Clustering Spotify Million Song & Recommender Systems

Korkrid Kyle Akepanidtaworn

DTSA 5510: Unsupervised Algorithms in Machine Learning
December 8, 2024

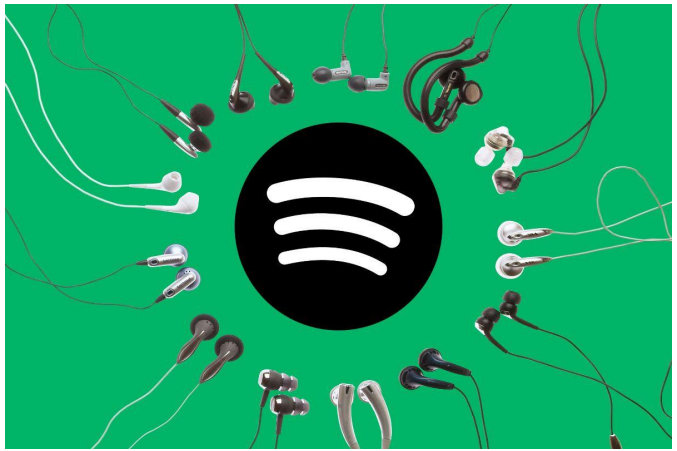
Agenda

1. Project Motivation
2. Exploratory Data Analysis (EDA)
3. Model Building and Training
    a. K-Means Clustering
    b. Hierarchical Clustering
    c. Recommender System (TF-IDF-based Cosine Similarity Approach)
4. Conclusion
5. Demo Code Walkthrough

Project Motivation

Motivation

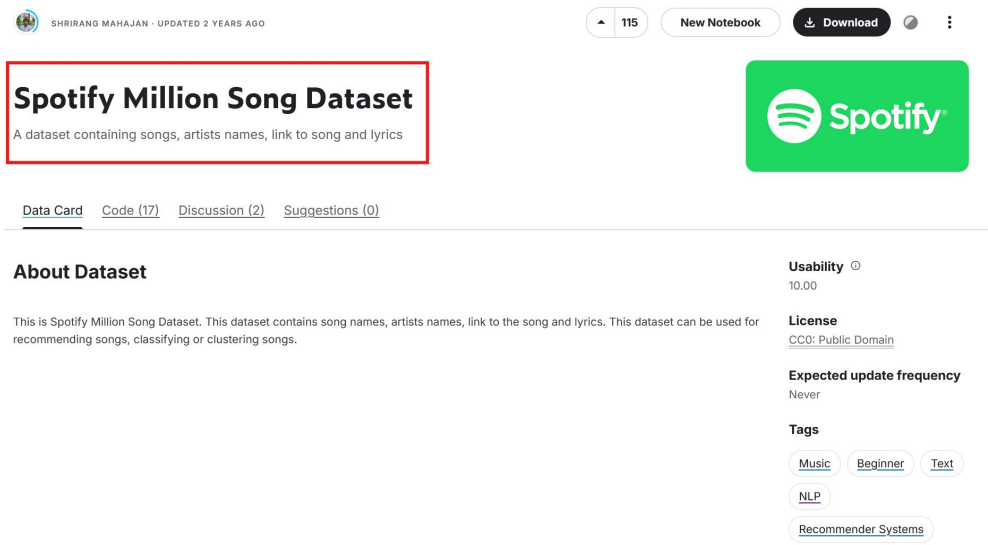
- **Spotify** is a digital music streaming service that provides instant access to its vast online library of music and podcasts. You can listen to any content of your choice anytime. It is both legal and easy to use. You will find millions of songs from various genres and artists, from obscure indie rock and top 40 pop, to movie soundtracks and classical music. It also has a complex algorithm to recommend music based on your listening history, curated playlists and internet radio station.
- This project utilizes **the Spotify Million Song Dataset**, which includes **song titles, artist names, song links, and lyrics**. This dataset is suitable for various applications, such as song recommendation and classification. This project specifically explores different clustering techniques from the scikit-learn library.



Spotify Data at a glance...

