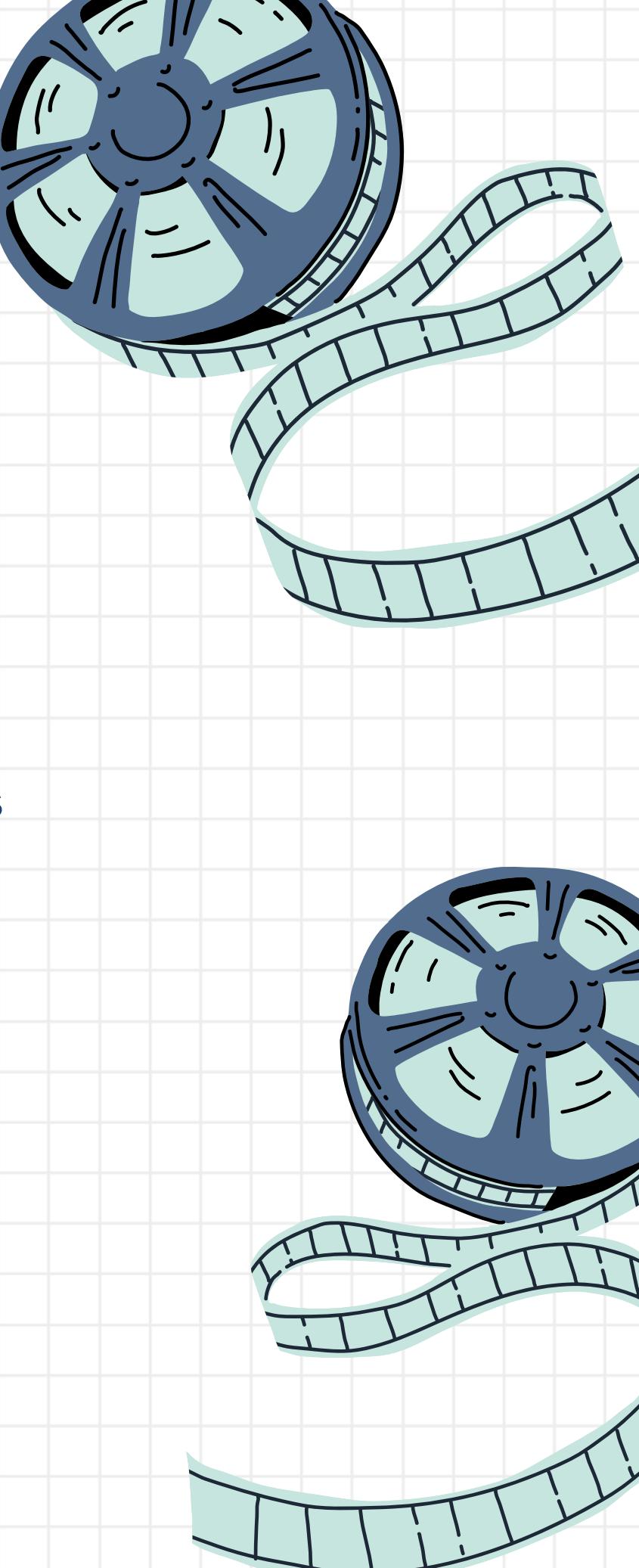
Consider the sentence from the plot of The Godfather: "Today (May 19, 2016) is his only daughter's wedding." If we do a 'tokenize-only' for this sentence, we have the following result:

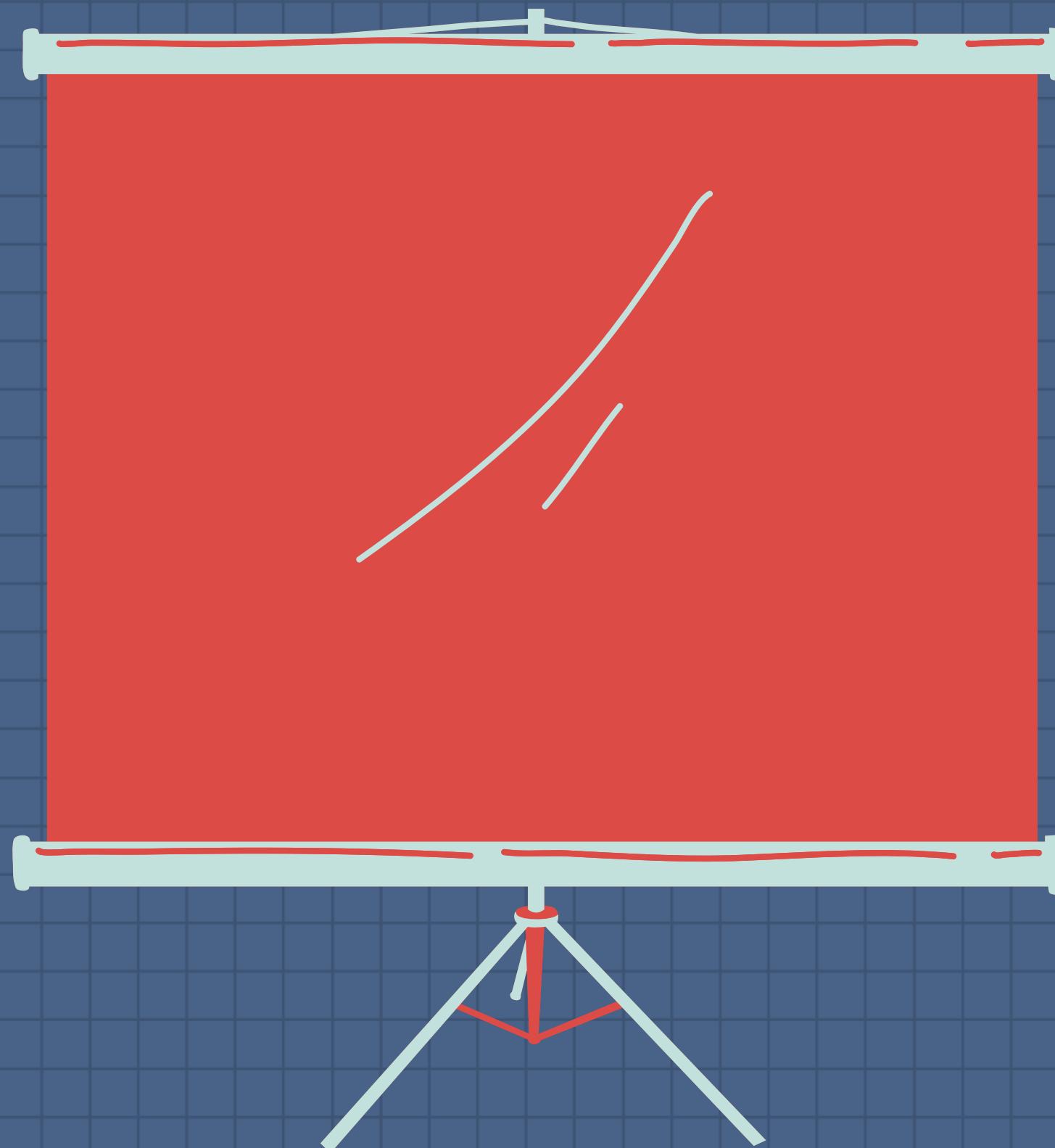
'today', 'may', 'is', 'his', 'only', 'daughter', "'s", 'wedding'

But when we do a 'tokenize-and-stem' operation we get:

'today', 'may', 'is', 'his', 'onli', 'daughte', "'s", 'wed'

All the words are in their root form, which will lead to a better establishment of meaning as some of the non-root forms may not be present in the NLTK training corpus.

