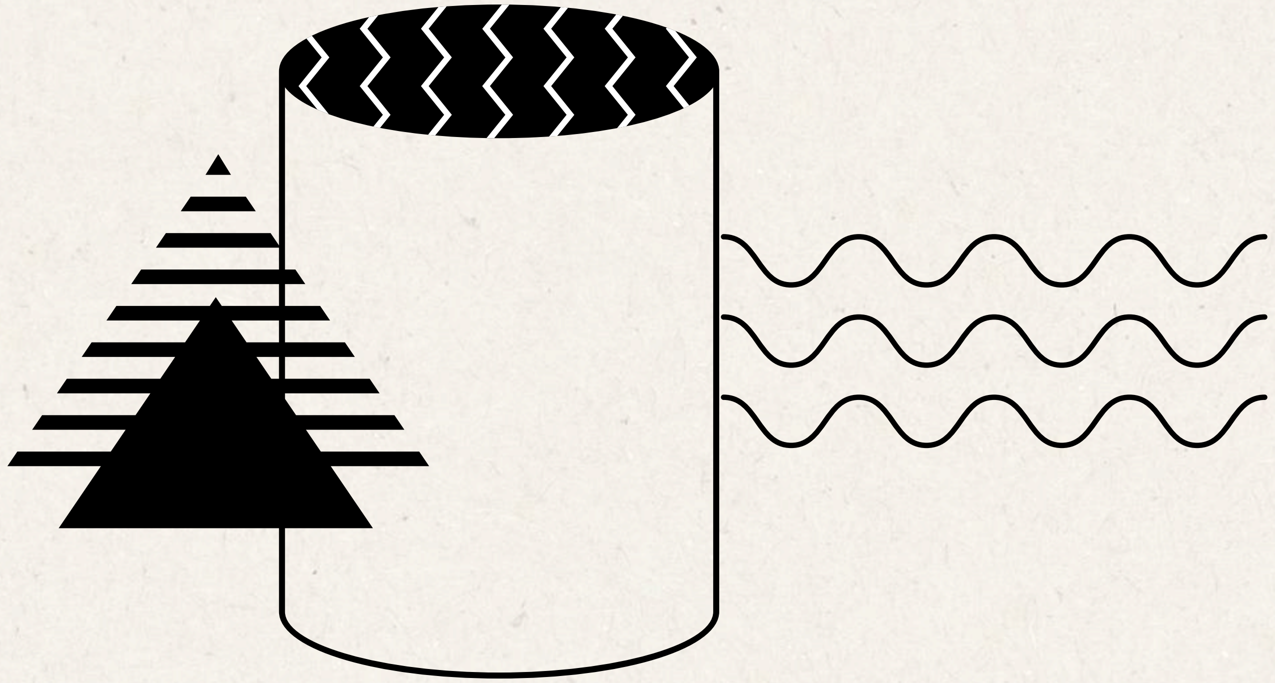
SENTIMENT ANALYSIS OF SONG LYRICS ACROSS DECADES

Beyda Bucak

İçerik

01	Overview
02	Timeline
03	Technologies
04	Code Examples and Workflow
05	Analysis Results

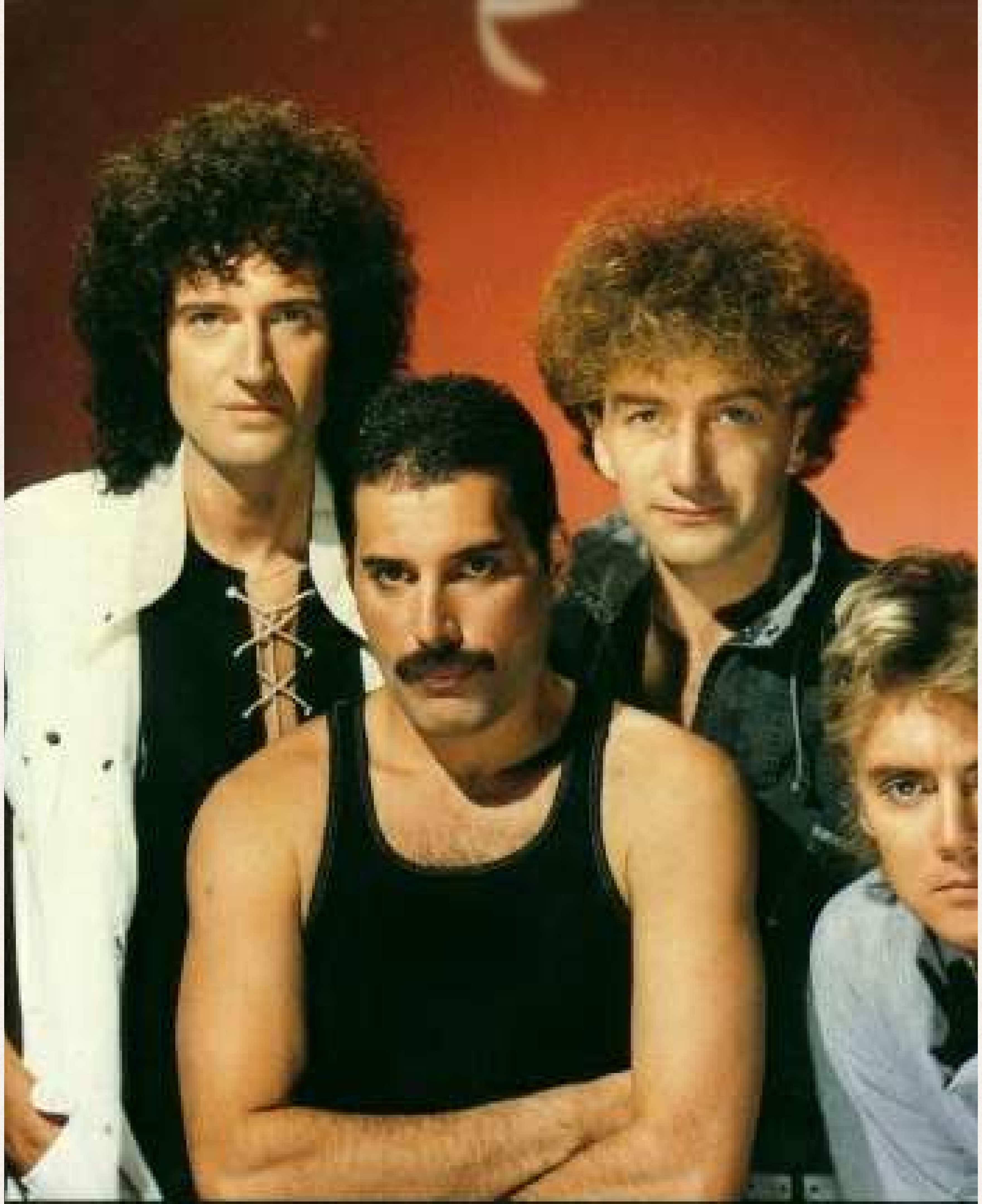


Overview

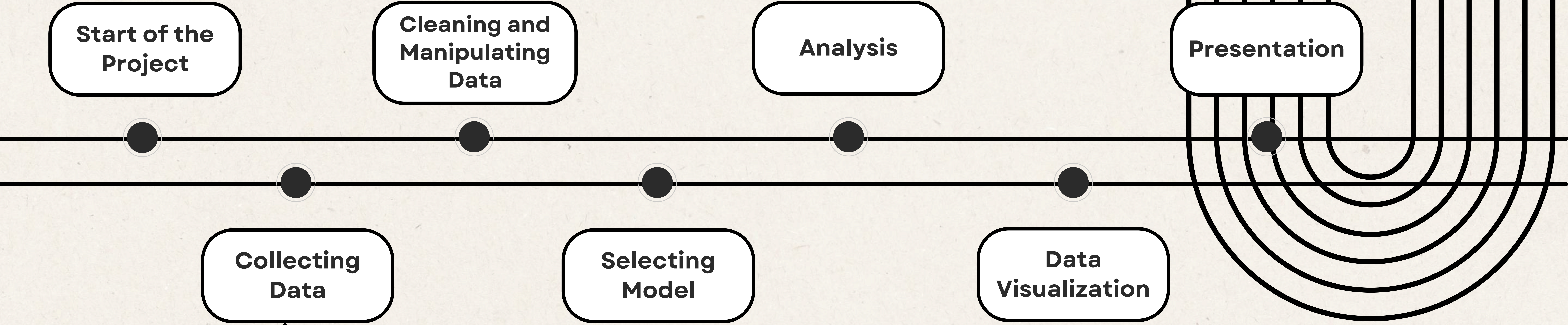
The project involves sentiment analysis of lyrics from songs released in specific periods.

01 The aim of the project is to clearly demonstrate how changing lyrics over different periods impact emotions.

02 It also aims to visualize how lyrics change over time and present the most frequently used words.



Project Phases



Start of the Project

Collecting Data

Cleaning and Manipulating Data

Selecting Model

Analysis

Data Visualization

Presentation

Technologies and Libraries

Programming Language:

Python

Libraries:

Data and File Management

pandas

pickle

os

Natural Language Processing

transformers

nltk

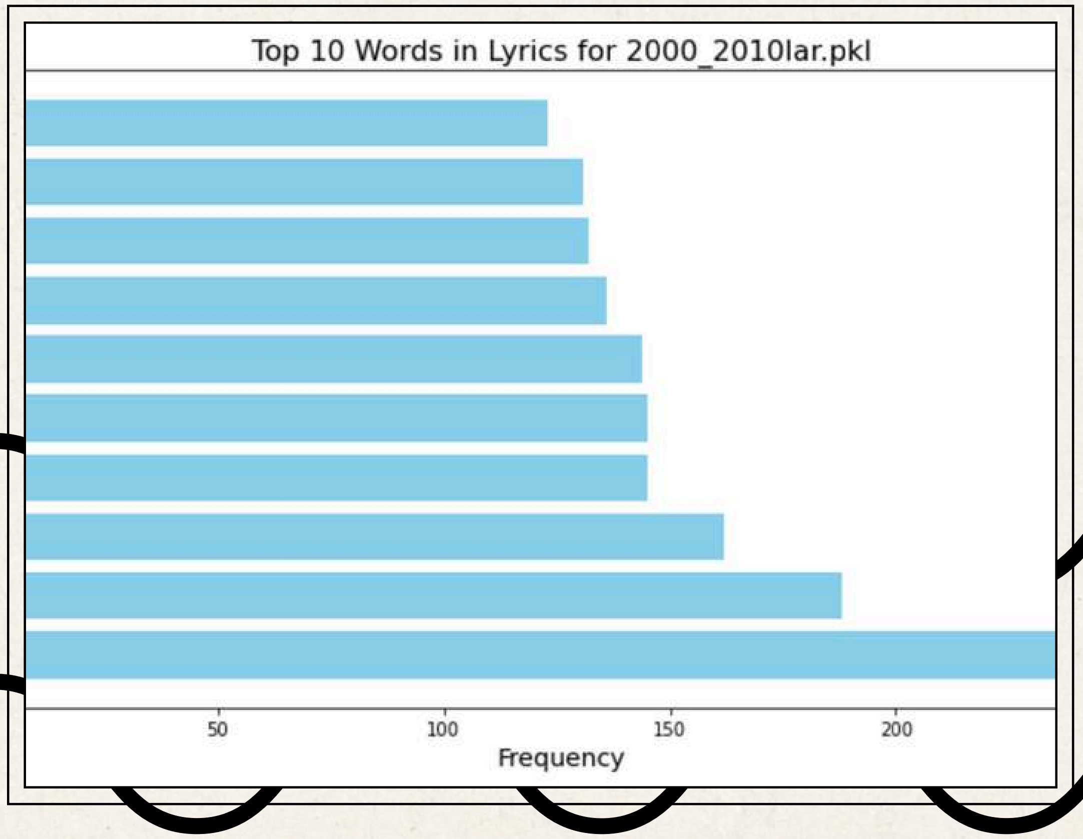
scikit-learn

hugging face

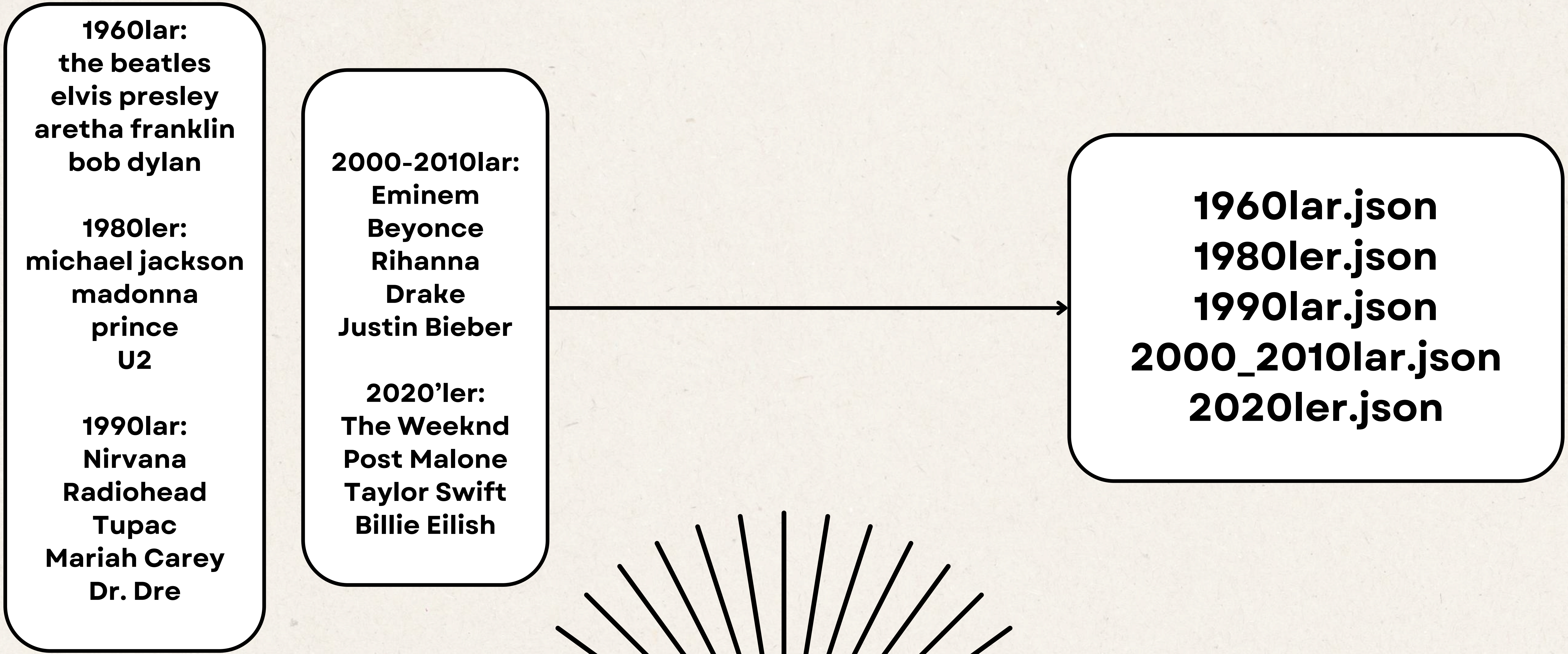
Visualization

matplotlib

seaborn



Datasets



1960lar:
the beatles
elvis presley
aretha franklin
bob dylan

1980ler:
michael jackson
madonna
prince
U2

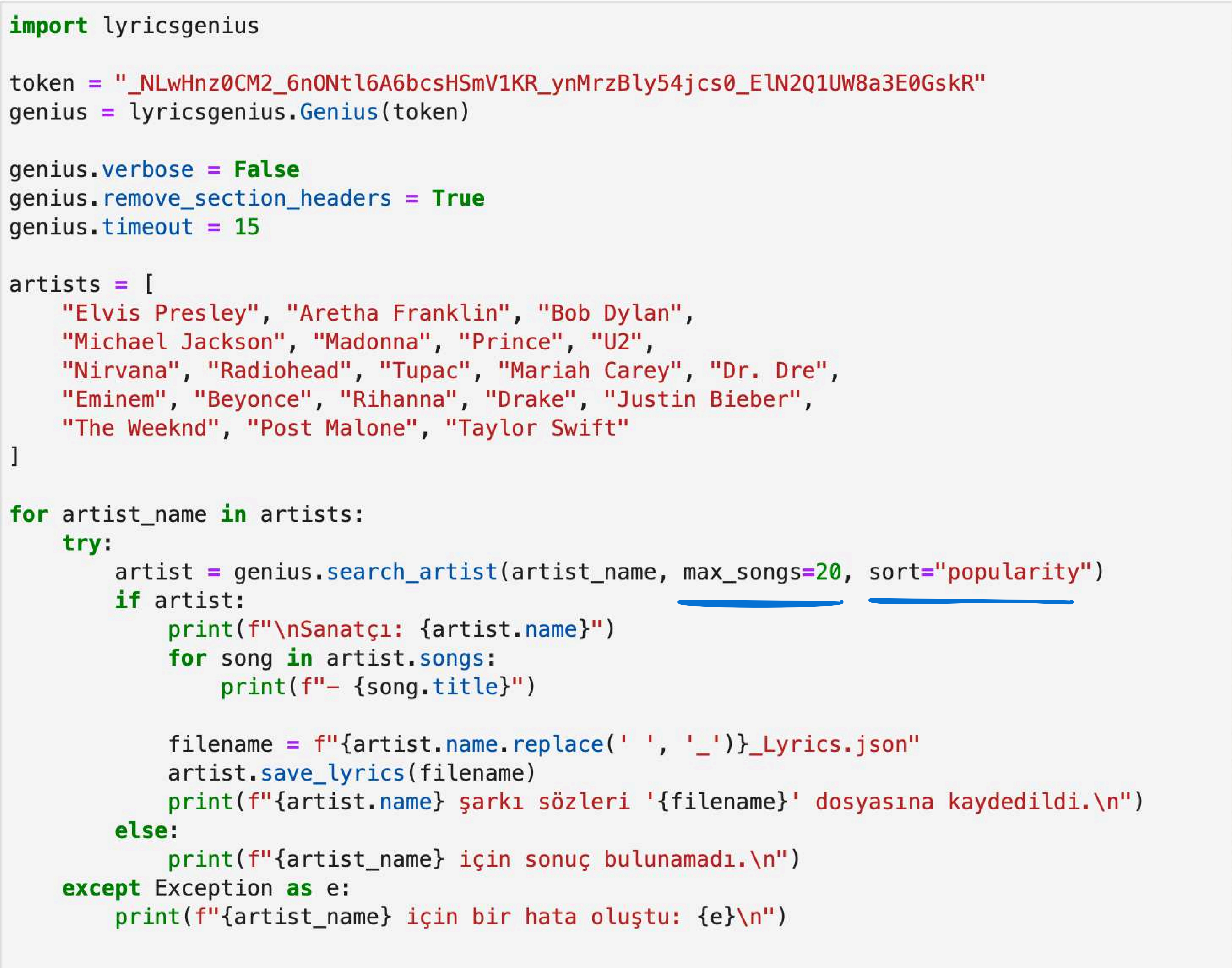
1990lar:
Nirvana
Radiohead
Tupac
Mariah Carey
Dr. Dre

2000-2010lar:
Eminem
Beyonce
Rihanna
Drake
Justin Bieber

2020’ler:
The Weeknd
Post Malone
Taylor Swift
Billie Eilish

1960lar.json
1980ler.json
1990lar.json
2000_2010lar.json
2020ler.json

Datasets

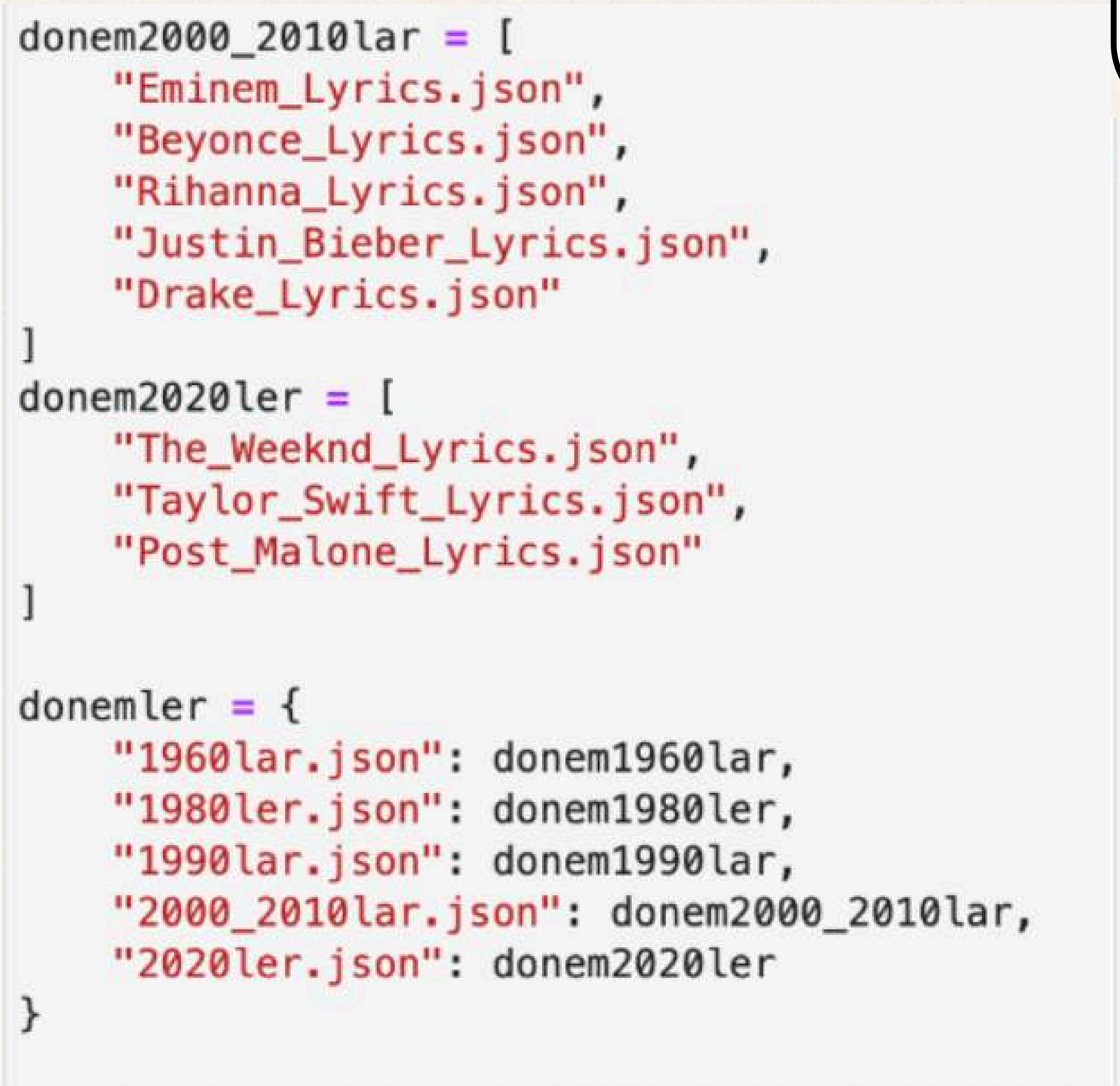


```python
import lyricsgenius

token = "_NLwHnz0CM2_6nONtl6A6bcsHSmV1KR_ynMrzBly54jcs0_ElN2Q1UW8a3E0GskR"
genius = lyricsgenius.Genius(token)

genius.verbose = False
genius.remove_section_headers = True
genius.timeout = 15

artists = [
    "Elvis Presley", "Aretha Franklin", "Bob Dylan",
    "Michael Jackson", "Madonna", "Prince", "U2",
    "Nirvana", "Radiohead", "Tupac", "Mariah Carey", "Dr. Dre",
    "Eminem", "Beyonce", "Rihanna", "Drake", "Justin Bieber",
    "The Weeknd", "Post Malone", "Taylor Swift"
]

for artist_name in artists:
    try:
        artist = genius.search_artist(artist_name, max_songs=20, sort="popularity")
        if artist:
            print(f"\nSanatçı: {artist.name}")
            for song in artist.songs:
                print(f"- {song.title}")

            filename = f"{artist.name.replace(' ', '_')}_Lyrics.json"
            artist.save_lyrics(filename)
            print(f"{artist.name} şarkı sözleri '{filename}' dosyasına kaydedildi.\n")
        else:
            print(f"{artist_name} için sonuç bulunamadı.\n")
    except Exception as e:
        print(f"{artist_name} için bir hata oluştu: {e}\n")
```

Creating Json Files For Artists

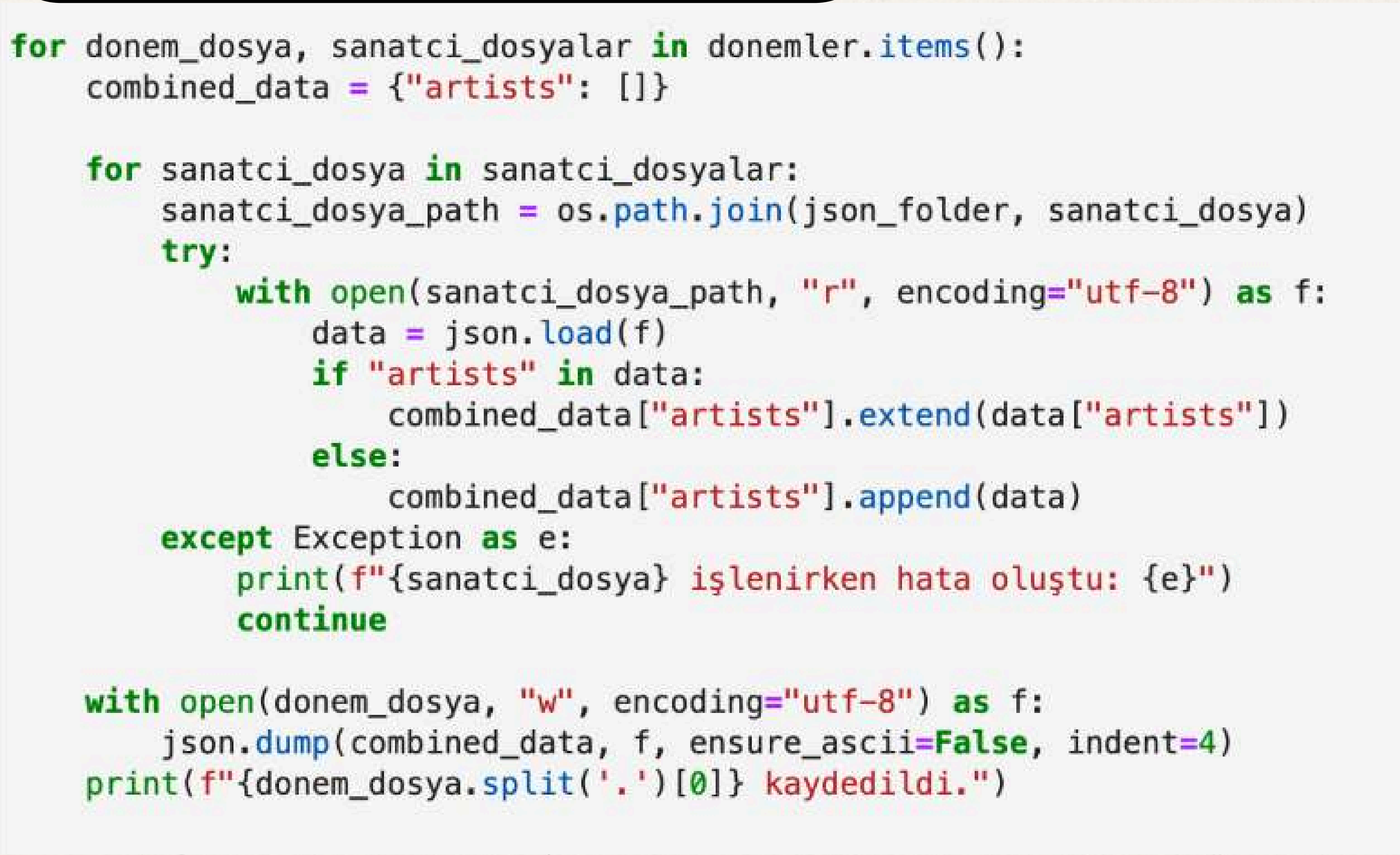
- lyricsgenius
- genius API

Datasets



```python
donem2000_2010lar = [
    "Eminem_Lyrics.json",
    "Beyonce_Lyrics.json",
    "Rihanna_Lyrics.json",
    "Justin_Bieber_Lyrics.json",
    "Drake_Lyrics.json"
]
donem2020ler = [
    "The_Weeknd_Lyrics.json",
    "Taylor_Swift_Lyrics.json",
    "Post_Malone_Lyrics.json"
]

donemler = {
    "1960lar.json": donem1960lar,
    "1980ler.json": donem1980ler,
    "1990lar.json": donem1990lar,
    "2000_2010lar.json": donem2000_2010lar,
    "2020ler.json": donem2020ler
}
```

Creating Json Files for Decades



```python
for donem_dosya, sanatci_dosyalar in donemler.items():
    combined_data = {"artists": []}

    for sanatci_dosya in sanatci_dosyalar:
        sanatci_dosya_path = os.path.join(json_folder, sanatci_dosya)
        try:
            with open(sanatci_dosya_path, "r", encoding="utf-8") as f:
                data = json.load(f)
                if "artists" in data:
                    combined_data["artists"].extend(data["artists"])
                else:
                    combined_data["artists"].append(data)
        except Exception as e:
            print(f"{sanatci_dosya} işlenirken hata oluştu: {e}")
            continue

    with open(donem_dosya, "w", encoding="utf-8") as f:
        json.dump(combined_data, f, ensure_ascii=False, indent=4)
    print(f"{donem_dosya.split('.')[0]} kaydedildi.")
```