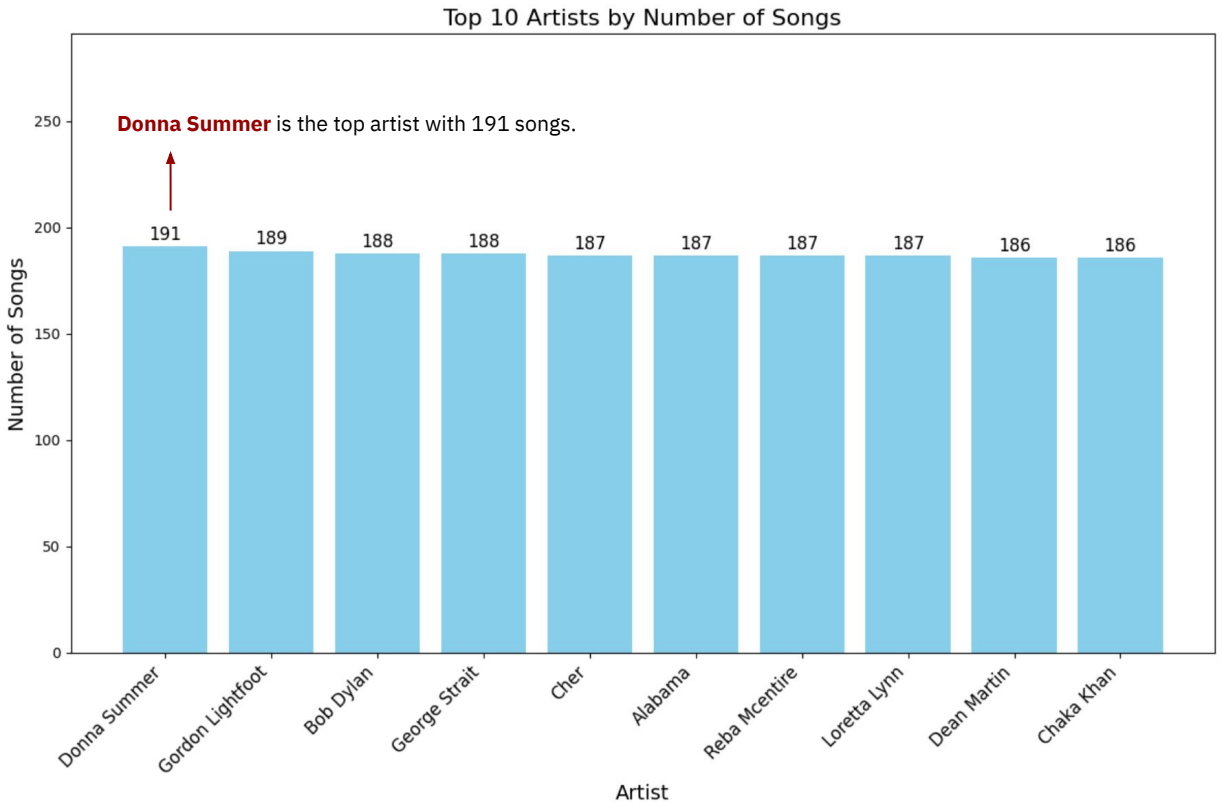
```python
# Load datasets into pandas DataFrames from CSV files
df = pd.read_csv("/root/.cache/kagglehub/datasets/notshrirang/spotify-million-song-dataset/versions/1/spotify_millsongdata.csv")

# Inspect data
print(df.head())

# Print overview of data
print(df.info())

# Perform basic data inspection such as getting the number of observations and number of features
print(f"The dataset has {df.shape[0]} rows and {df.shape[1]} columns.")
```

```
  artist                song                                        link  \
0   ABBA  Ahe's My Kind Of Girl  /a/abba/ahes+my+kind+of+girl_20598417.html
1   ABBA       Andante, Andante       /a/abba/andante+andante_20002708.html
2   ABBA         As Good As New        /a/abba/as+good+as+new_20003033.html
3   ABBA                   Bang                  /a/abba/bang_20598415.html
4   ABBA       Bang-A-Boomerang      /a/abba/bang+a+boomerang_20002668.html

                                                text
0  Look at her face, it's a wonderful face  \r\nA...
1  Take it easy with me, please  \r\nTouch me gen...
2  I'll never know why I had to go  \r\nWhy I had...
3  Making somebody happy is a question of give an...
4  Making somebody happy is a question of give an...
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 57650 entries, 0 to 57649
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   artist  57650 non-null  object
 1   song    57650 non-null  object
 2   link    57650 non-null  object
 3   text    57650 non-null  object
dtypes: object(4)
memory usage: 1.8+ MB
None
The dataset has 57650 rows and 4 columns
```