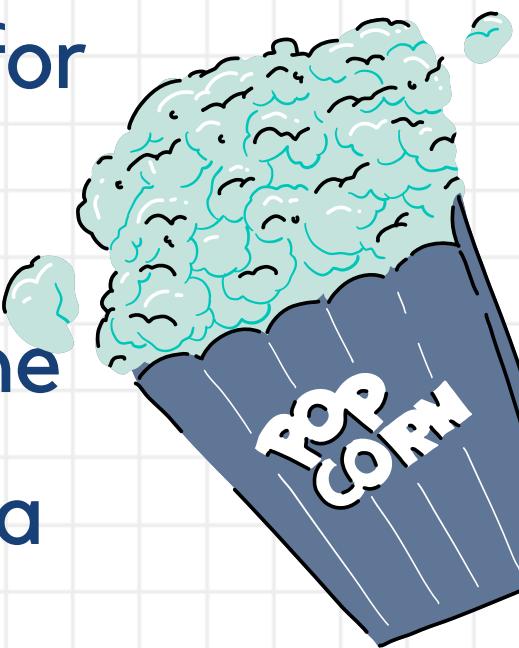


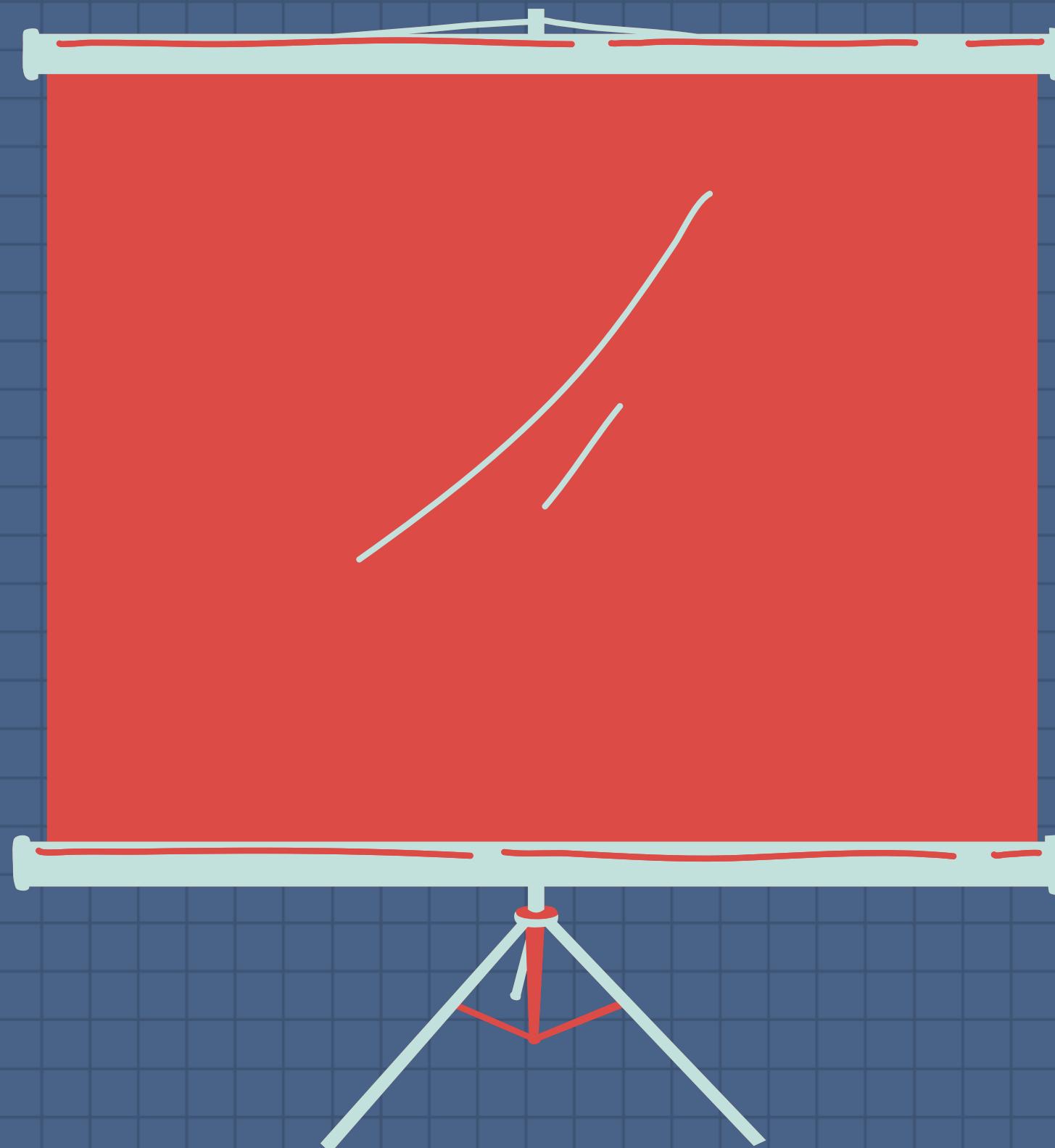


Create  
TfidfVectorizer



- Textual plot summaries need to be converted into numerical representations for computers to process effectively.
- One method to achieve this is by using the 'CountVectorizer', which counts the occurrences of each word in the entire vocabulary and returns the counts in a vector.
- However, 'CountVectorizer' has a limitation in that it treats all words equally, even common words that may not be important for understanding the plot's theme.
- To overcome this limitation, the Term Frequency-Inverse Document Frequency method is introduced.
- TF-IDF combines two metrics: Term Frequency measures how often a word appears in a document, while Inverse Document Frequency reduces the importance of words that appear frequently across multiple documents.
- TF-IDF recognizes words that are significant and distinctive within a particular document, making it a valuable tool for text analysis and information retrieval tasks.





