# SONG GENERATION

By: Nada Bakeer
Nour Shehab

# CONTENT

1. MS1 (Data preprocessing and analysis) edits

2. MS2 (Defining a model)

3. MS3 (Using a pre-trained model)

# MS1
# (Data preprocessing and analysis) edits

**Objective:**

Prepare, clean, and analyze song lyrics dataset

# DATA CLEANING - INITIAL STEPS

- Initial Exploratory Data Analysis
    - Check for null values
    - Examine data types
    - Summarize dataset
- Rename the 'text' column to 'lyrics'
- Remove missing values and duplicates (none found)
- Drop unnecessary columns: 'link' and 'song'

# DATA CLEANING - TEXT PREPROCESSING

- Remove non-alphabetic characters
- Remove repetitive terms: 'chorus', 'verse', 'intro', 'original', 'outro'
- Convert lyrics to lowercase
- Regular expressions for unwanted characters
- Apply the same preprocessing to artist names

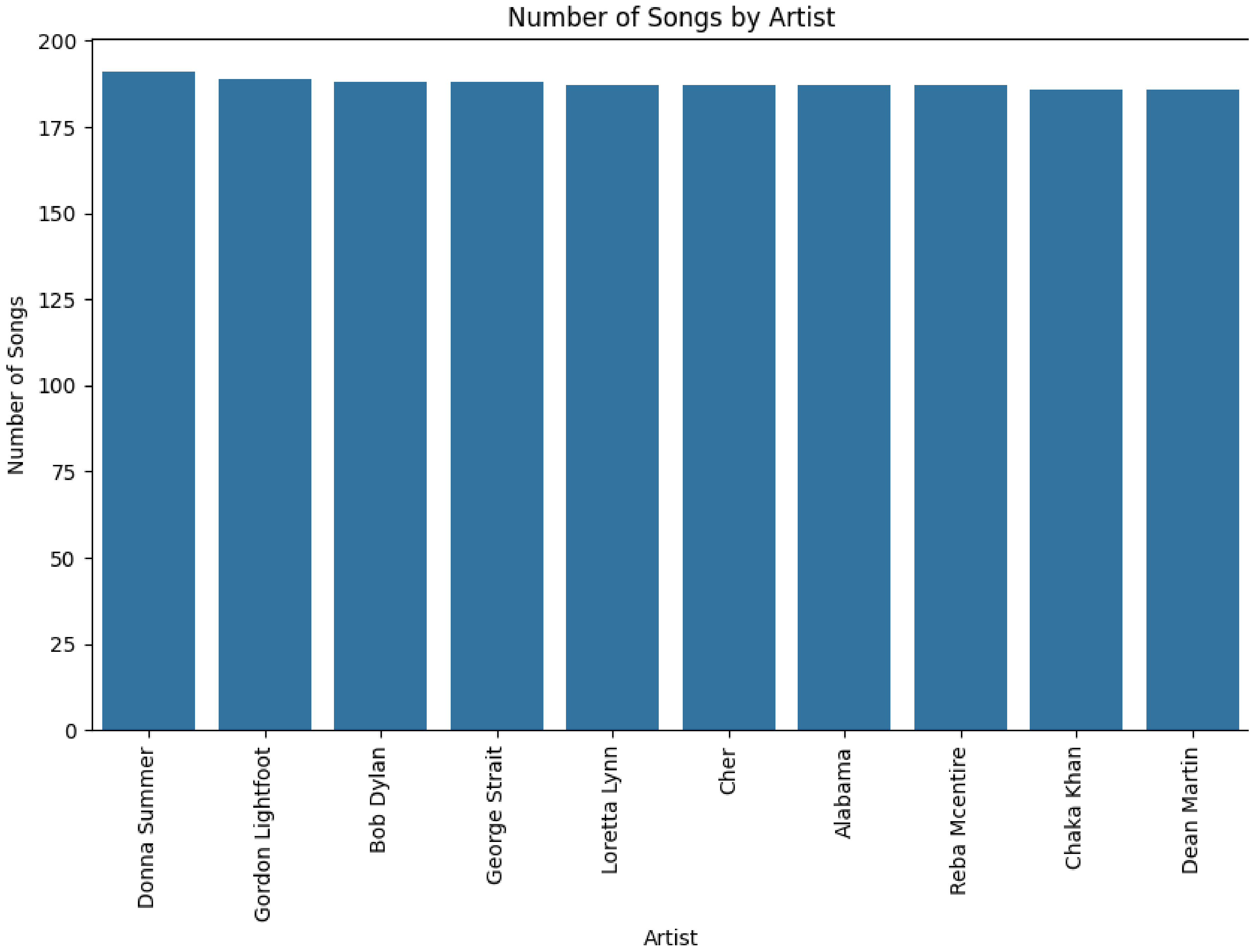
# DATA CLEANING - FINAL STEPS

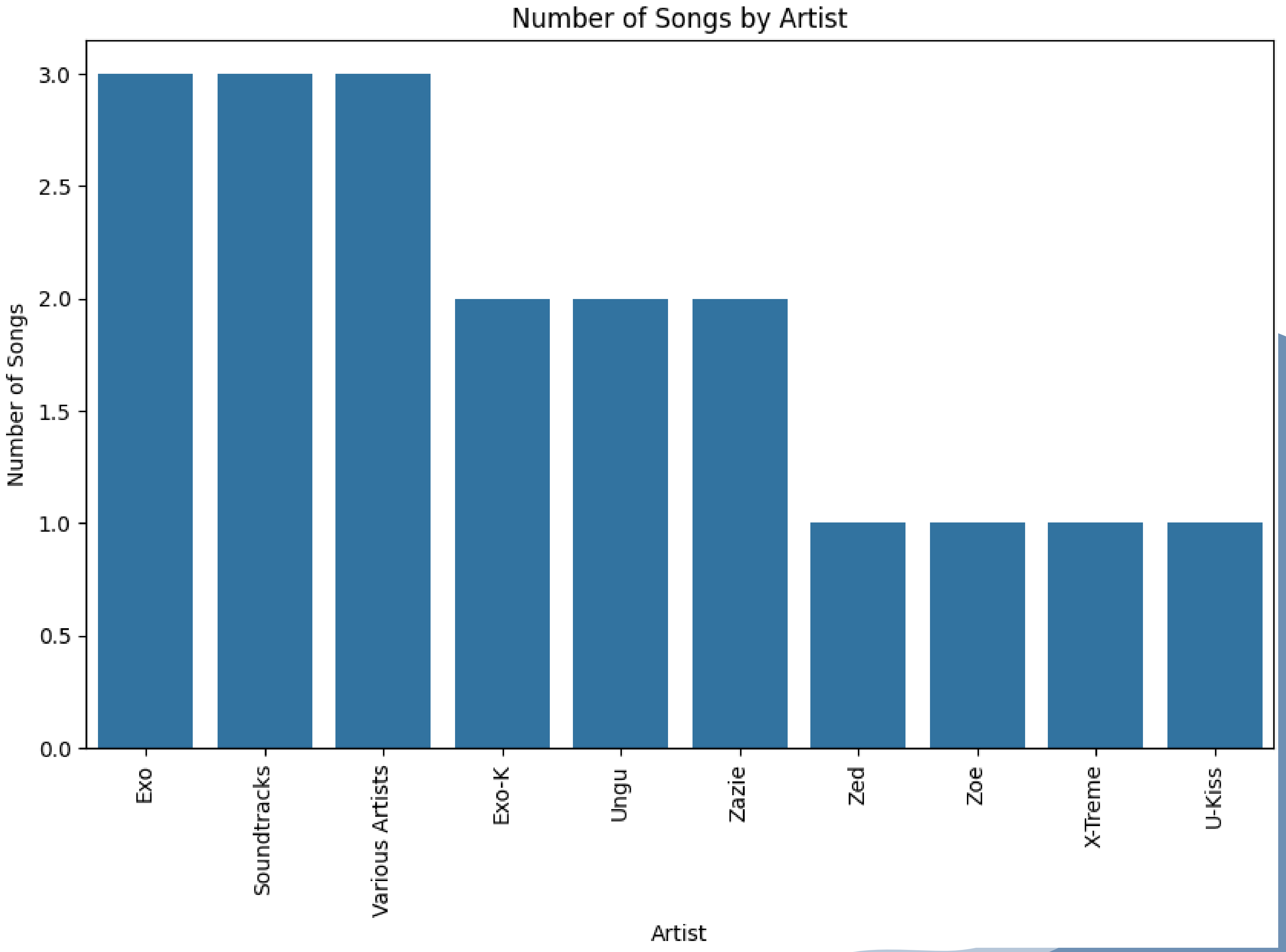
- The decision against stop words removal and lemmatization
- Preserve sentence structure and rhyme
- Ignore spelling check to maintain the original lyric style