Exploratory Data Analysis (EDA)

EDA: Top 10 Artists by Number of Songs

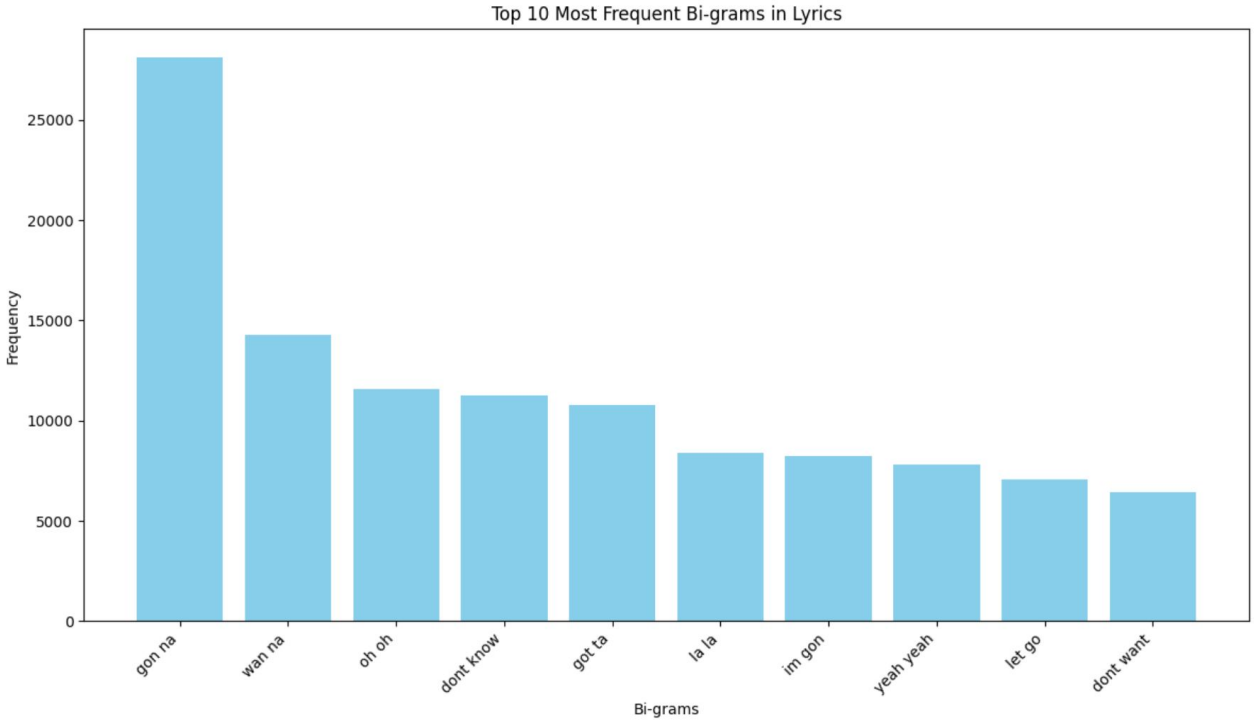


Observations

Close Competition: The next few artists have very similar numbers of songs, with Gordon Lightfoot (189), Bob Dylan (188), and George Strait (188) being close behind.

Even Distribution: The top 10 artists have a relatively even distribution of songs, with the difference between the top and bottom artists being only 15 songs.