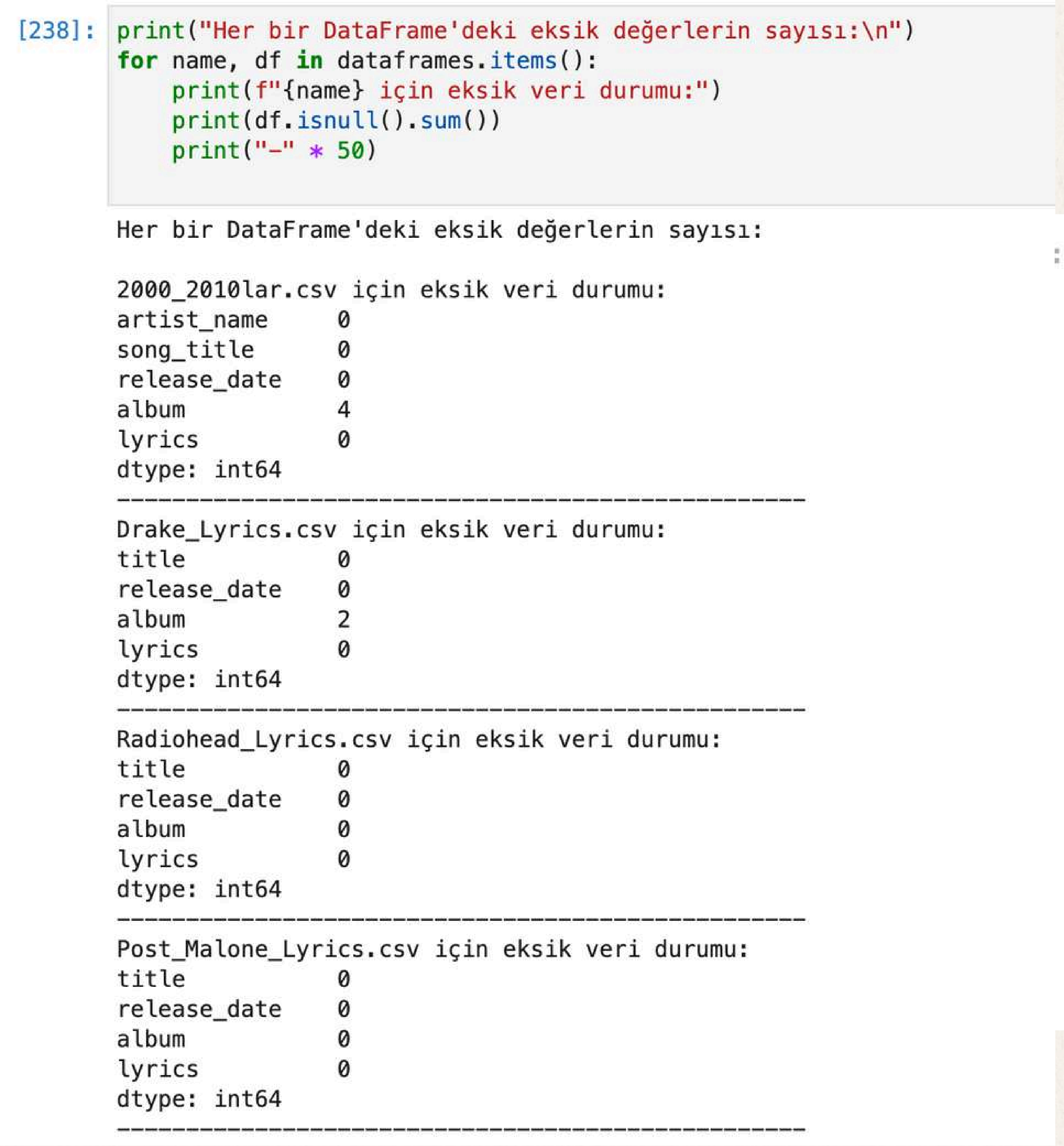
Datasets

Artists & Decades Json Files

Artists and Decades CSV Files

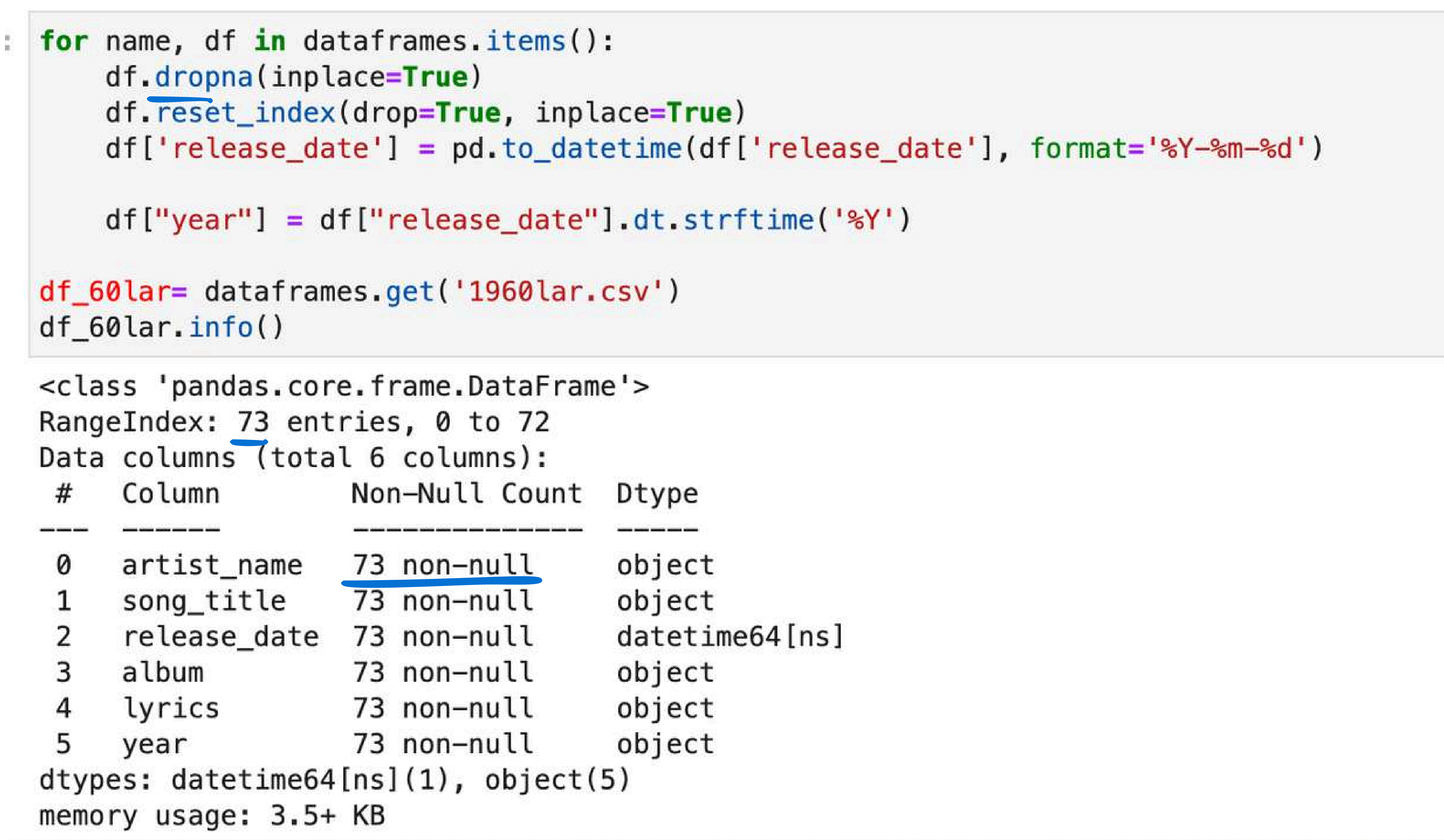
- Better readability
- Standardization for data processing
- Making it compatible with Pandas and other tools
- Providing flexibility for future use

Cleaning Data



```
[238]: print("Her bir DataFrame'deki eksik değerlerin sayısı:\n")
       for name, df in dataframes.items():
           print(f"{name} için eksik veri durumu:")
           print(df.isnull().sum())
           print("-" * 50)
```

```
Her bir DataFrame'deki eksik değerlerin sayısı:

2000_2010lar.csv için eksik veri durumu:
artist_name     0
song_title      0
release_date    0
album           4
lyrics          0
dtype: int64
--------------------------------------------------
Drake_Lyrics.csv için eksik veri durumu:
title           0
release_date    0
album           2
lyrics          0
dtype: int64
--------------------------------------------------
Radiohead_Lyrics.csv için eksik veri durumu:
title           0
release_date    0
album           0
lyrics          0
dtype: int64
--------------------------------------------------
Post_Malone_Lyrics.csv için eksik veri durumu:
title           0
release_date    0
album           0
lyrics          0
dtype: int64
--------------------------------------------------
```



```
for name, df in dataframes.items():
    df.dropna(inplace=True)
    df.reset_index(drop=True, inplace=True)
    df['release_date'] = pd.to_datetime(df['release_date'], format='%Y-%m-%d')

    df["year"] = df["release_date"].dt.strftime('%Y')

df_60lar= dataframes.get('1960lar.csv')
df_60lar.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73 entries, 0 to 72
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   artist_name   73 non-null     object
 1   song_title    73 non-null     object
 2   release_date  73 non-null     datetime64[ns]
 3   album         73 non-null     object
 4   lyrics        73 non-null     object
 5   year          73 non-null     object
dtypes: datetime64[ns](1), object(5)
memory usage: 3.5+ KB
```

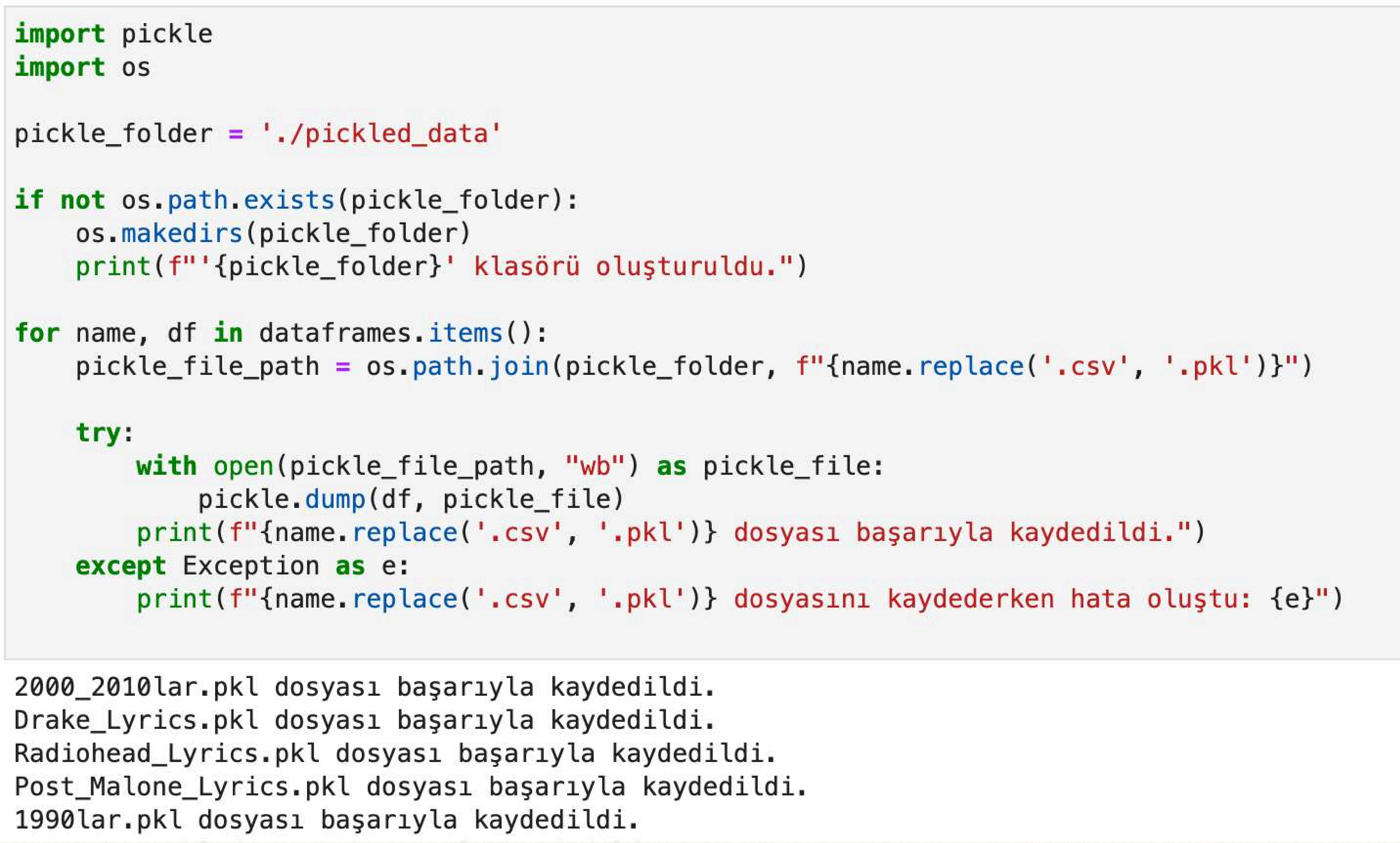
Cleaning Data

Artists and Decades CSV Files

Artists and Decades Pickle Files

- Faster loading
- Less disk space usage
- Data storage compatible with Python
- Data transfer without reprocessing

Datasets

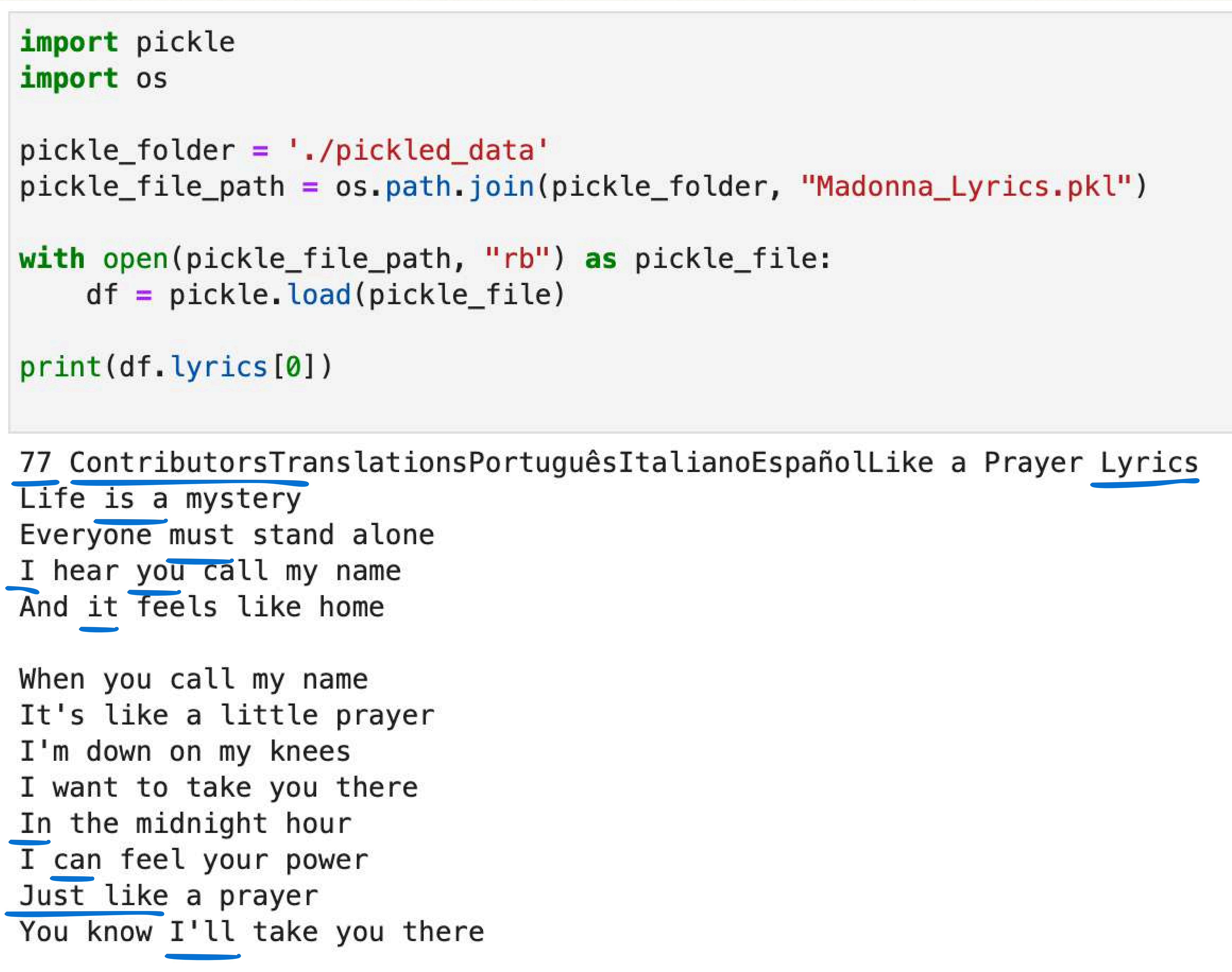


```
import pickle
import os

pickle_folder = './pickled_data'

if not os.path.exists(pickle_folder):
    os.makedirs(pickle_folder)
    print(f"'{pickle_folder}' klasörü oluşturuldu.")

for name, df in dataframes.items():
    pickle_file_path = os.path.join(pickle_folder, f"{name.replace('.csv', '.pkl')}")

    try:
        with open(pickle_file_path, "wb") as pickle_file:
            pickle.dump(df, pickle_file)
        print(f"{name.replace('.csv', '.pkl')} dosyası başarıyla kaydedildi.")
    except Exception as e:
        print(f"{name.replace('.csv', '.pkl')} dosyasını kaydederken hata oluştu: {e}")
```

```
2000_2010lar.pkl dosyası başarıyla kaydedildi.
Drake_Lyrics.pkl dosyası başarıyla kaydedildi.
Radiohead_Lyrics.pkl dosyası başarıyla kaydedildi.
Post_Malone_Lyrics.pkl dosyası başarıyla kaydedildi.
1990lar.pkl dosyası başarıyla kaydedildi.
```