Create  
Clusters



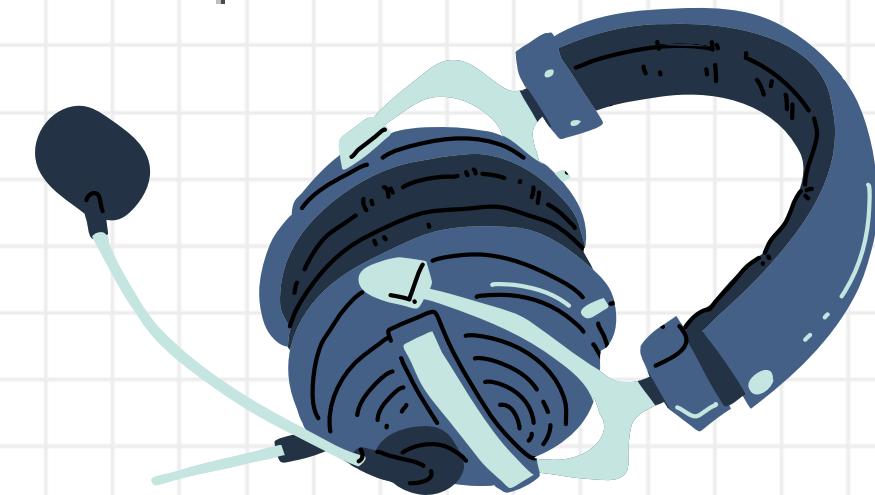
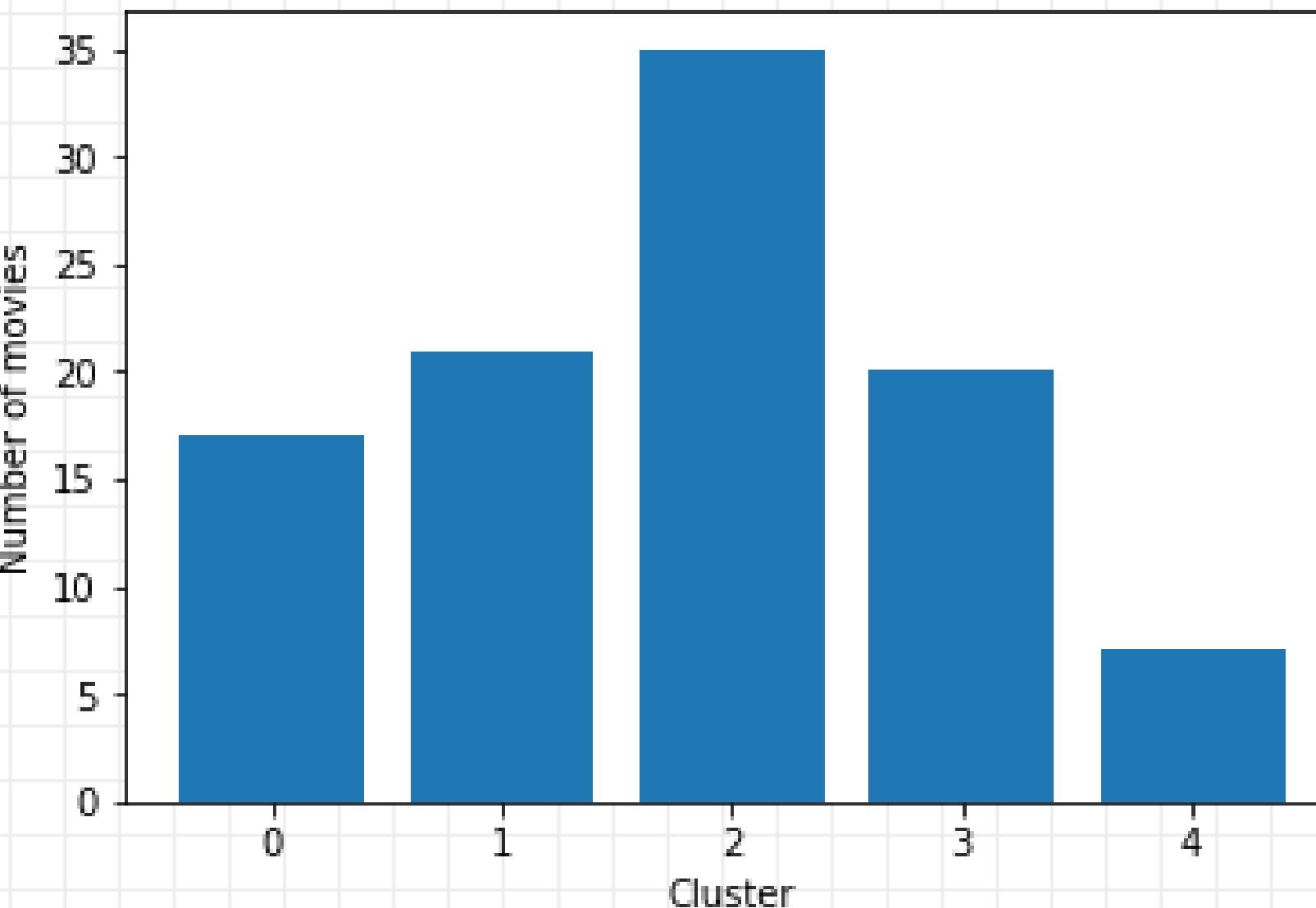


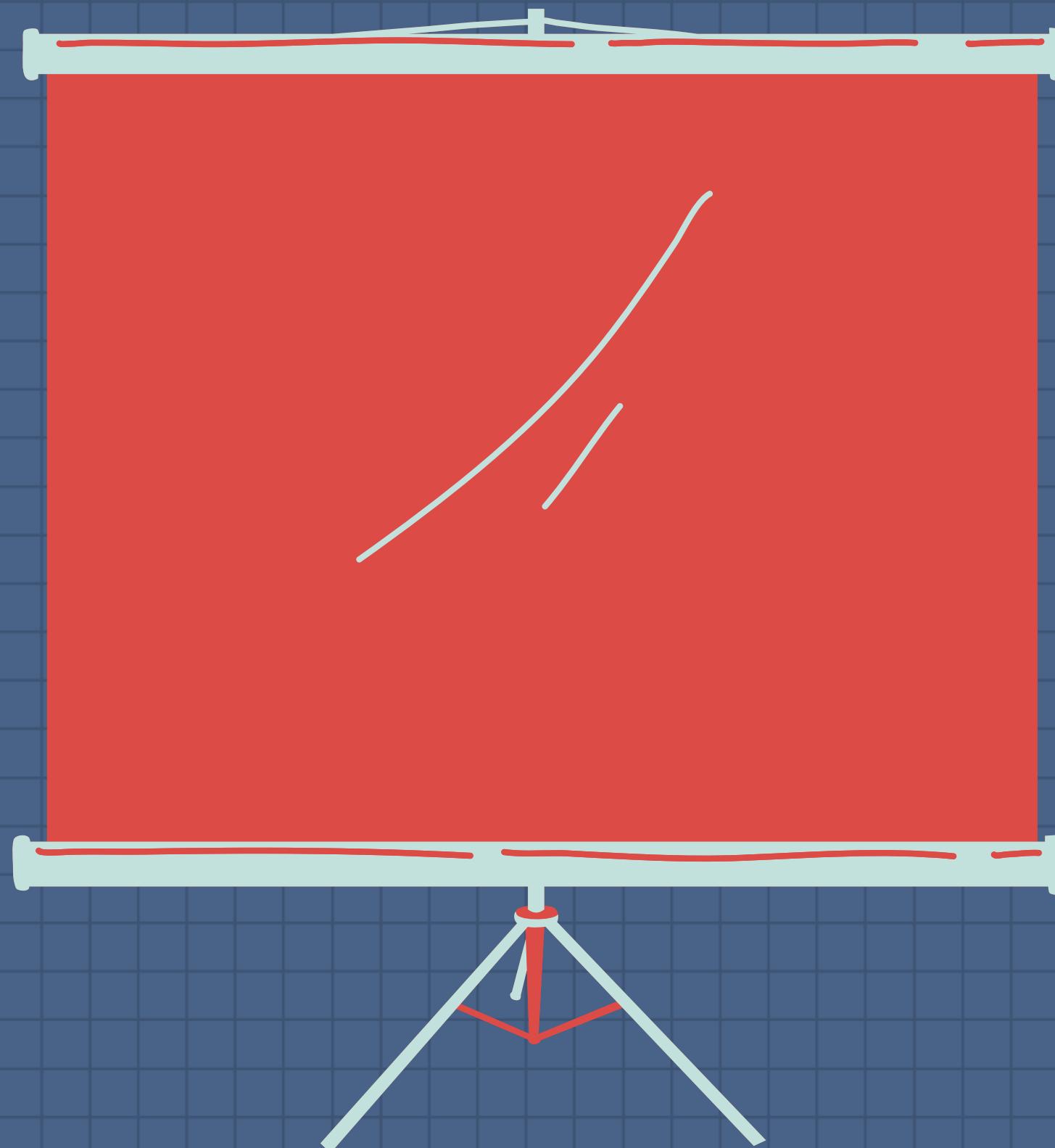
## Defination:

Clustering is the method of grouping together a number of items such that they exhibit similar properties

K-means is an algorithm which helps us to implement clustering in Python.

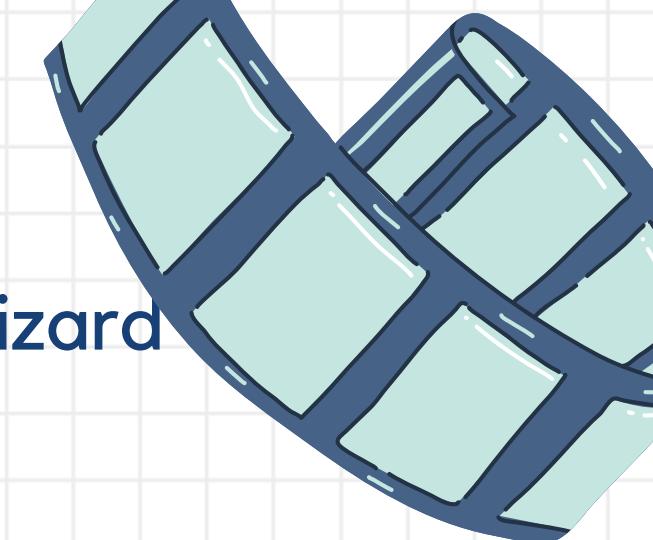
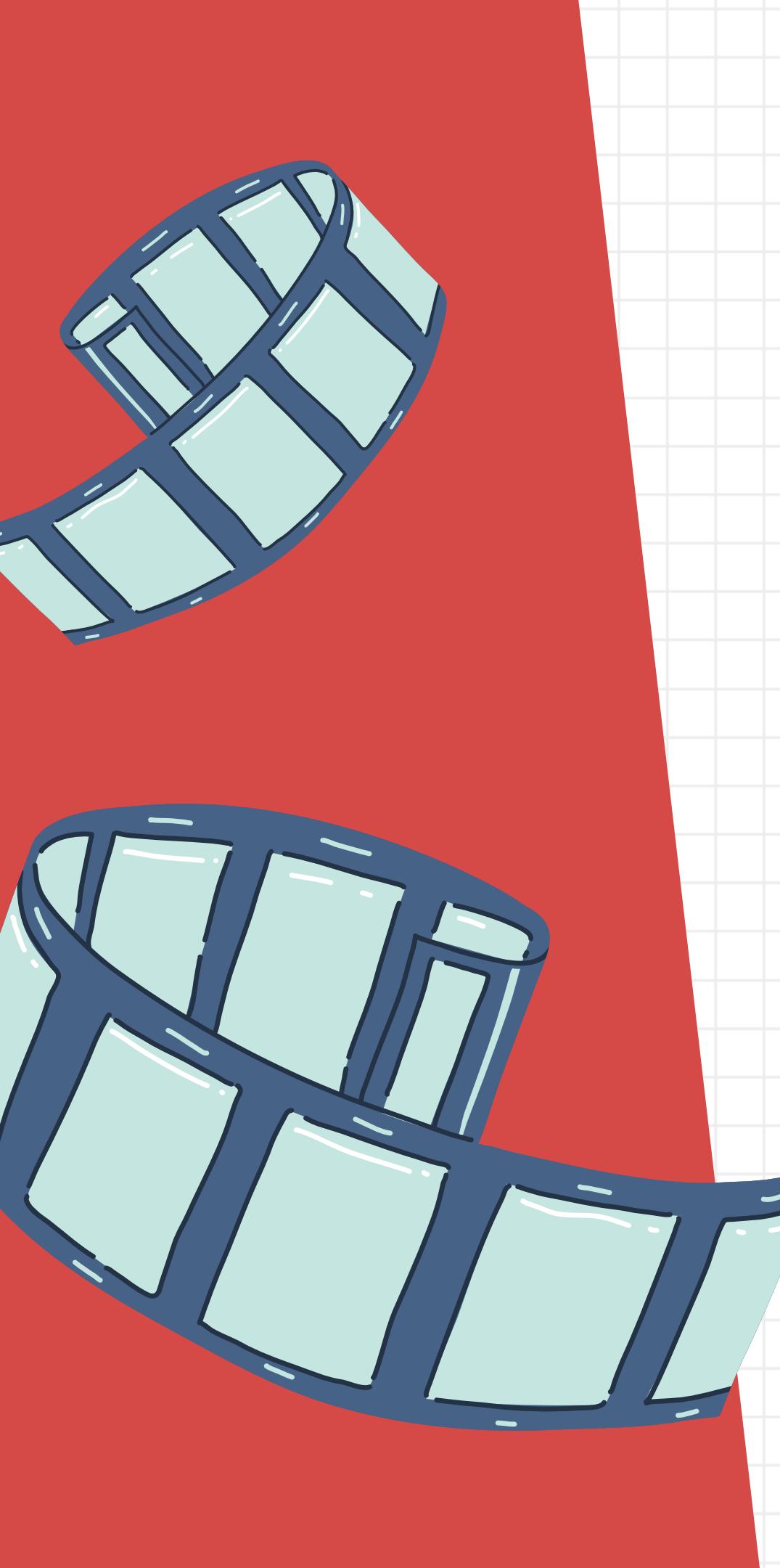
- The given sample is divided into K clusters where each cluster is denoted by the mean of all the items lying in that cluster.