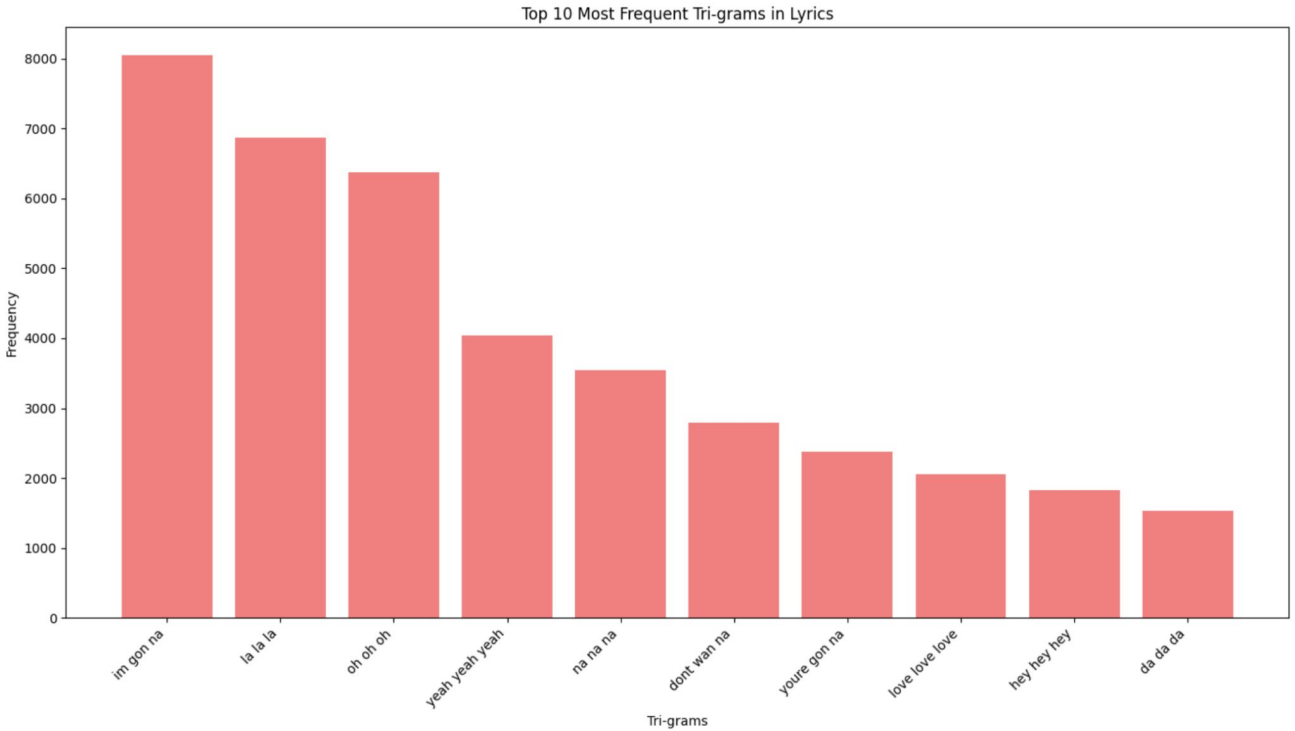
EDA: Bi-gram Analysis



Observations

The plot identifies the most frequent bi-grams (two-word phrases) in the song lyrics. **Gon na** is the most common, followed by **wan na, oh oh,** and others. The presence of contractions like **gon na and wan na** suggests their frequent use in song lyrics, likely for rhythmic and poetic purposes. Bi-grams like **oh oh** and **yeah yeah** often express emotions such as surprise, excitement, or agreement, indicating their role in conveying feelings through music. These are common songwriting techniques used to create catchy and memorable phrases.

EDA: Tri-gram Analysis



Observations

The plot reveals the most commonly used tri-grams in song lyrics. **im gon na** takes the lead, followed by **la la la, oh oh oh**, and so on. It highlights the common phrases that resonate with listeners and contribute to the overall musical experience.