- **pickle**
- **os**

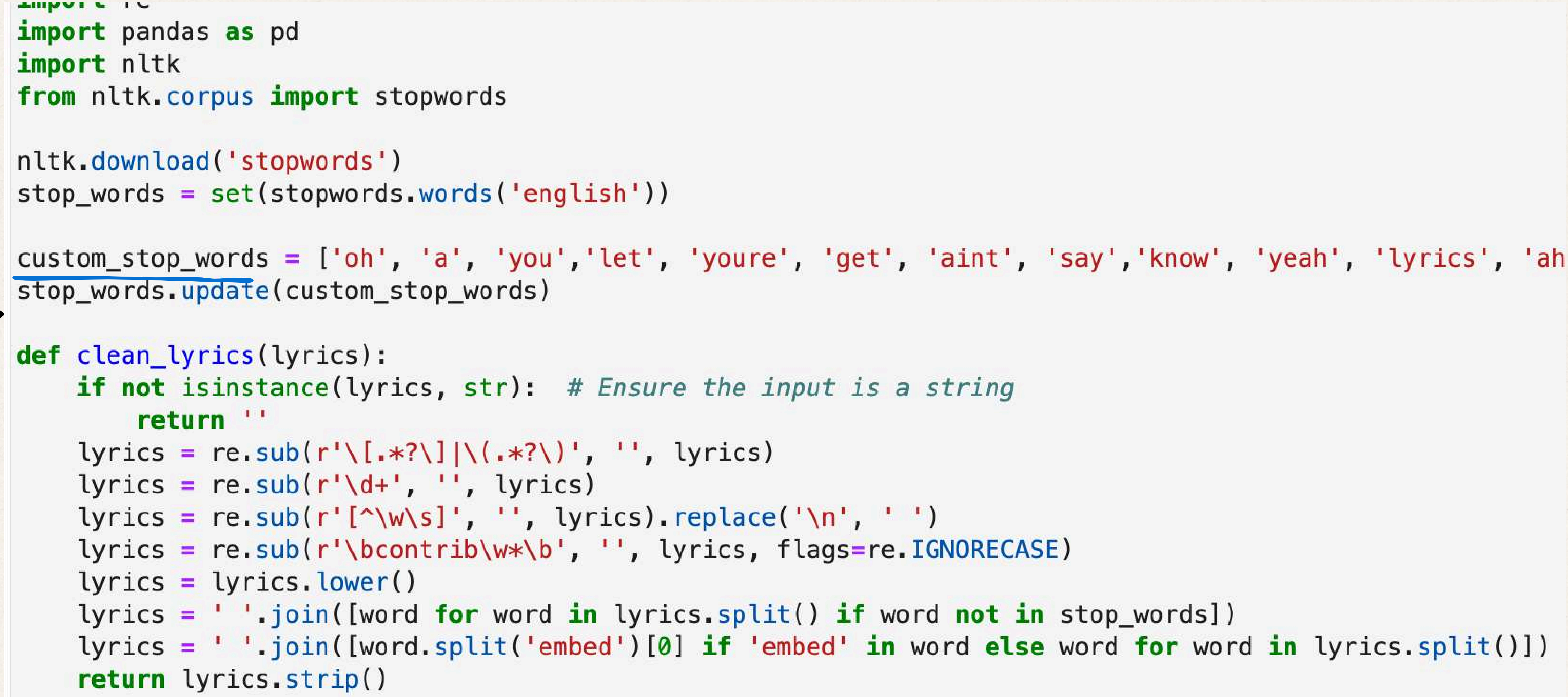
Cleaning Data



```
import pickle
import os

pickle_folder = './pickled_data'
pickle_file_path = os.path.join(pickle_folder, "Madonna_Lyrics.pkl")

with open(pickle_file_path, "rb") as pickle_file:
    df = pickle.load(pickle_file)

print(df.lyrics[0])
```

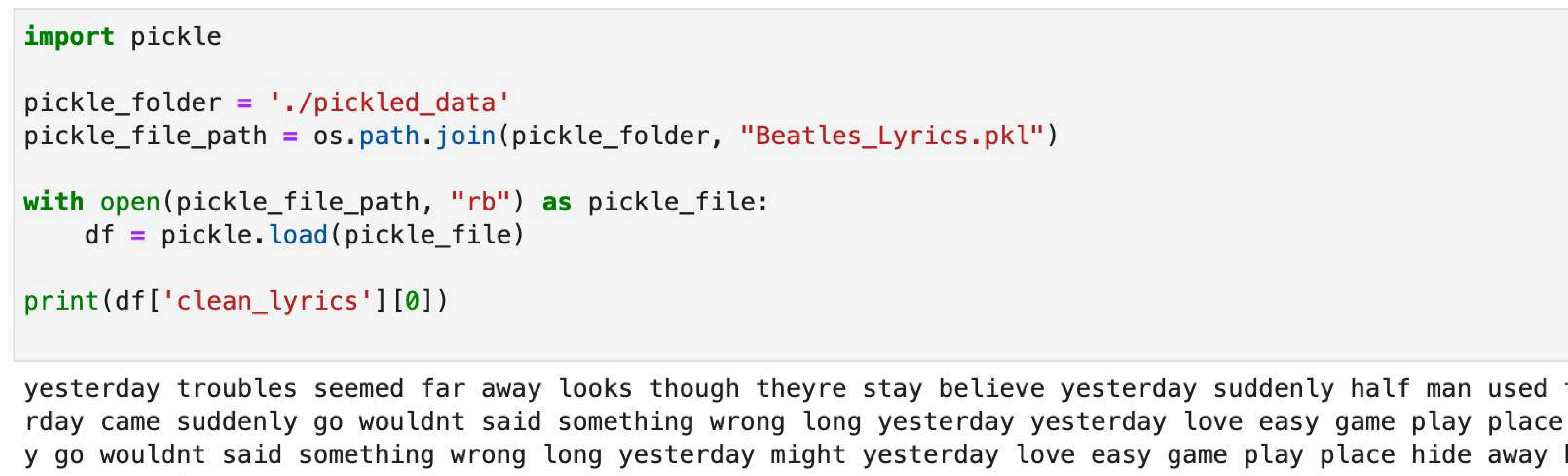
```
77 ContributorsTranslationsPortuguêsItalianoEspañolLike a Prayer Lyrics
Life is a mystery
Everyone must stand alone
I hear you call my name
And it feels like home

When you call my name
It's like a little prayer
I'm down on my knees
I want to take you there
In the midnight hour
I can feel your power
Just like a prayer
You know I'll take you there
```

Cleaning Data



```python
import re
import pandas as pd
import nltk
from nltk.corpus import stopwords

nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

custom_stop_words = ['oh', 'a', 'you','let', 'youre', 'get', 'aint', 'say','know', 'yeah', 'lyrics', 'ah
stop_words.update(custom_stop_words)

def clean_lyrics(lyrics):
    if not isinstance(lyrics, str):  # Ensure the input is a string
        return ''
    lyrics = re.sub(r'\[.*?\]|\(.*?\)', '', lyrics)
    lyrics = re.sub(r'\d+', '', lyrics)
    lyrics = re.sub(r'[^\w\s]', '', lyrics).replace('\n', ' ')
    lyrics = re.sub(r'\bcontrib\w*\b', '', lyrics, flags=re.IGNORECASE)
    lyrics = lyrics.lower()
    lyrics = ' '.join([word for word in lyrics.split() if word not in stop_words])
    lyrics = ' '.join([word.split('embed')[0] if 'embed' in word else word for word in lyrics.split()])
    return lyrics.strip()
```

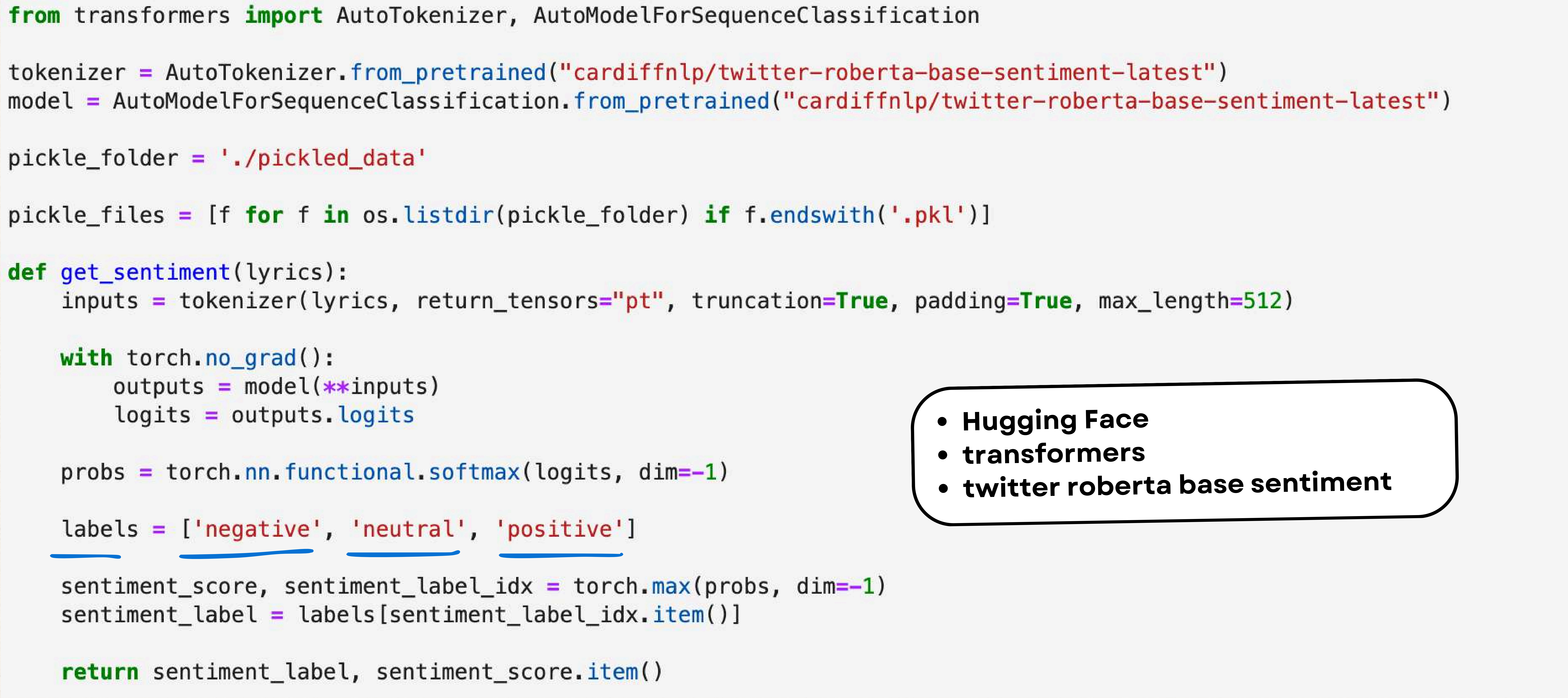
Cleaning Data



```python
import pickle

pickle_folder = './pickled_data'
pickle_file_path = os.path.join(pickle_folder, "Beatles_Lyrics.pkl")

with open(pickle_file_path, "rb") as pickle_file:
    df = pickle.load(pickle_file)

print(df['clean_lyrics'][0])
```

```
yesterday troubles seemed far away looks though theyre stay believe yesterday suddenly half man used
rday came suddenly go wouldnt said something wrong long yesterday yesterday love easy game play place
y go wouldnt said something wrong long yesterday might yesterday love easy game play place hide away
```