Calculate  
Similarity  
Distance





Consider the following two sentences from the movie The Wizard of Oz:

"they find in the Emerald City"

"they finally reach the Emerald City"

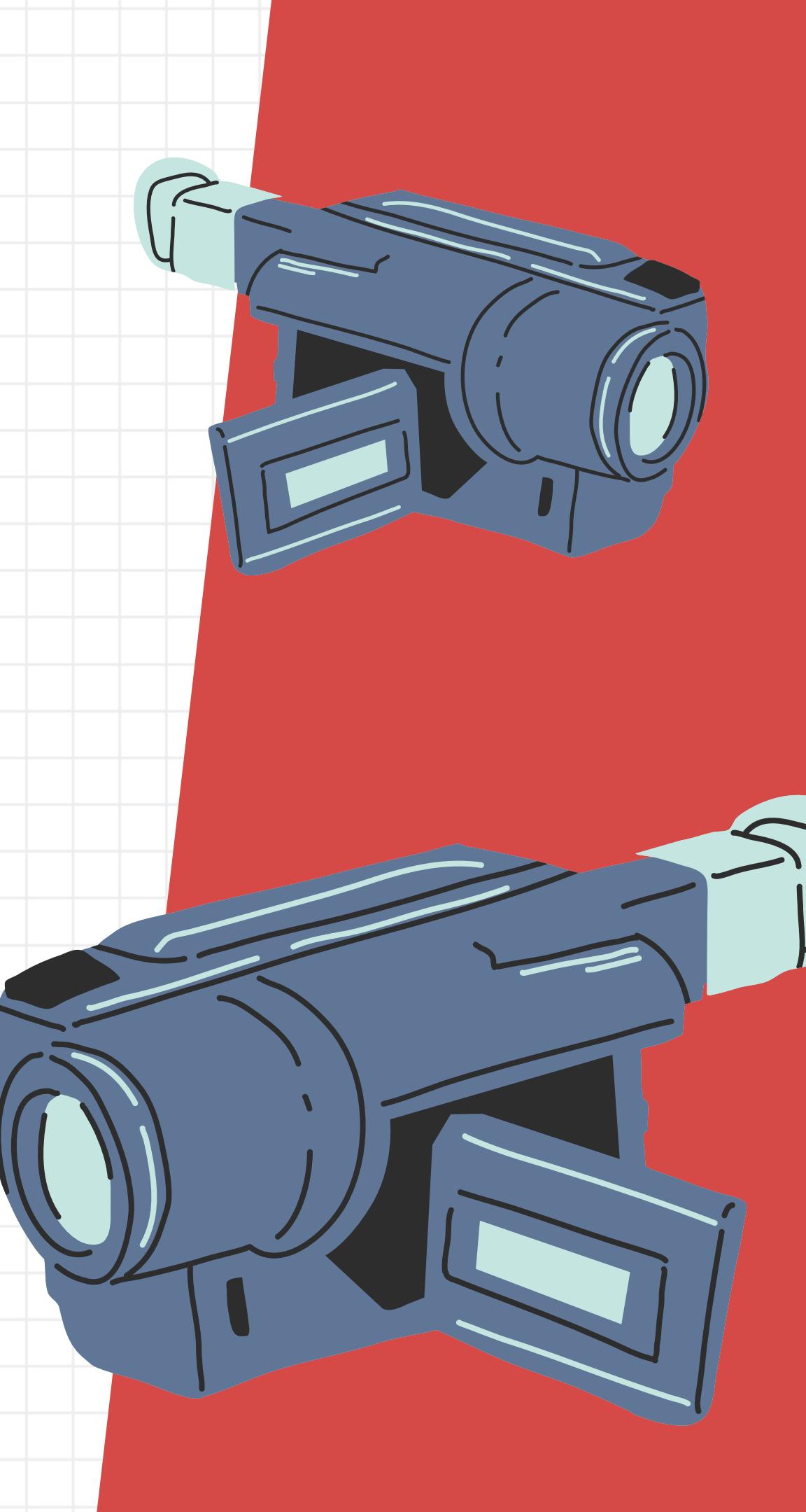
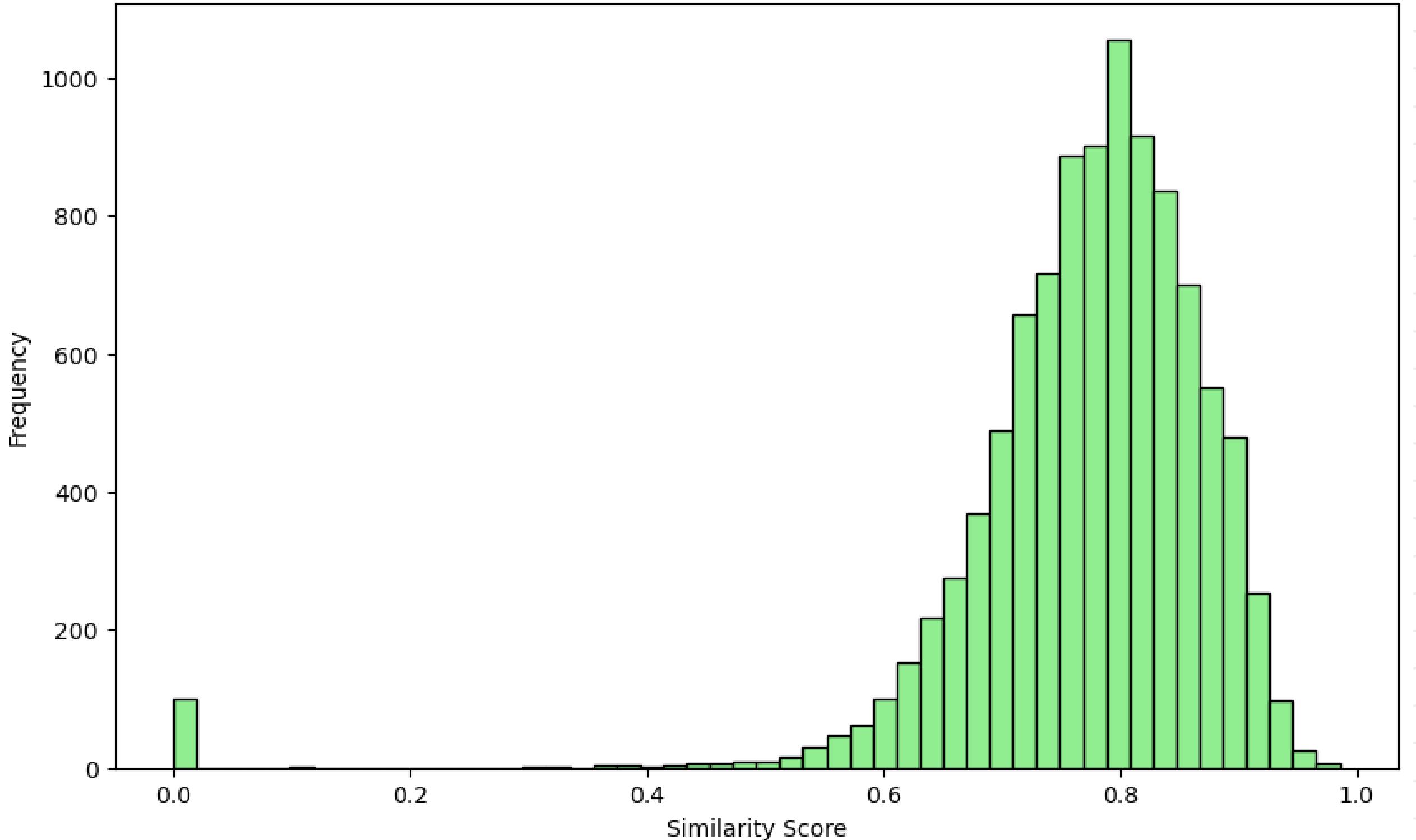
If we put the above sentences in a CountVectorizer, the vocabulary produced would be "they, find, in, the, Emerald, City, finally, reach" and the vectors for each sentence would be as follows:

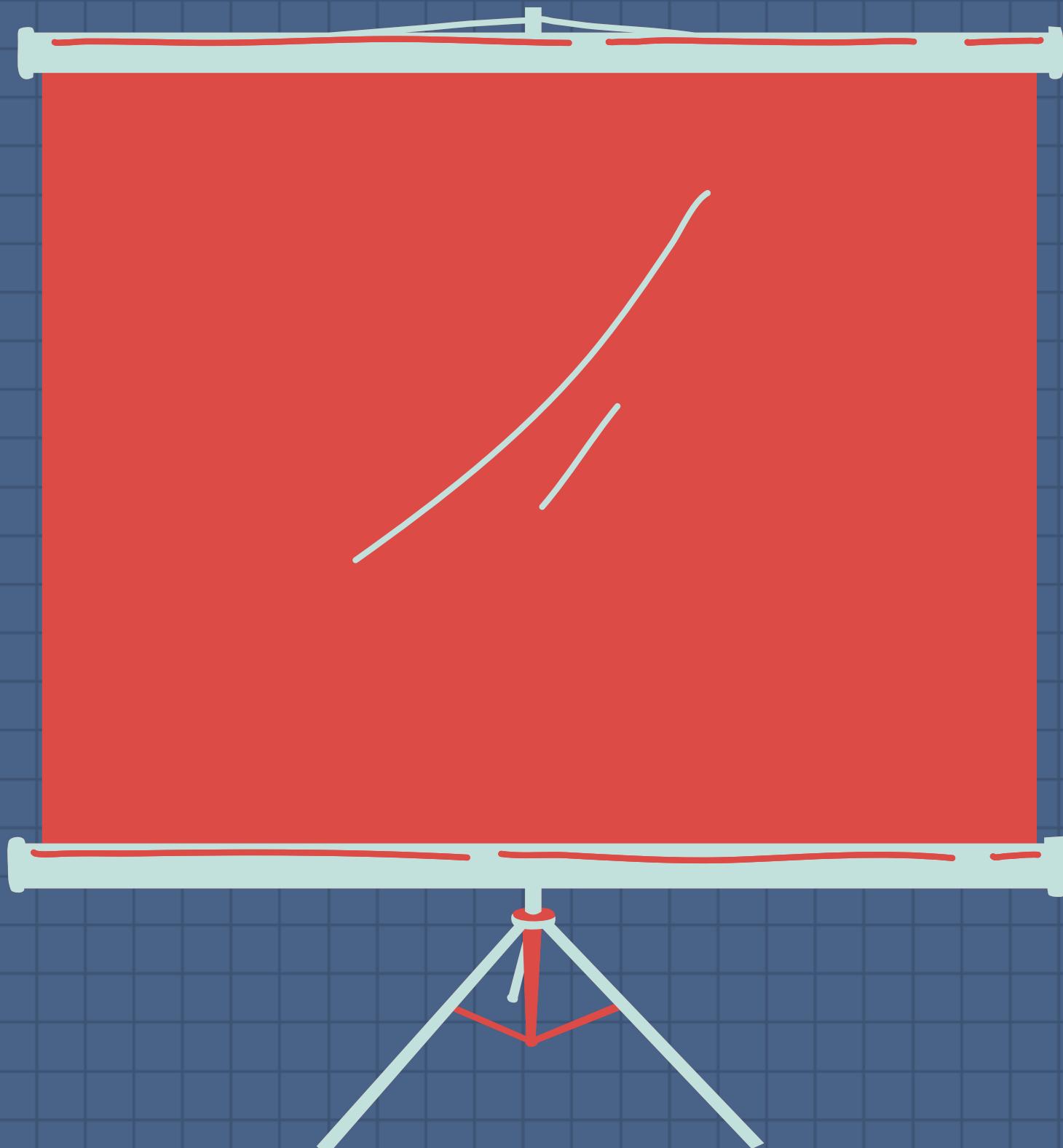
1, 1, 1, 1, 1, 0, 0

1, 0, 0, 1, 1, 1, 1

When we calculate the cosine angle formed between the vectors represented by the above, we get a score of 0.667. This means the above sentences are very closely related. Similarity distance is  $1 - \underline{\text{cosine similarity angle}}$ . This follows from that if the vectors are similar, the cosine of their angle would be 1 and hence, the distance between them would be  $1 - 1 = 0$ .