Model Building and Training

K-Means Clustering

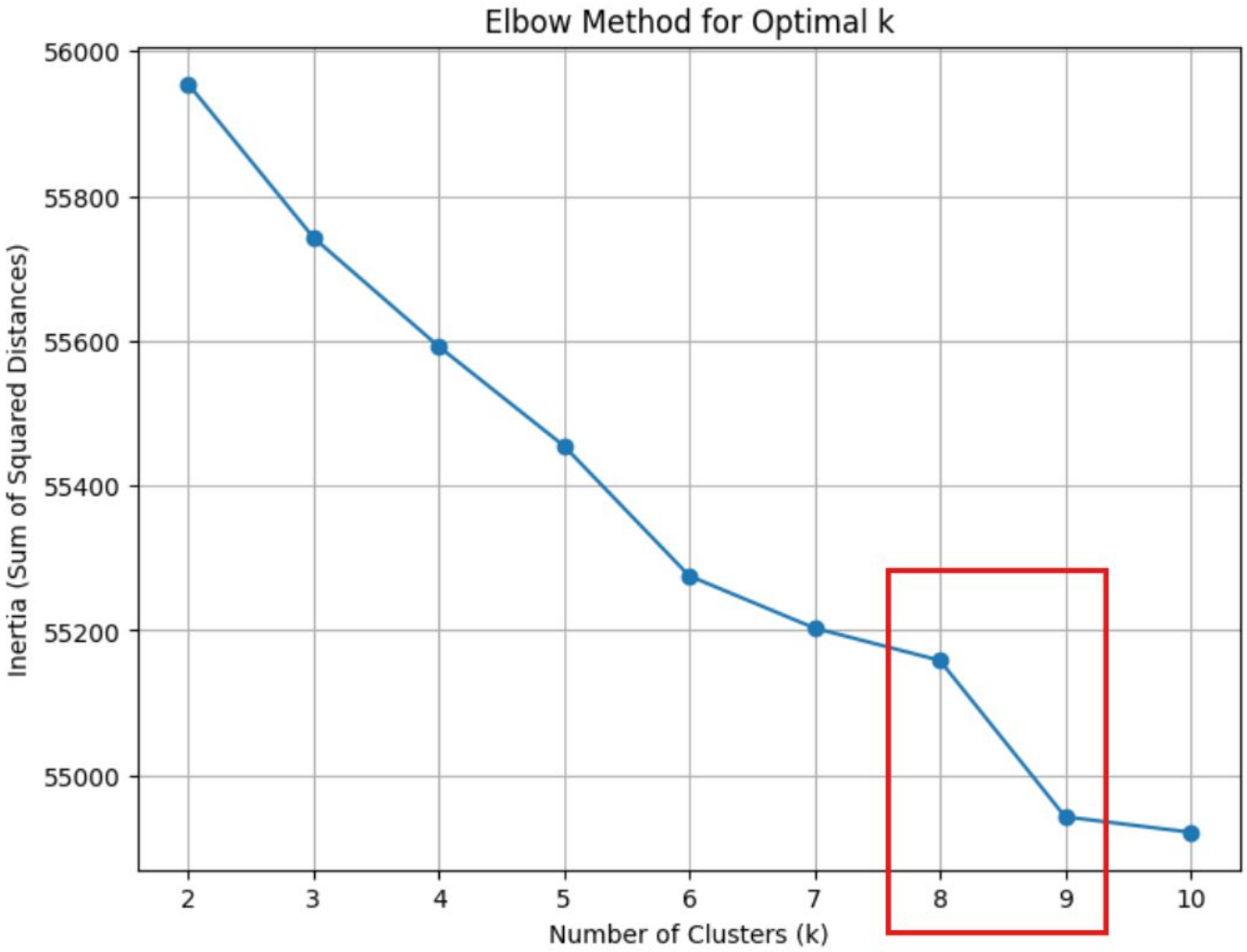
- **The K-means clustering algorithm c**omputes centroids and repeats until the optimal centroid is found. It is presumptively known how many clusters there are. It is also known as the flat clustering algorithm. The optimal number of clusters found from data by the method is denoted by the letter K in K-means. In this method, data points are assigned to initial clusters in such a way that the sum of the squared distances between the data points and the centroid is as small as possible. It is essential to note that reduced diversity within clusters leads to more identical data points within the same cluster.
- The **"Elbow Method"** is used in K-means clustering to identify the optimal number of clusters (k) by plotting the within-cluster sum of squares (WCSS) against different values of k, and selecting the point where the curve sharply changes, resembling an "elbow," indicating that adding more clusters doesn't significantly reduce the overall distance within clusters, thus representing the best balance between capturing patterns and avoiding overfitting; essentially, it helps determine the point where adding more clusters provides diminishing returns in terms of improved clustering quality
- **The silhouette score** is a metric used to evaluate the quality of clusters created by K-means clustering.