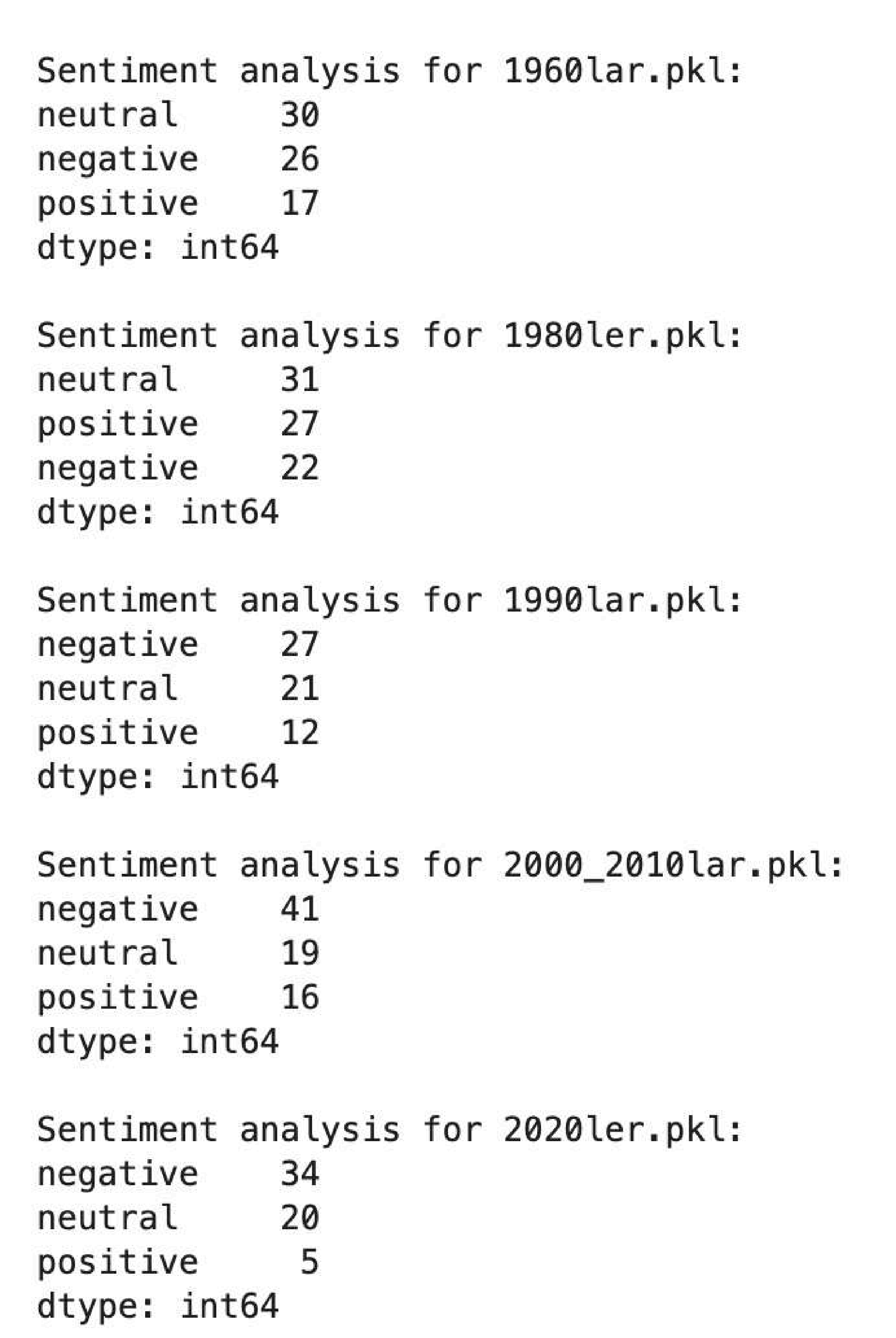
Model ve Analysis



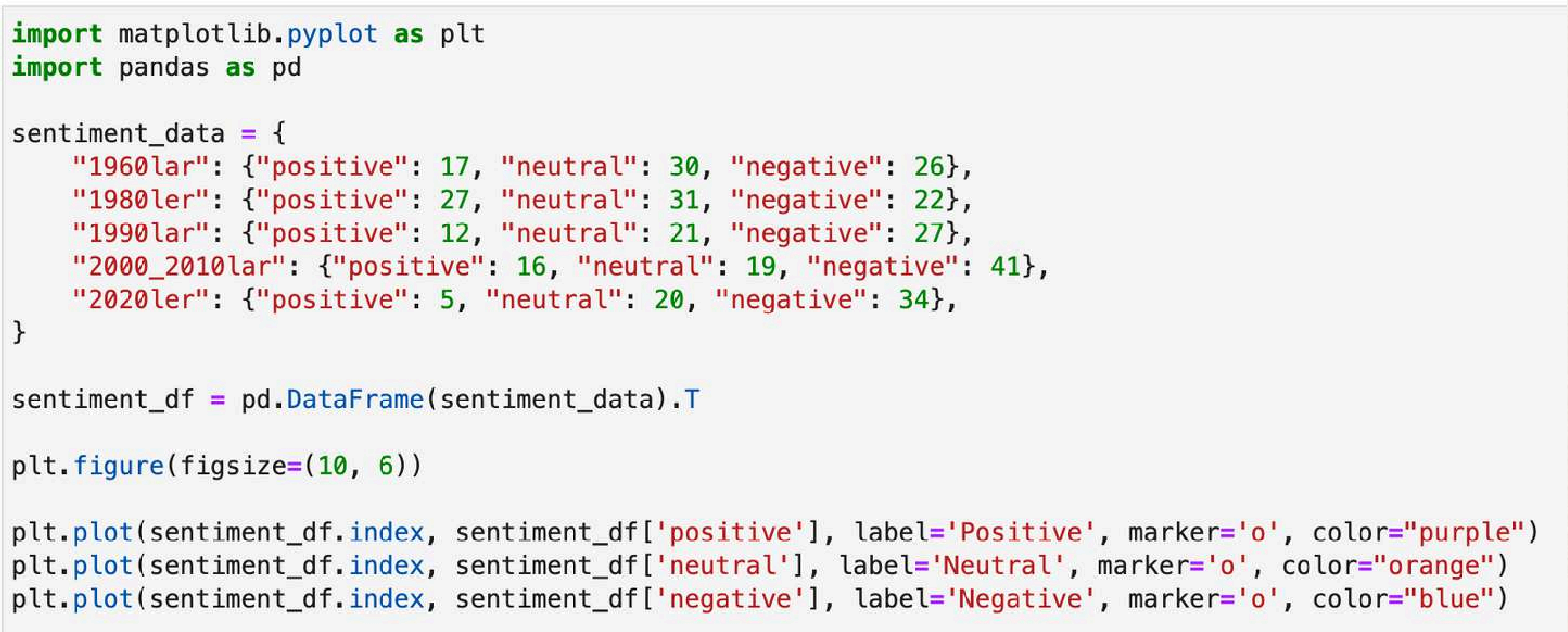
```
from transformers import AutoTokenizer, AutoModelForSequenceClassification

tokenizer = AutoTokenizer.from_pretrained("cardiffnlp/twitter-roberta-base-sentiment-latest")
model = AutoModelForSequenceClassification.from_pretrained("cardiffnlp/twitter-roberta-base-sentiment-latest")

pickle_folder = './pickled_data'

pickle_files = [f for f in os.listdir(pickle_folder) if f.endswith('.pkl')]

def get_sentiment(lyrics):
    inputs = tokenizer(lyrics, return_tensors="pt", truncation=True, padding=True, max_length=512)

    with torch.no_grad():
        outputs = model(**inputs)
        logits = outputs.logits

    probs = torch.nn.functional.softmax(logits, dim=-1)

    labels = ['negative', 'neutral', 'positive']

    sentiment_score, sentiment_label_idx = torch.max(probs, dim=-1)
    sentiment_label = labels[sentiment_label_idx.item()]

    return sentiment_label, sentiment_score.item()
```

- **Hugging Face**
- **transformers**
- **twitter roberta base sentiment**

Model ve Analysis



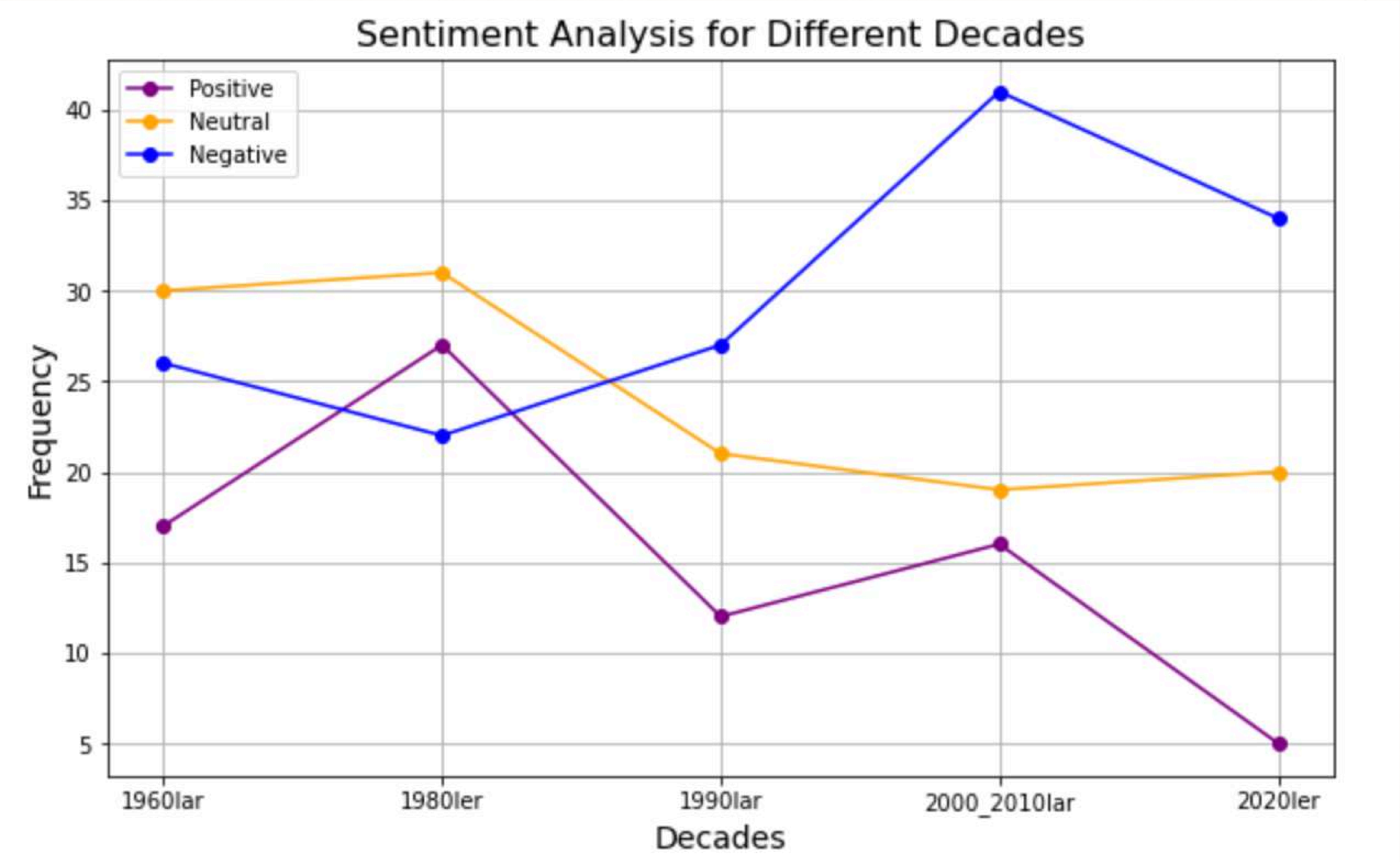
```
Sentiment analysis for 1960lar.pkl:
neutral     30
negative    26
positive    17
dtype: int64

Sentiment analysis for 1980ler.pkl:
neutral     31
positive    27
negative    22
dtype: int64

Sentiment analysis for 1990lar.pkl:
negative    27
neutral     21
positive    12
dtype: int64

Sentiment analysis for 2000_2010lar.pkl:
negative    41
neutral     19
positive    16
dtype: int64

Sentiment analysis for 2020ler.pkl:
negative    34
neutral     20
positive     5
dtype: int64
```

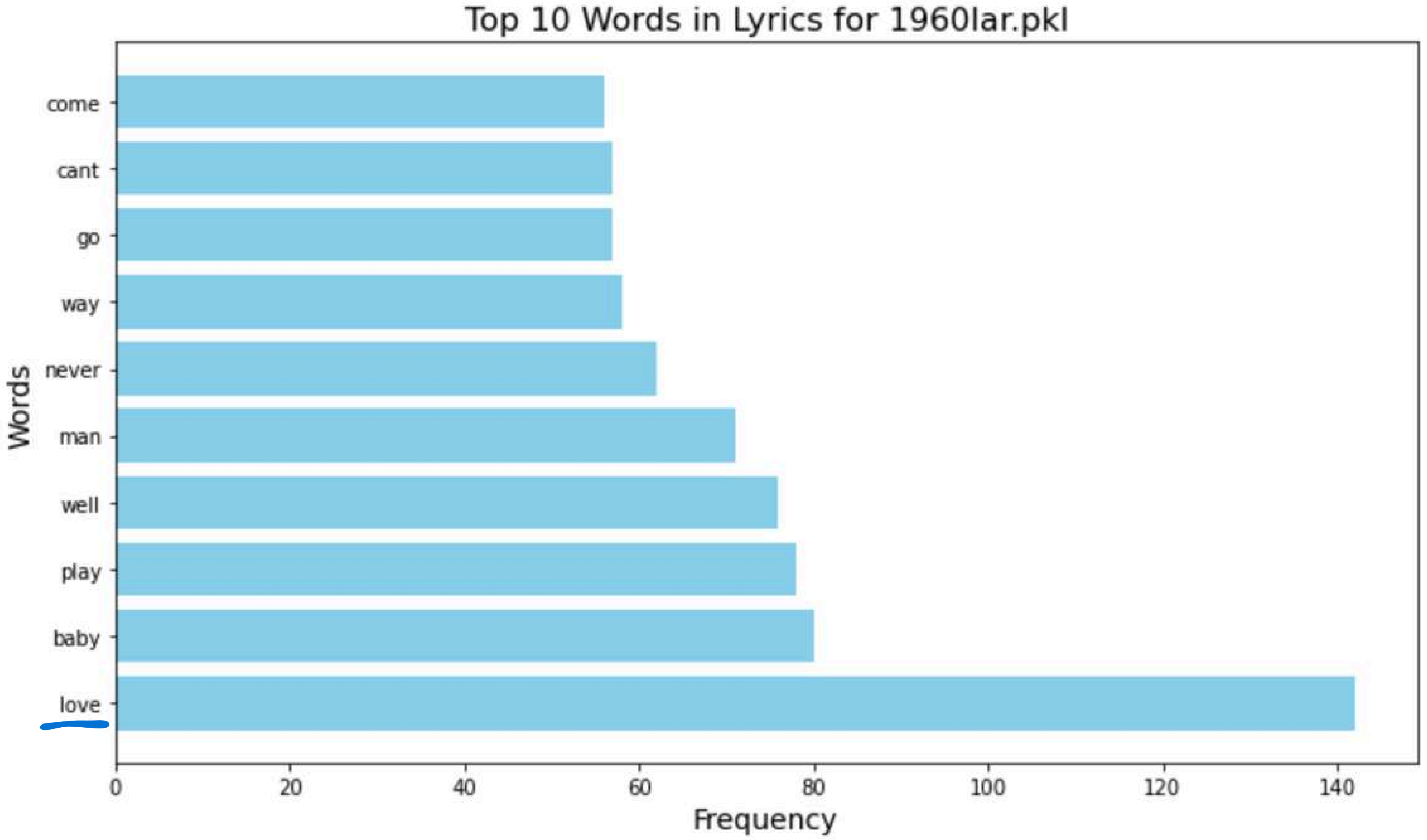


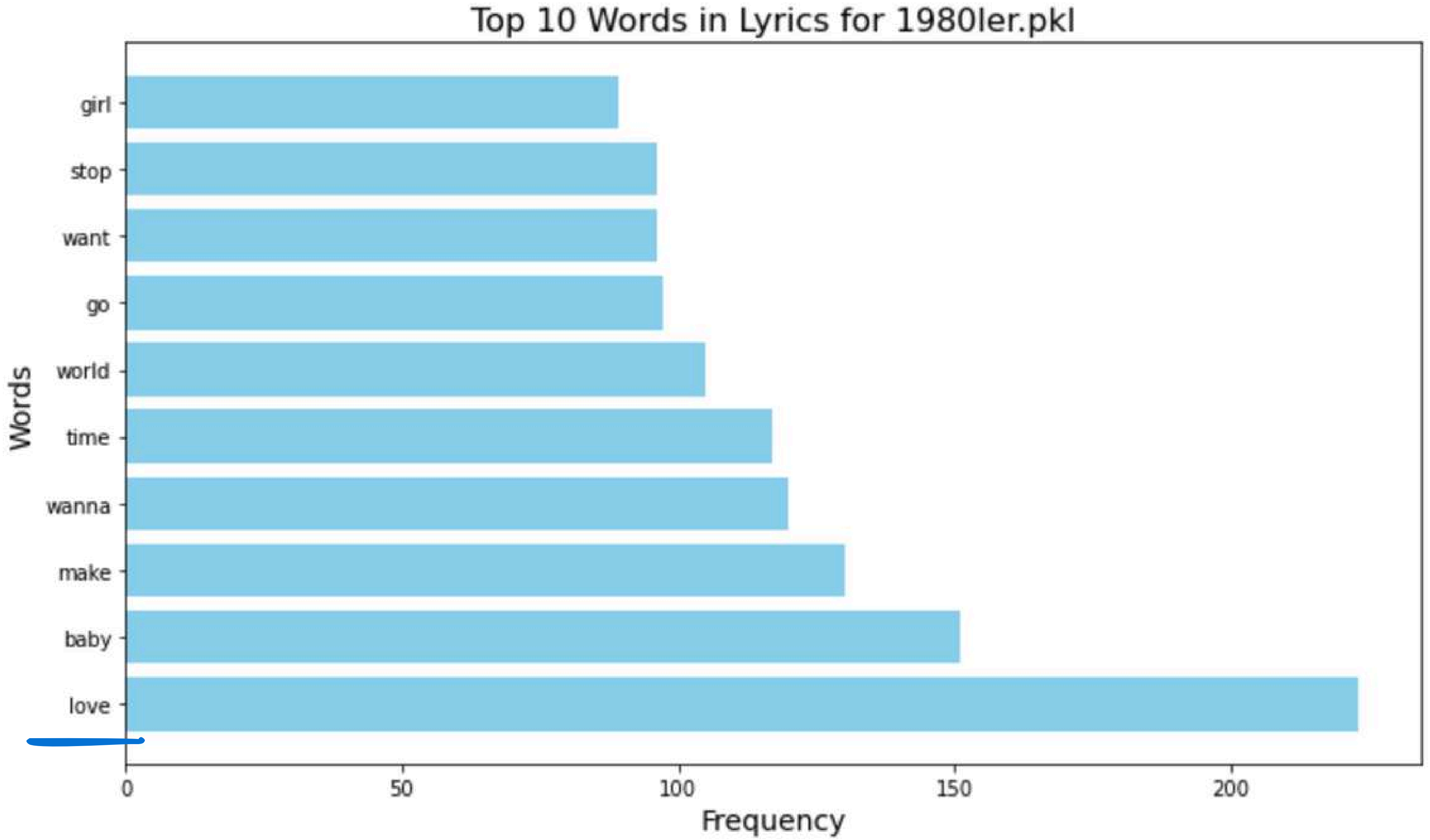
```python
import matplotlib.pyplot as plt
import pandas as pd

sentiment_data = {
    "1960lar": {"positive": 17, "neutral": 30, "negative": 26},
    "1980ler": {"positive": 27, "neutral": 31, "negative": 22},
    "1990lar": {"positive": 12, "neutral": 21, "negative": 27},
    "2000_2010lar": {"positive": 16, "neutral": 19, "negative": 41},
    "2020ler": {"positive": 5, "neutral": 20, "negative": 34},
}

sentiment_df = pd.DataFrame(sentiment_data).T

plt.figure(figsize=(10, 6))

plt.plot(sentiment_df.index, sentiment_df['positive'], label='Positive', marker='o', color="purple")
plt.plot(sentiment_df.index, sentiment_df['neutral'], label='Neutral', marker='o', color="orange")
plt.plot(sentiment_df.index, sentiment_df['negative'], label='Negative', marker='o', color="blue")
```