# DATA ANALYSIS - SONG DISTRIBUTION

- Visualize the distribution of songs per artist
- Bar plots: top 10 and bottom 10 artists
- Insights: dataset composition
- Top Artist: Donna Summer with 191 songs

# DATA ANALYSIS - SONG DISTRIBUTION



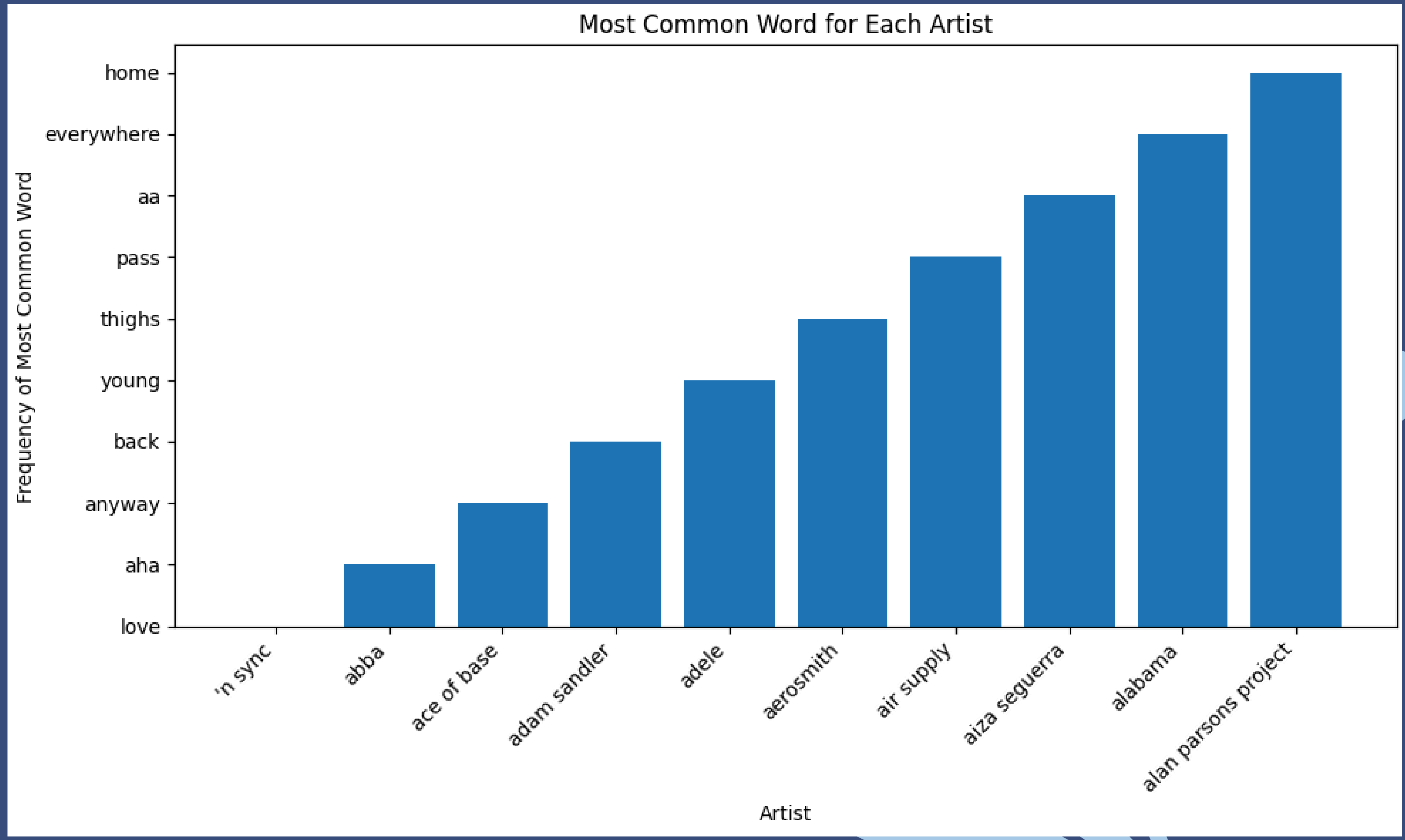



# DATA ANALYSIS - FREQUENT WORDS

- Identify the most frequent word per artist