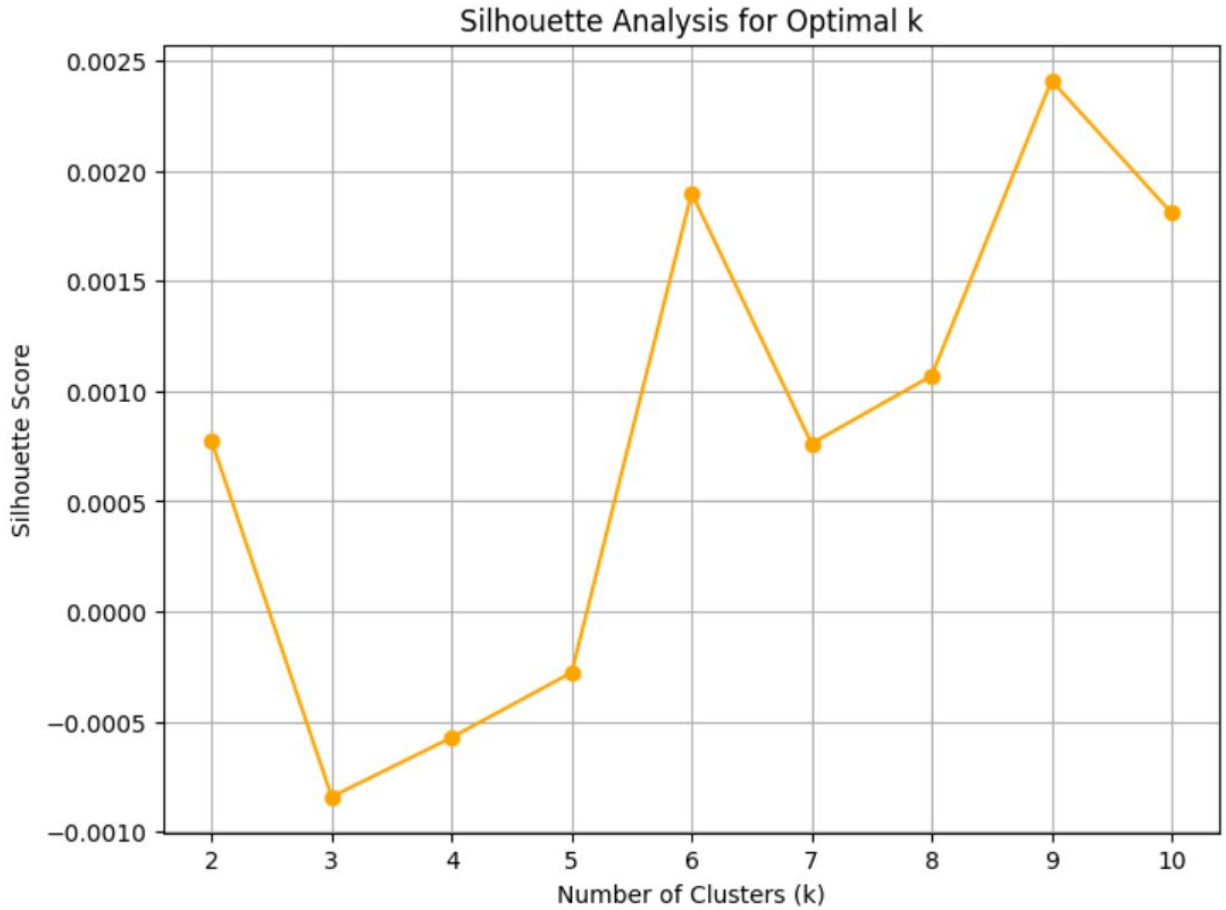
$$S = \frac{(b - a)}{max(a, b)}$$

What the score means

The silhouette score ranges from -1 to 1:

- **1**: Clusters are well-separated and clearly distinguished
- **0**: Clusters are indifferent, or the distance between clusters is not significant
- **-1**: Clusters are assigned in the wrong way


Elbow Method for Optimal k
Inertia (Sum of Squared Distances)
56000
55800
55600
55400
55200
55000
2
3
4
5
6
7
8
9
10
Number of Clusters (k)




```
[INFO] Optimal number of clusters (k) determined: 9
[INFO] Applying K-means clustering with k=9...
[INFO] Clustering completed in 83.74 seconds.
```

Cluster Examination

```
[INFO] Cluster 1 example songs:
      artist                          song  \
107     ABBA  We Wish You A Merry Christmas
364  Alabama             Christmas In Dixie
365  Alabama         Christmas In Your Arms
366  Alabama              Christmas Is Love
367  Alabama                Christmas Shoes

                                                  text
107  We wish you a merry Christmas  \r\nWe wish you...
364  By now in New York City, there's snow on the g...
365  All my friends are asking me where I plan to s...
366  It's that time of year when the whole world is...
367  It was almost Christmas time  \r\nThere I stoo...
```