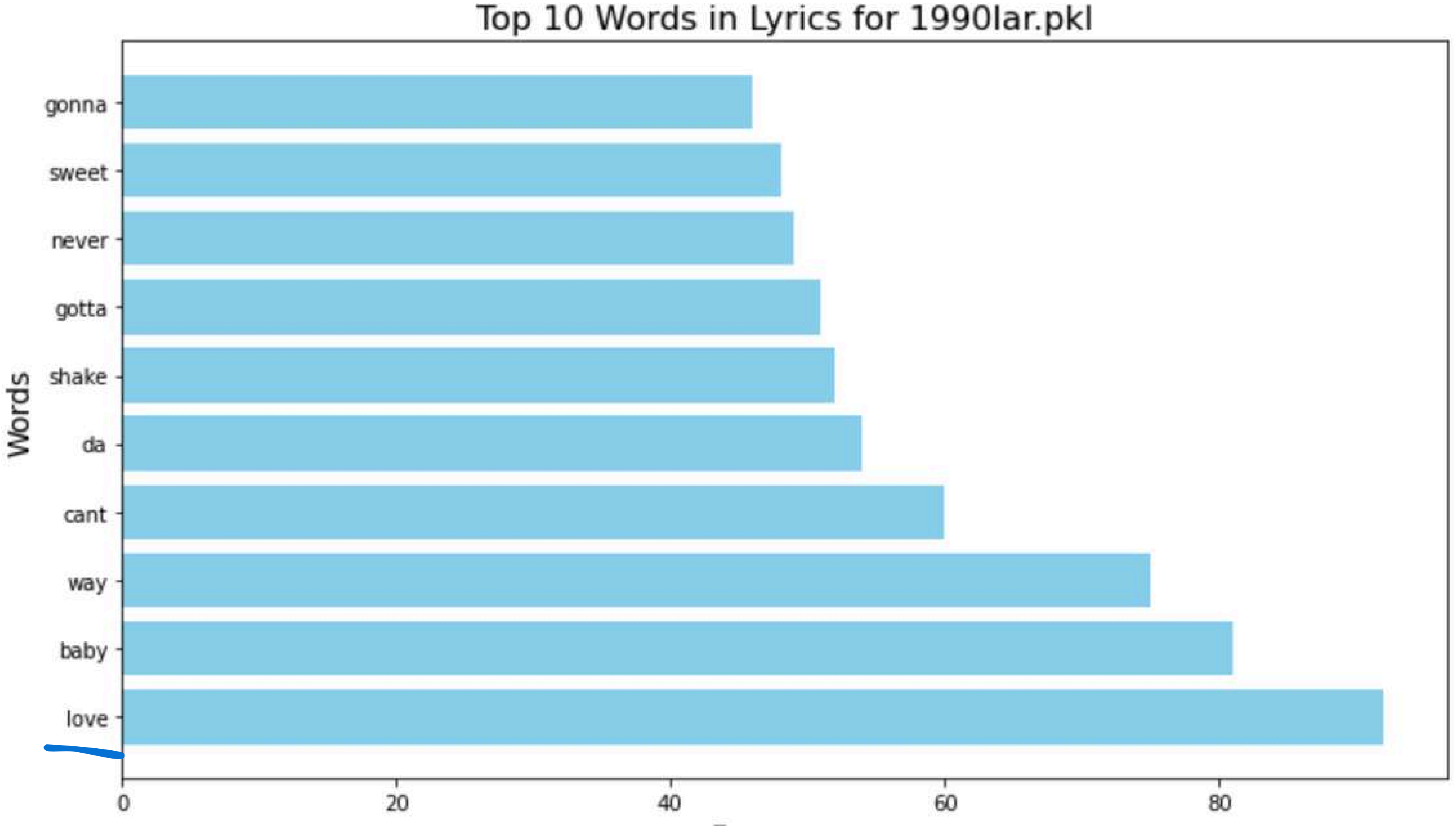
- matplotlib

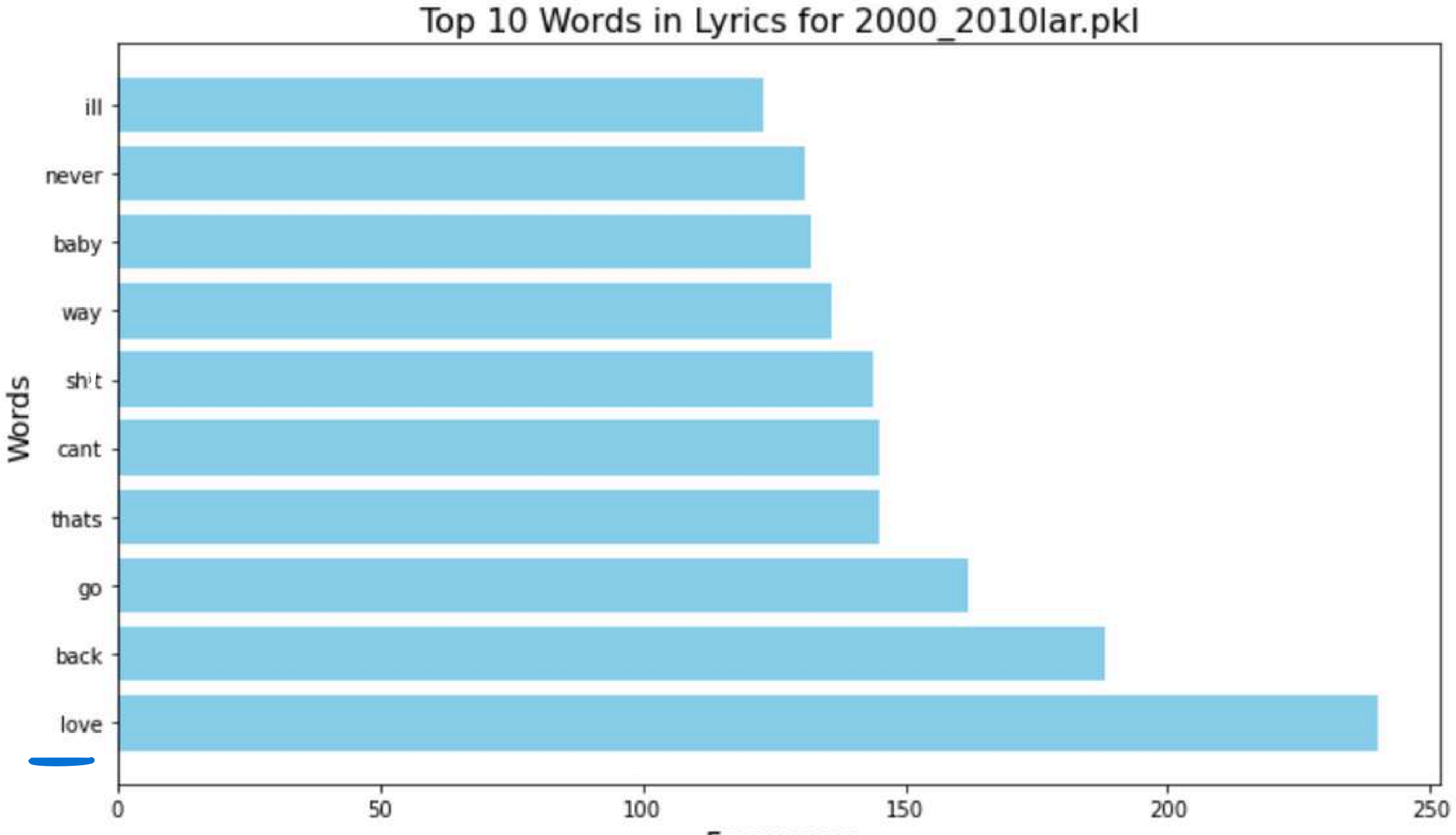
Visualization

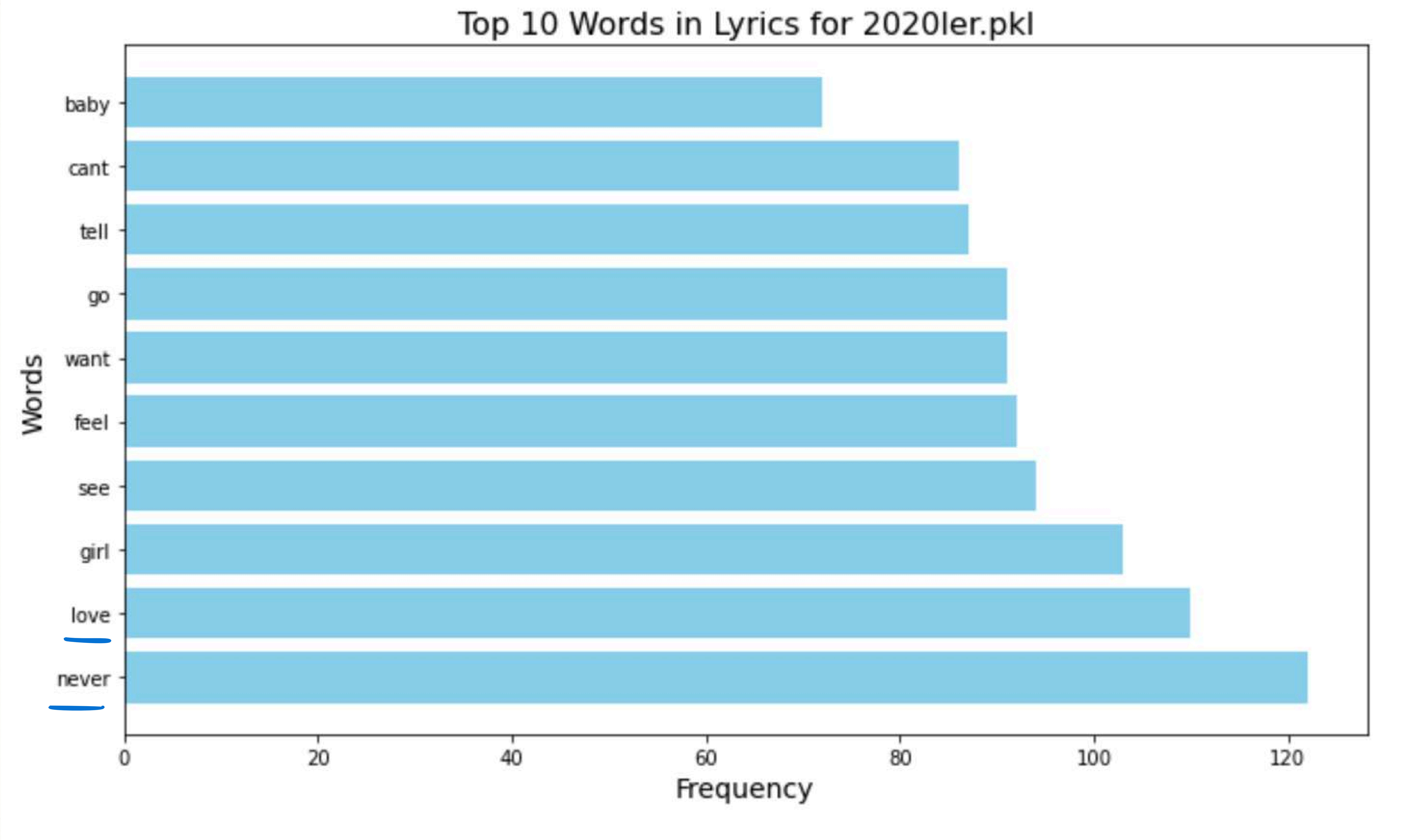


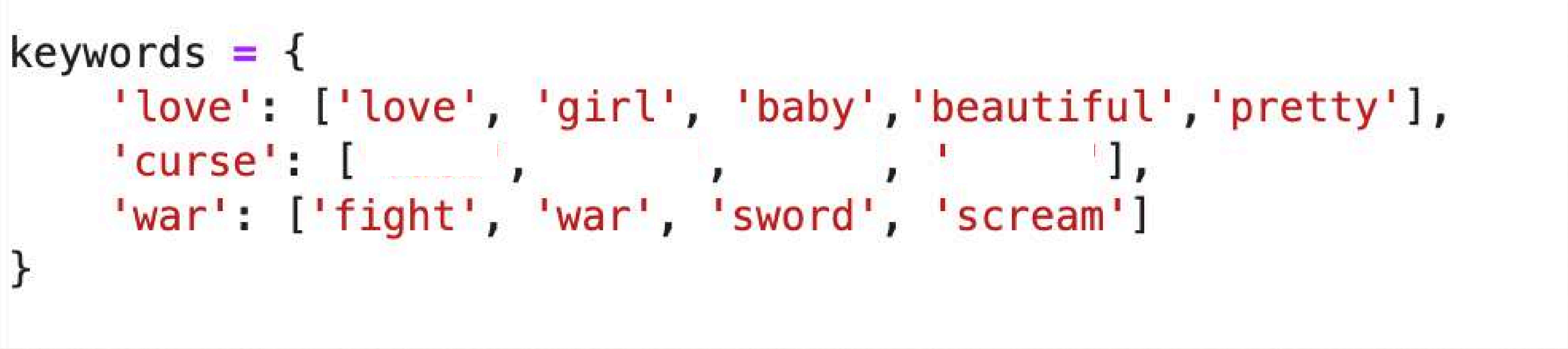

Top 10 Words in Lyrics for 1960lar.pkl


Top 10 Words in Lyrics for 1980ler.pkl

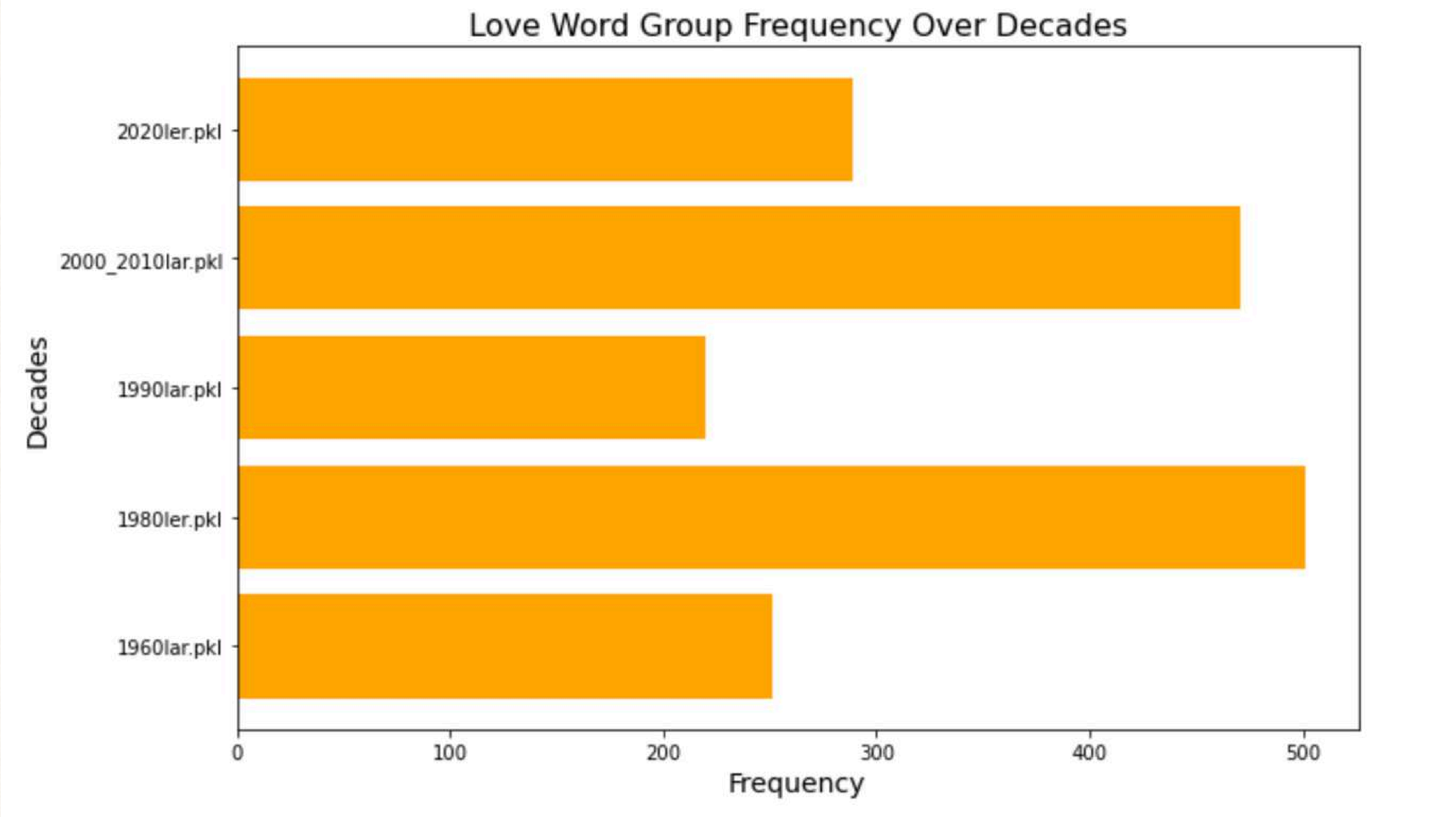

Top 10 Words in Lyrics for 1990lar.pkl


Top 10 Words in Lyrics for 2000_2010lar.pkl


Top 10 Words in Lyrics for 2020ler.pkl
Words
baby
cant
tell
go
want
feel
see
girl
love
never
0
20
40
60
80
100
120
Frequency

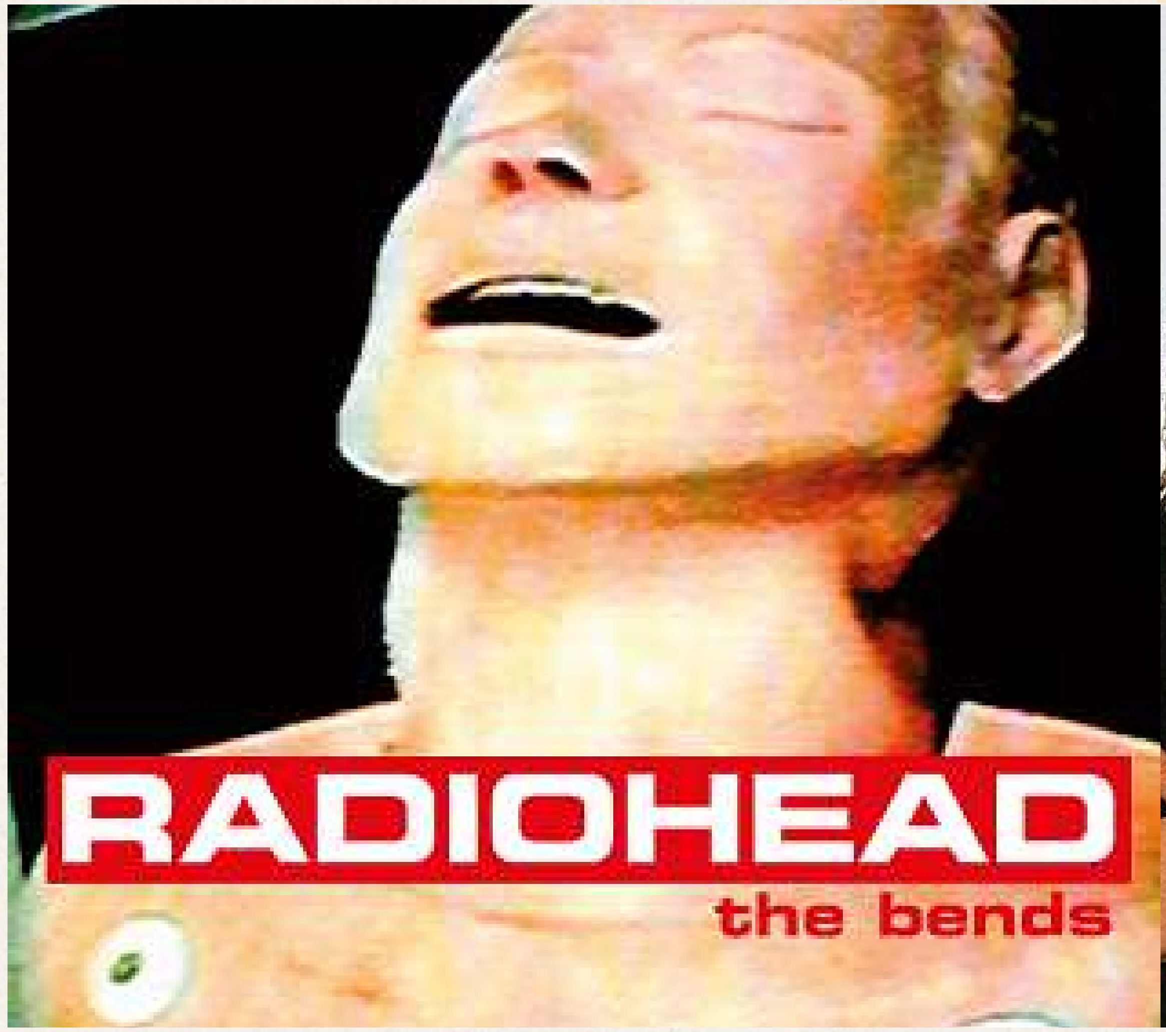


```
keywords = {