- Create a table of artists and their most frequent words
- Importance: Analyze model output later

# DATA ANALYSIS - TF-IDF

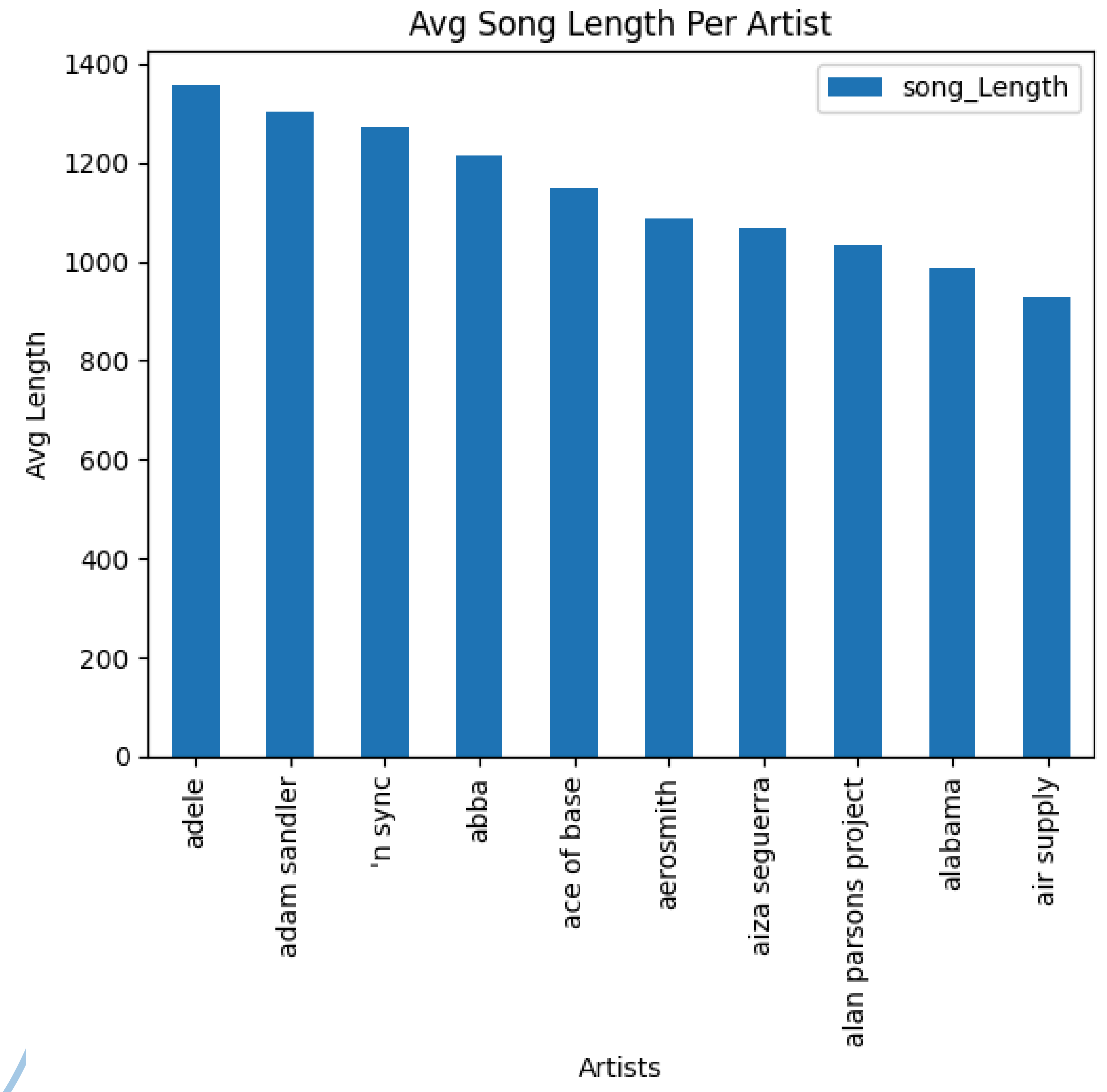
- Calculate TF-IDF for each artist's frequent word
- Purpose: Keyword extraction, information retrieval
- Remove stop words and determine frequent words
- Visualize frequencies using a bar plot