Cluster 1 primarily consists of **Christmas songs**.

```
[INFO] Cluster 3 example songs:
       artist               song  \
78        ABBA  Rock 'n Roll Band
138      Adele         I Miss You
145  Aerosmith      All Your Love
173  Aerosmith    No More No More
189  Aerosmith          Red House

                                                 text
78   Sitting in the darkest corner  \r\nIn the tend...
138  [Verse 1]  \r\nI want every single piece of yo...
145  All your love I miss lovin'  \r\nAll your kiss...
173  Blood stains the ivorys  \r\nOf my Daddy's bab...
189  There's a Red House over yonder  \r\nThat's wh...
```

Cluster 3 contains predominantly **rock 'n roll songs.**

Cluster Examination

```
[INFO] Cluster 6 example songs:
        artist                song  \
9         ABBA    Crying Over You
101       ABBA  Tropical Loveland
178  Aerosmith     Once Is Enough
190  Aerosmith           Remember
196  Aerosmith     Scream In Pain

                                                  text
9    I'm waitin' for you baby  \r\nI'm sitting all ...
101  Come to my loveland, wander along  \r\nBeautif...
178  Been lucked out  \r\nBroke a woman's heart  \r...
190  Seems like the other day,  \r\nMy baby went aw...
196  It's deep insignus  \r\nThat knows me well  \r...
```

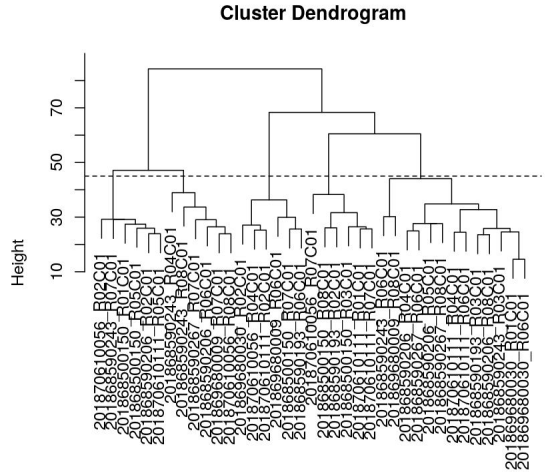
Cluster 6, on the other hand, is composed of **sad, melancholic songs.**

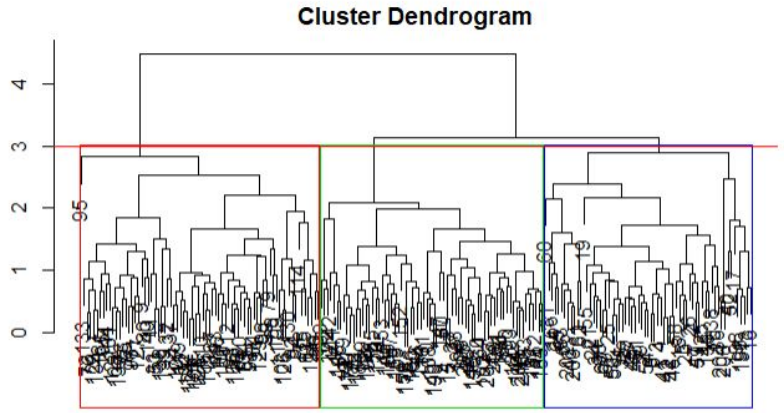
```
[INFO] Optimal number of clusters (k) determined: 9
[INFO] Applying K-means clustering with k=9...
[INFO] Clustering completed in 83.74 seconds.
[INFO] Cluster sizes:
cluster
2    26817
5    11516
0     7373
7     3730
4     3475
3     2121
6     1717
1      557
8      344
Name: count, dtype: int64
```

Based on the results, the optimal number of clusters is determined to be **9**, categorized by song content. While the distribution may not be perfectly even, the clusters are programmatically meaningful.

Hierarchical Clustering

- While hierarchical clustering was considered, its computational complexity and potential performance issues on high-dimensional datasets make it unsuitable for this project. Like K-Means clustering, hierarchical clustering is an unsupervised learning technique that does not require a target variable. However, it can be computationally expensive, especially for large datasets. Due to the large number of data points, visualizing the dendrogram is impractical.