    'love': ['love', 'girl', 'baby','beautiful','pretty'],
    'curse': [[redacted]', [redacted], [redacted], '[redacted]'],
    'war': ['fight', 'war', 'sword', 'scream']
}
```



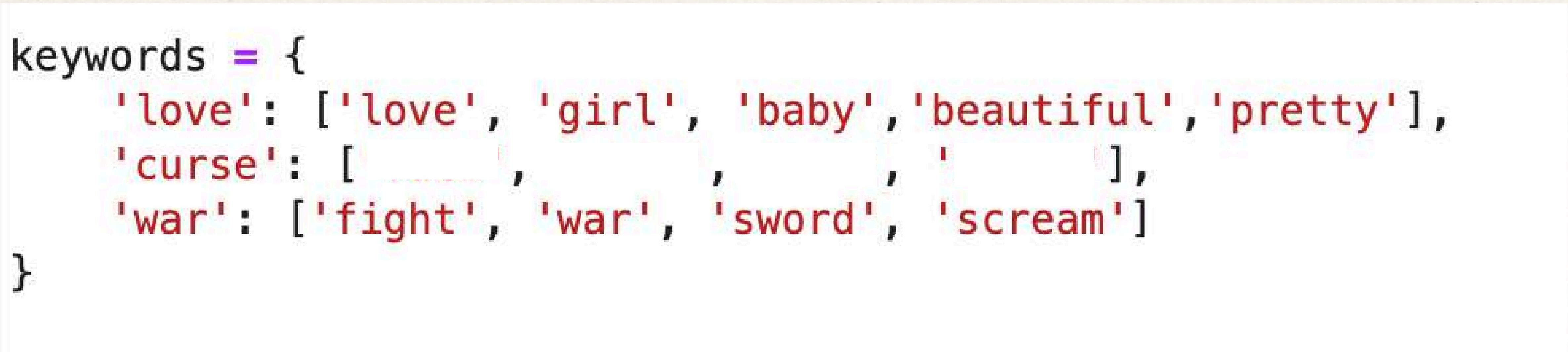
1980s

- Pop
- Romance Ballad

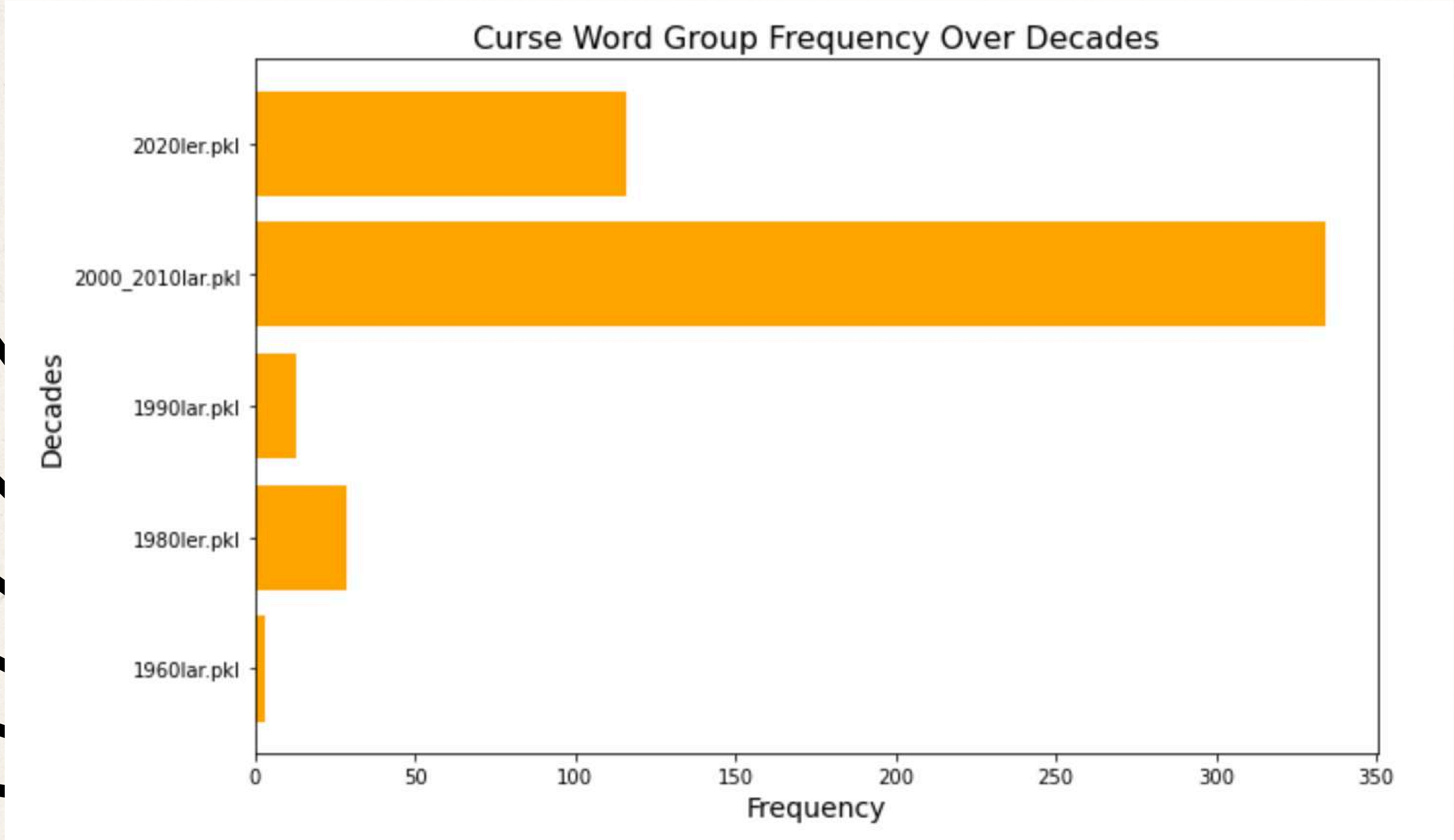


1990s

- Grunge
- Alternative
- Rebellion

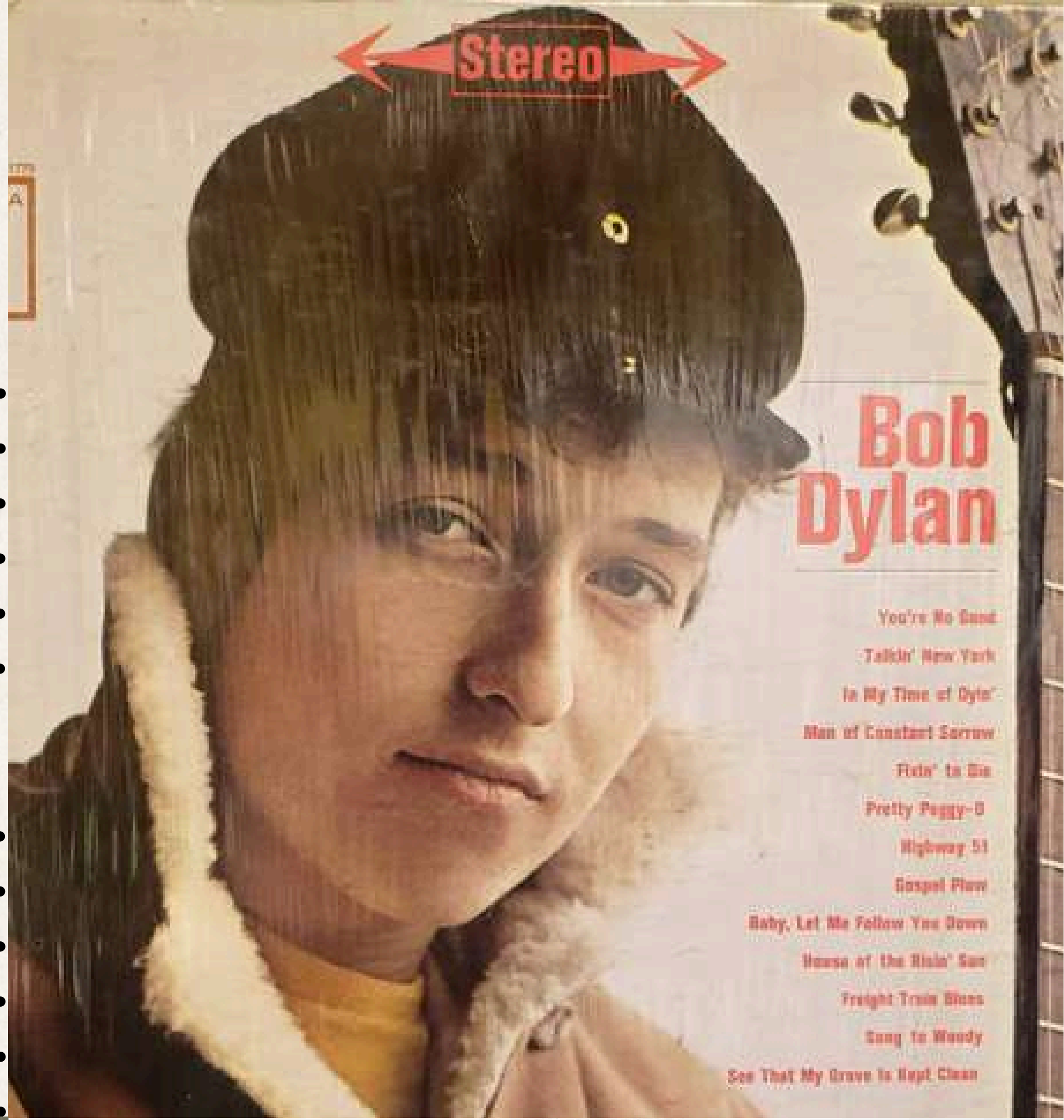


```
keywords = {
    'love': ['love', 'girl', 'baby','beautiful','pretty'],
    'curse': [[redacted]', [redacted], [redacted], '[redacted]'],
    'war': ['fight', 'war', 'sword', 'scream']
}
```