# DATA ANALYSIS - SONG LENGTH

- Analyze average song length per artist
- Visualize the average song length for the top 10 artists
- Create histograms for song length distribution
- Insight: Few songs longer than 1500 words, hence truncated

# DATA ANALYSIS - SONG LENGTH



# MS2
## (Defining a model)

Objectives:

Creating neural network

# DATA PREPARATION

- Joined both artist name and song lyrics
- Removed all songs with length >1500
- Eliminated the last word of each song and stored it in a new column
- Converted the lyrics to Ragged Tensor
  - Nested- variable
  - Adapts to varying song length
- One-hot encoded the lyrics

# MODEL SETUP

- We split the data into a training set and a testing set
- Initialized tokenizer
- Converted the lyrics of both sets to sequences
    - Padded the lyrics to a maximum length of 1500
- Defining a sequential model with:
    - Embedding layer
    - GRU layer
    - Dense layer
    - Sigmoid activation function

# MODEL EVALUATION

- The model was trained for 10 epochs
- Evaluated model based on loss value
- Defined a function to retrieve the word of the highest probability
- The function was used to compare actual eliminated word with predicted output
- The model resulted in a Loss of 6.97

# MS3
# (Using a pre-trained model)

## Objective:

Train a BERT2BERT model to generate the second half of song lyrics given the first half

# DATA PREPARATION

- Data Splitting: Each song's lyrics were split into two halves
- Input Text: Combined artist's name with the first half of the lyrics using BERT2BERT separation token and start/end tokens
- Output Text: Added start and end tokens to the second half of the lyrics
- Data Cleaning: Removed songs longer than 1500 characters
- Training and Test Sets: Data was split into training and test sets

# MODEL SETUP

- Model Components: Loaded BERT encoder and decoder for BERT2BERT model
- Initialized tokenizer: Chose BERT Tokenizer for seamless pipeline integration
- Version Selection:
  - bert_large_uncased not supported due to RAM constraints
  - bert_tiny_uncased did not yield optimal results
  - bert_base_uncased

# TRAINING PROCESS

- Data Processing:
    - Tokenized training data
    - Converted data to PyTorch tensors
    - Created DataLoader for batching (batch size = 8)

- Training Configuration:
    - Defined optimizer and loss function
    - Trained the model for 10 epochs (approx. 7 hours)
    - Printed loss for each batch

# MODEL EVALUATION

- Test Data Processing:
  - Tokenized test set
  - Converted to PyTorch tensors
  - Created DataLoader

- Evaluation: Calculated mean loss on the test dataset

- Result: Mean loss was approximately 0.02

# Conclusion

# CONCLUSION

- BERT2BERT exhibited a significantly lower loss value compared to our GRU-based model.
- Bidirectional Processing: BERT is bidirectional, considering both left and right context when encoding a token, unlike GRUs
- Context Over Long Sequences: BERT can understand context over long sequences providing a more complex understanding.
- Pre-trained on Large Datasets: BERT's pre-training on large datasets makes it reliable for tasks like lyrics generation

# Thank You