Recommender System

- Spotify is a well-known use case for recommendation engines. To recommend songs based on lyrical similarity, I employ a **TF-IDF-based cosine similarity approach**.
- **Cosine similarity** checks how alike two vectors are in an inner product space by looking at the cosine of the angle between them. It tells us if the vectors are heading in the same direction. This method is often used in text analysis, like comparing song lyrics.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

```python
# Function to provide top relevant 5 song recommendations.
def recommend_songs(df, song_title, top_n=5):
    """
    Author: @Korkrid Kyle Akepanidtaworn, MSDS
    Recommends songs based on cosine similarity of TF-IDF vectors.

    Args:
        df: DataFrame containing 'content' (combined text features) and 'song' columns.
        song_title: Title of the song for which recommendations are needed.
        top_n: Number of top recommendations to return.

    Returns:
        DataFrame: Top N recommended songs (based on similarity).
    """

    # Ensure the song title exists in the DataFrame
    if song_title not in df['song'].values:
        print(f"Song '{song_title}' not found in the dataset.")
        return pd.DataFrame()  # Return empty DataFrame if song not found

    # Step 1: Initialize the TF-IDF Vectorizer (removing common English stop words)
    vectorizer = TfidfVectorizer(stop_words='english')  # Convert text data into TF-IDF vectors
    X = vectorizer.fit_transform(df['content'])  # Apply the vectorizer to the 'content' column

    # Step 2: Find the index of the input song in the DataFrame
    song_index = df[df['song'] == song_title].index[0]

    # Step 3: Calculate the cosine similarity between the input song's vector and all other song vectors
    similarity_scores = cosine_similarity(X[song_index], X)

    # Step 4: Get the indices of the most similar songs (excluding the input song itself)
    similar_song_indices = similarity_scores.argsort()[0][-top_n-1:-1][::-1]

    # Step 5: Create a DataFrame of the recommended songs based on the similar indices
    recommendations = df.iloc[similar_song_indices][['artist', 'song']]
    recommendations.reset_index(inplace=True, drop=True)  # Reset the index for neatness

    # Return the DataFrame containing the recommendations
    return recommendations
```

Recommender Examination

```python
# End time logging and print execution duration
end_time = time.time()
execution_time = end_time - start_time
print(f"Execution completed in {execution_time:.4f} seconds.")
```

```
Recommended songs similar to 'Friend':
              artist                  song
0          Iggy Pop              Sickness
1      Foo Fighters           On The Mend
2           'n Sync   Tearing Up My Heart
3  Steve Miller Band            My Friend
4           Rainbow  Tearin' Out My Heart
Execution completed in 13.6997 seconds.
```

1. Iggy Pop - Sickness
2. Foo Fighters - On The Mend
3. 'N Sync - Tearing Up My Heart
4. Steve Miller Band - My Friend
5. Rainbow - Tearin' Out My Heart

```
Recommended songs similar to 'I'm Money':
               artist                       song
0         Roy Orbison                      Money
1  Grand Funk Railroad   All You've Got Is Money
2      Britney Spears               Intimidated
3               Doors                      Money
4           R. Kelly                      Hotel
5              Ne-Yo          Cause I said so
6               Rush            The Big Money
7           Fabolous  Money Money Money Shouts
8       Avril Lavigne  I Always Get What I Want
9             J Cole                 Mo Money
Execution completed in 13.7128 seconds.
```

1. Roy Orbison - Money
2. Grand Funk Railroad - All You've Got Is Money
3. Britney Spears - Intimidated
4. The Doors - Money
5. R. Kelly - Hotel
6. Ne-Yo - Cause I said so
7. Rush - The Big Money
8. Fabolous - Money Money Money Shouts
9. Avril Lavigne - I Always Get What I Want
10. J Cole - Mo Money

Conclusion

Conclusion

1. **Top Artist**: Donna Summer is the most represented artist with 191 songs.
2. **Common Phrases**: The most frequent tri-grams in song lyrics are "im gon na," "la la la," "oh oh oh," and others. These phrases likely contribute to the overall musical experience and listener appeal.
3. **Optimal Clustering:** The optimal number of clusters, determined through the elbow method and silhouette analysis, is 9. These clusters are categorized by song content, with examples including Christmas songs, rock 'n roll, and melancholic songs. Hierarchical clustering was not suitable due to the large dataset and computational complexity.
4. **Song Recommendation:** A TF-IDF-based cosine similarity approach was used to develop a song recommendation function. It's important to note that these recommendations are algorithmic and may not perfectly align with individual preferences. Musical taste is subjective, and what one person finds similar, another may not.

Demo Code Walkthrough

Thank you!