### Histogram of Similarity Scores





# Visualizations



# WordCloud from each Cluster

## Cluster 0 Word Cloud



## Cluster 2 Word Cloud



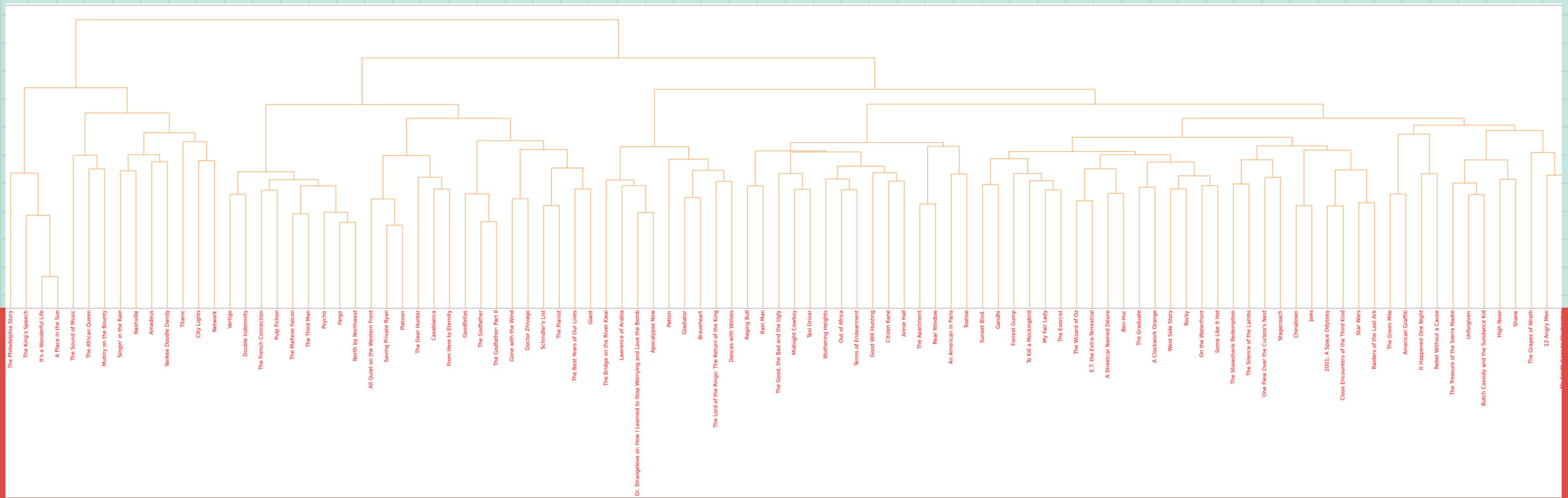
## Cluster 4 Word Cloud

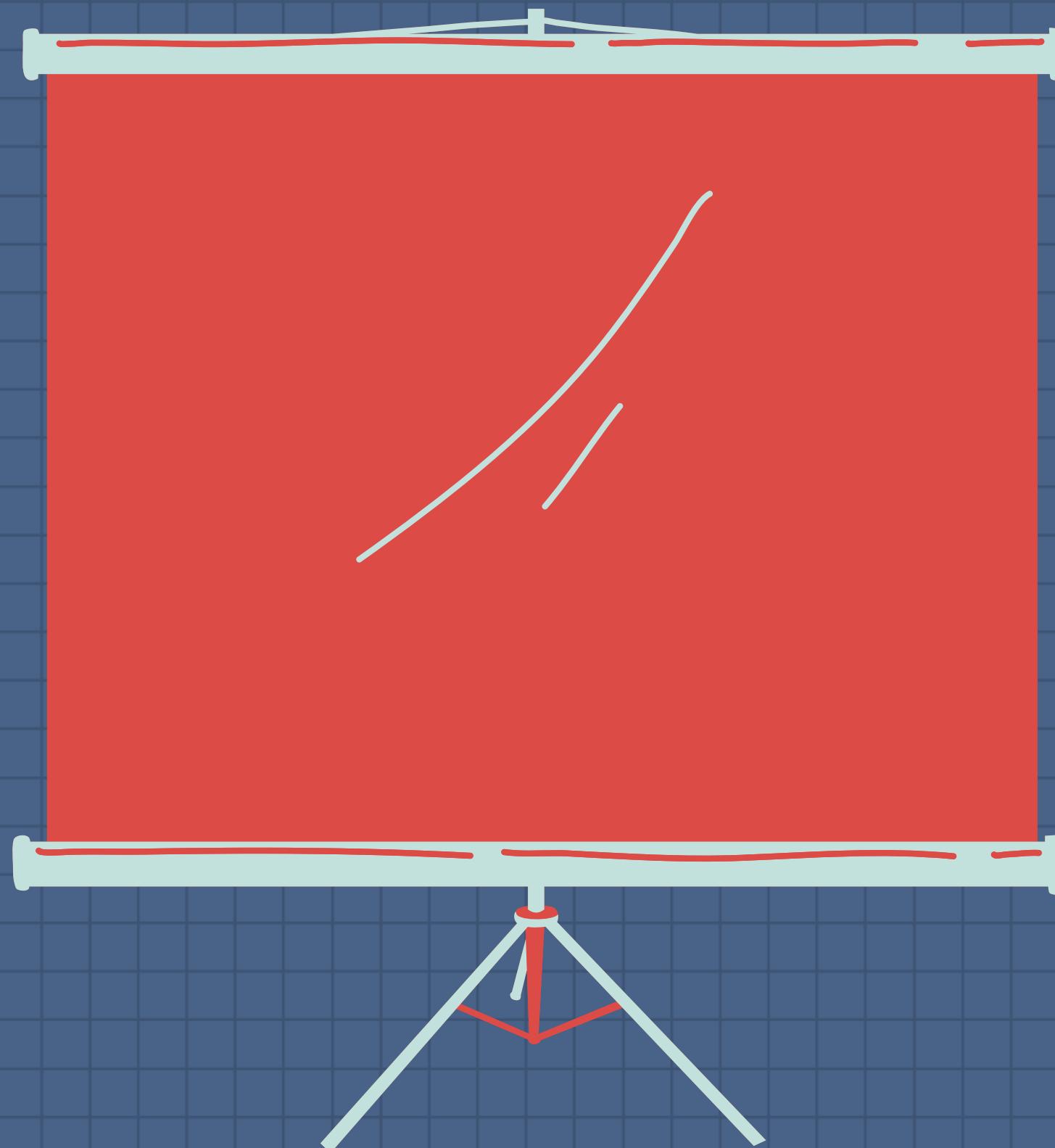


## Cluster 1 Word Cloud



# Dendrogram



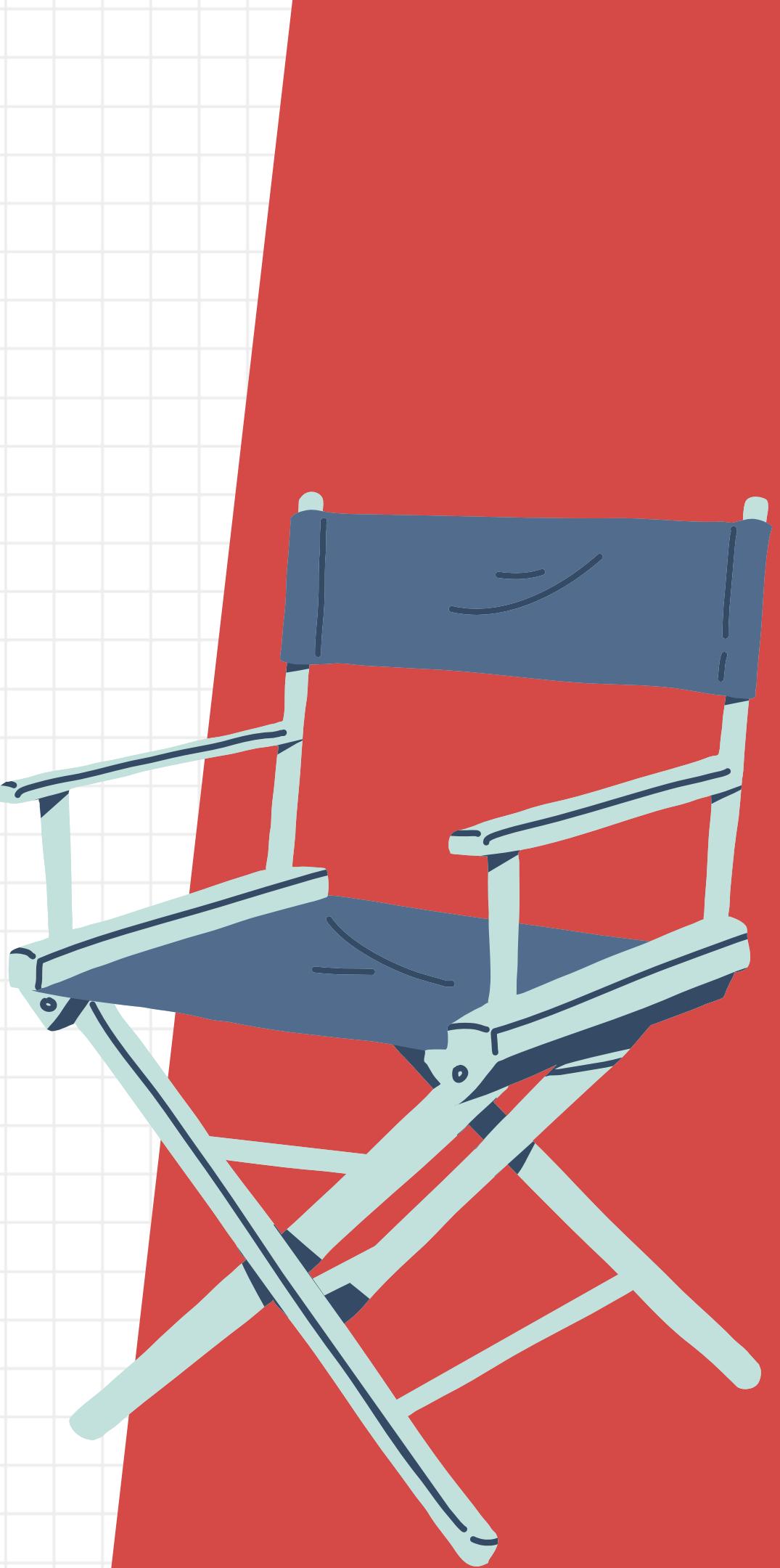


# Some Conclusions



# What is the average similarity score between movies in the same cluster?

- Average similarity score for cluster 0: 0.4732157414926383
- Average similarity score for cluster 1: 0.7159361615388798
- Average similarity score for cluster 2: 0.7353582797346853
- Average similarity score for cluster 3: 0.6578554341809363
- Average similarity score for cluster 4: 0.7246925351957314





# Which movies are most similar?

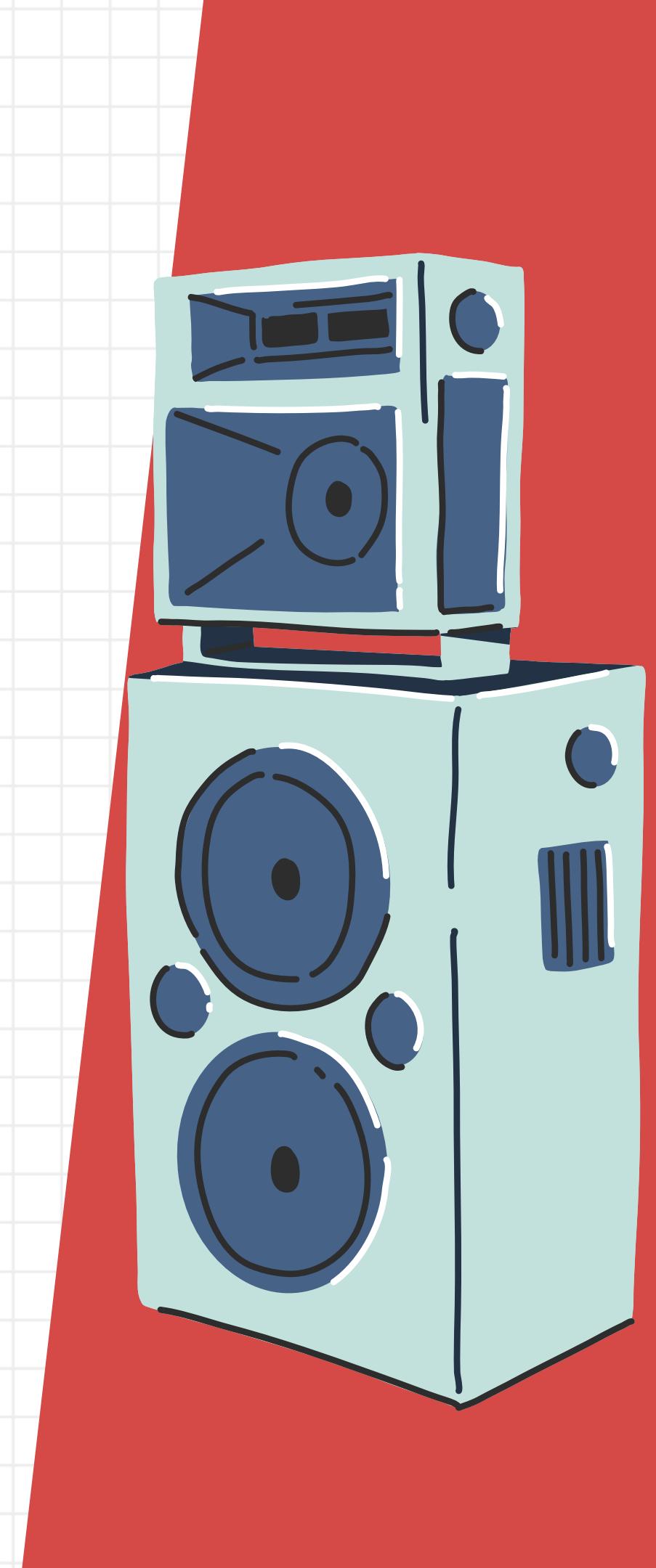


The movie most similar to *Braveheart* is: *Gladiator*

# Which movies belong to the same cluster as "The Godfather"?

The movies in the same cluster as The Godfather are:

- 0                    The Godfather
- 12                  Psycho
- 13                  Sunset Blvd.
- 14                  Vertigo
- 18                  West Side Story
- 20                  E.T. the Extra-Terrestrial
- 21                  2001: A Space Odyssey
- 23                  Chinatown
- 32                  Gandhi
- 37                  Unforgiven
- 42                  To Kill a Mockingbird
- 49                  Jaws.....and so on



# Thank You