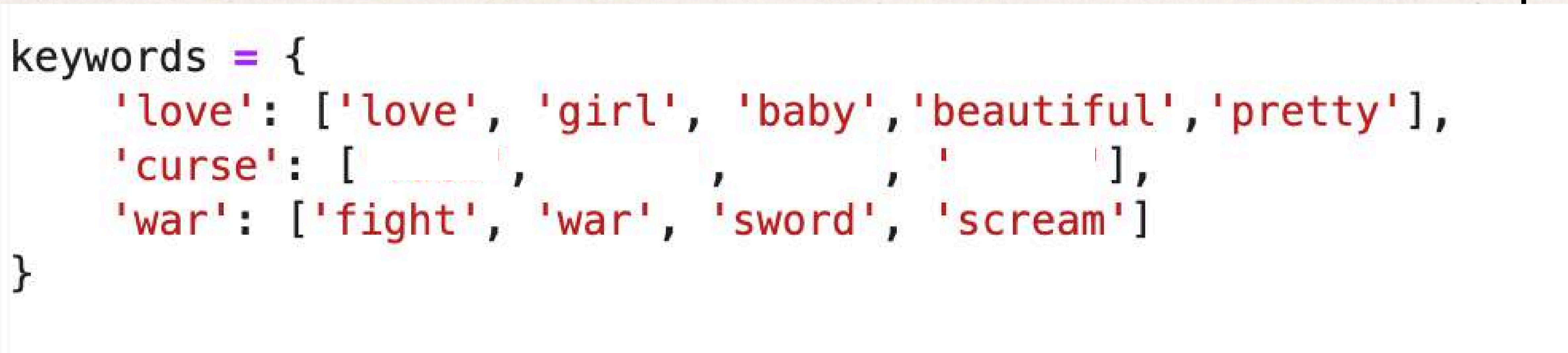
1960s

- Conservatism
- Censorship

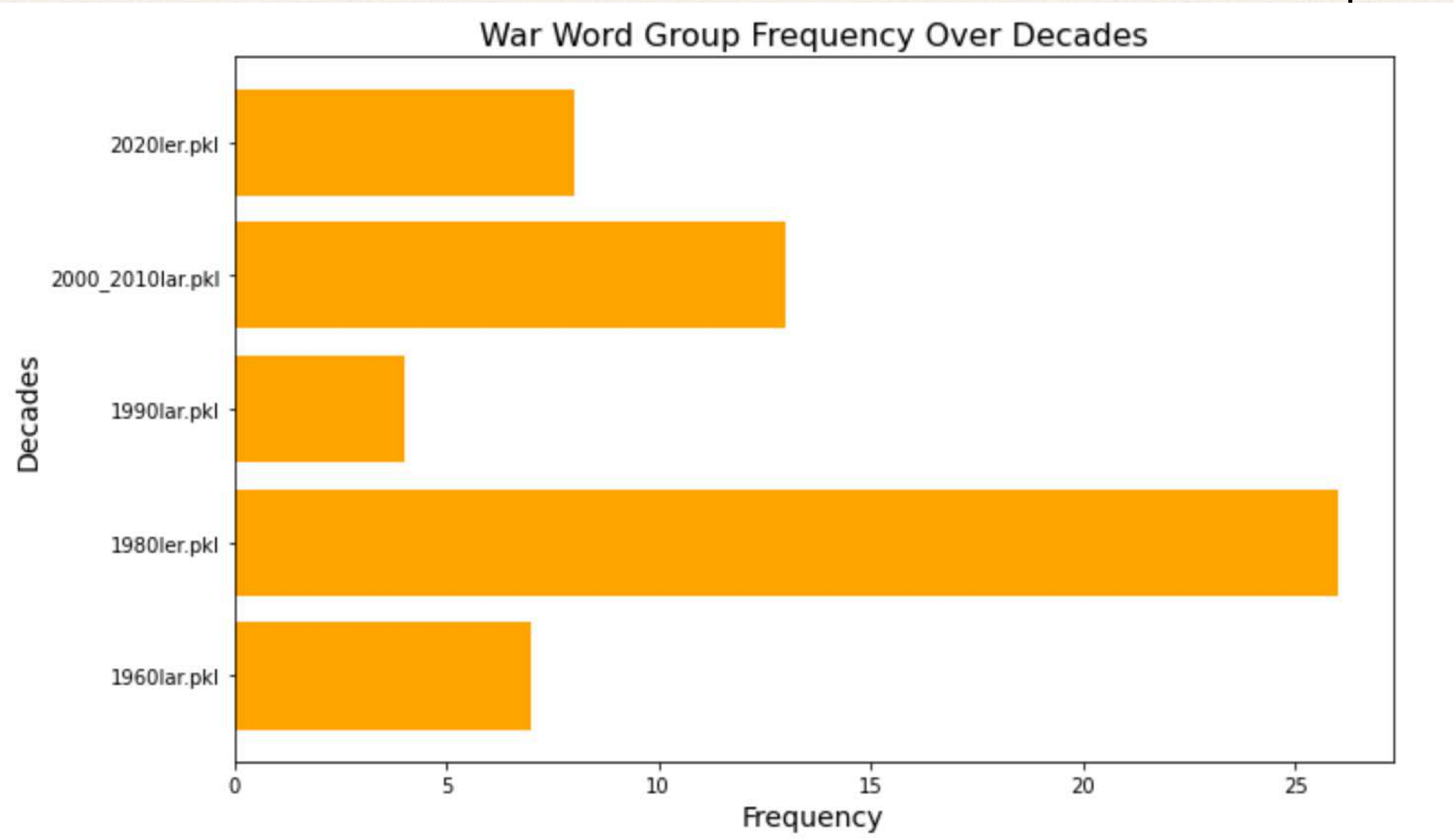


2000s

- Gangsta Culture
- Rap and Hiphop
- Language without censors

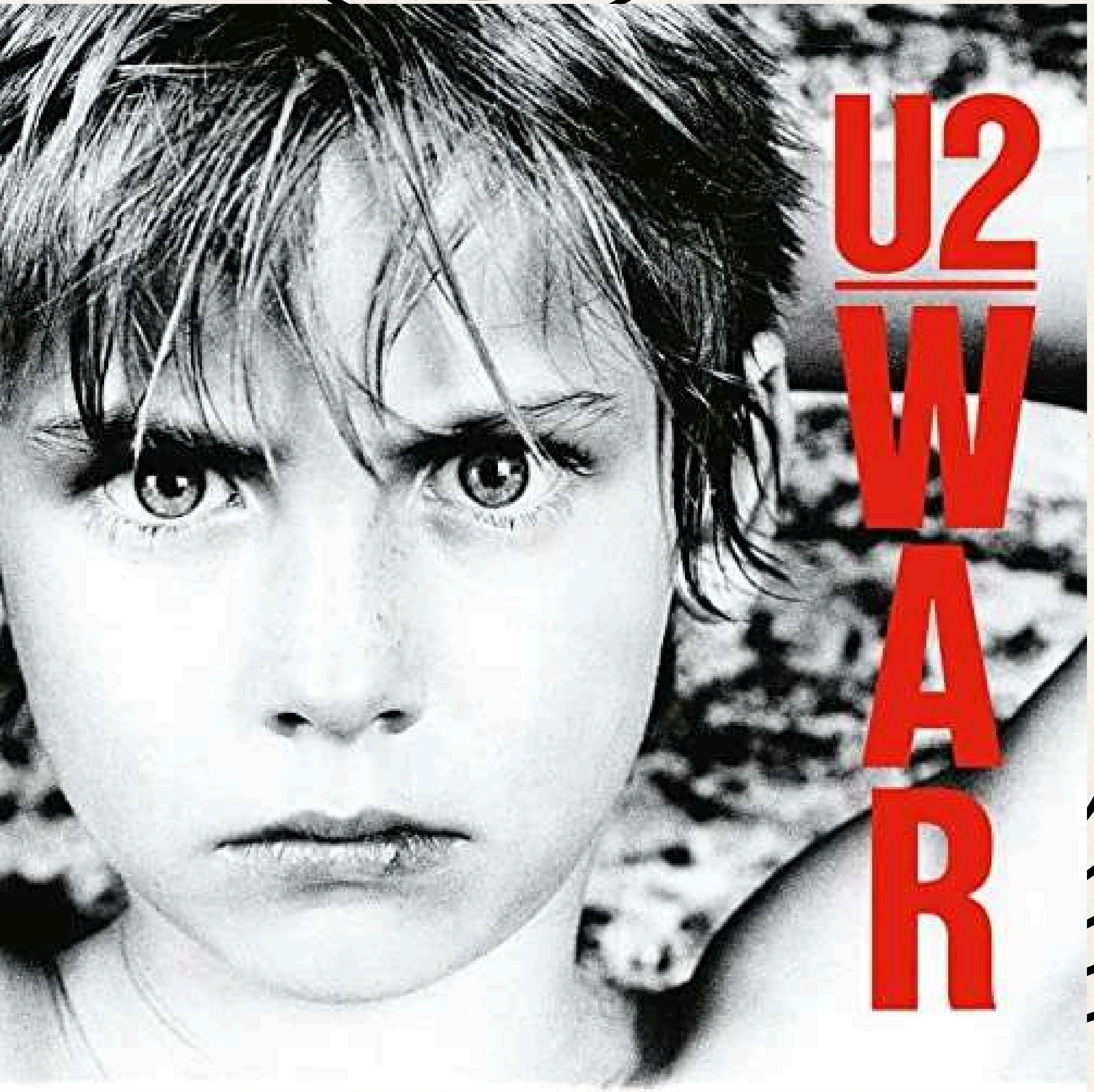


```
keywords = {
    'love': ['love', 'girl', 'baby','beautiful','pretty'],
    'curse': [[redacted], [redacted], [redacted], [redacted]],
    'war': ['fight', 'war', 'sword', 'scream']
}
```



1980s

- End of Cold War
- Protest Music



Result

This project examined the emotional and thematic changes in song lyrics over time. The analyses showed an increase in negative emotions and profanity, as well as a more prominent reflection of societal events like war in the lyrics. These findings reveal that music is a powerful tool reflecting the mood of society and the socio-political structure of the era.

Thank you.

Sources

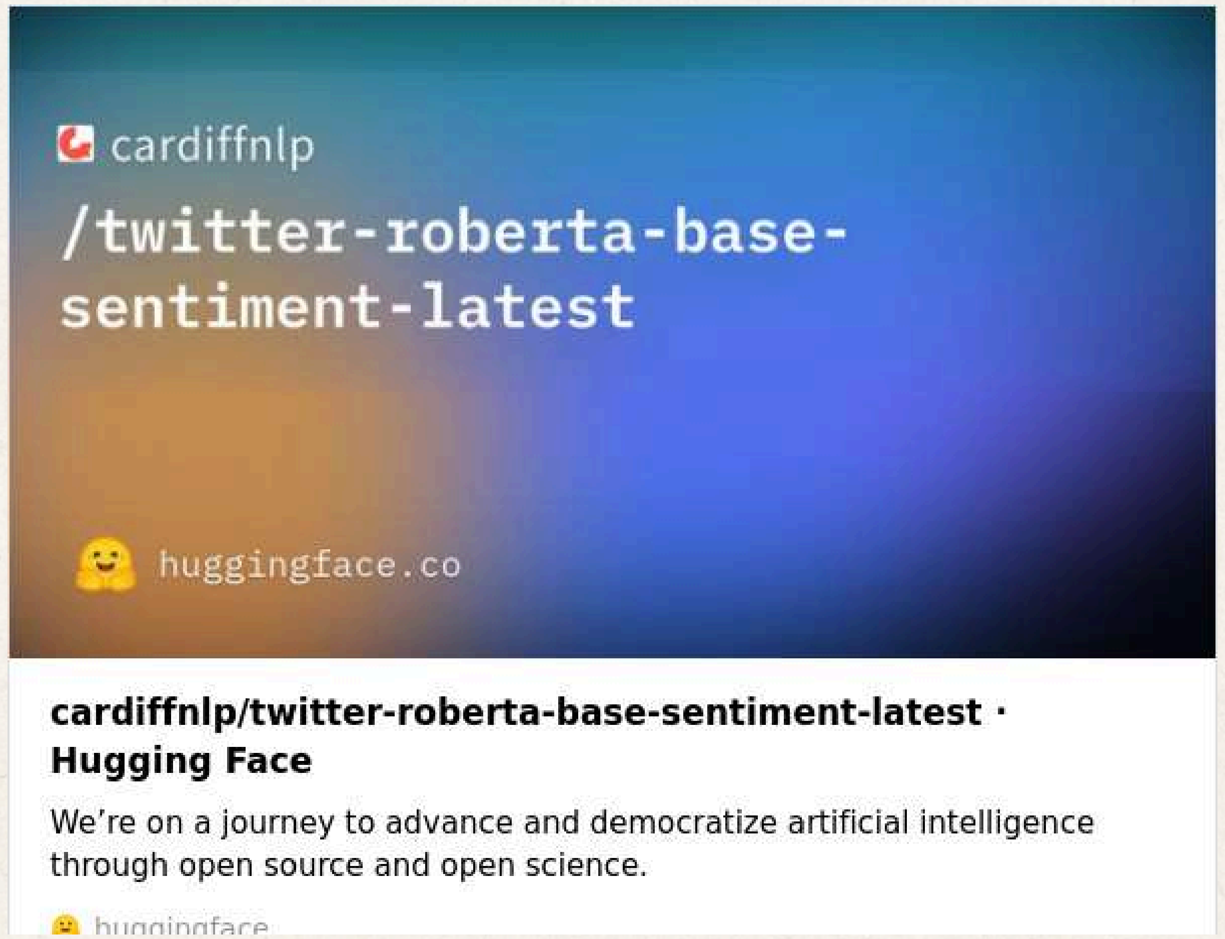
https://chartmasters.org/most-successful-artists-by-decade/

https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest

https://lyricsgenius.readthedocs.io/en/master/

https://genius.com/api-clients

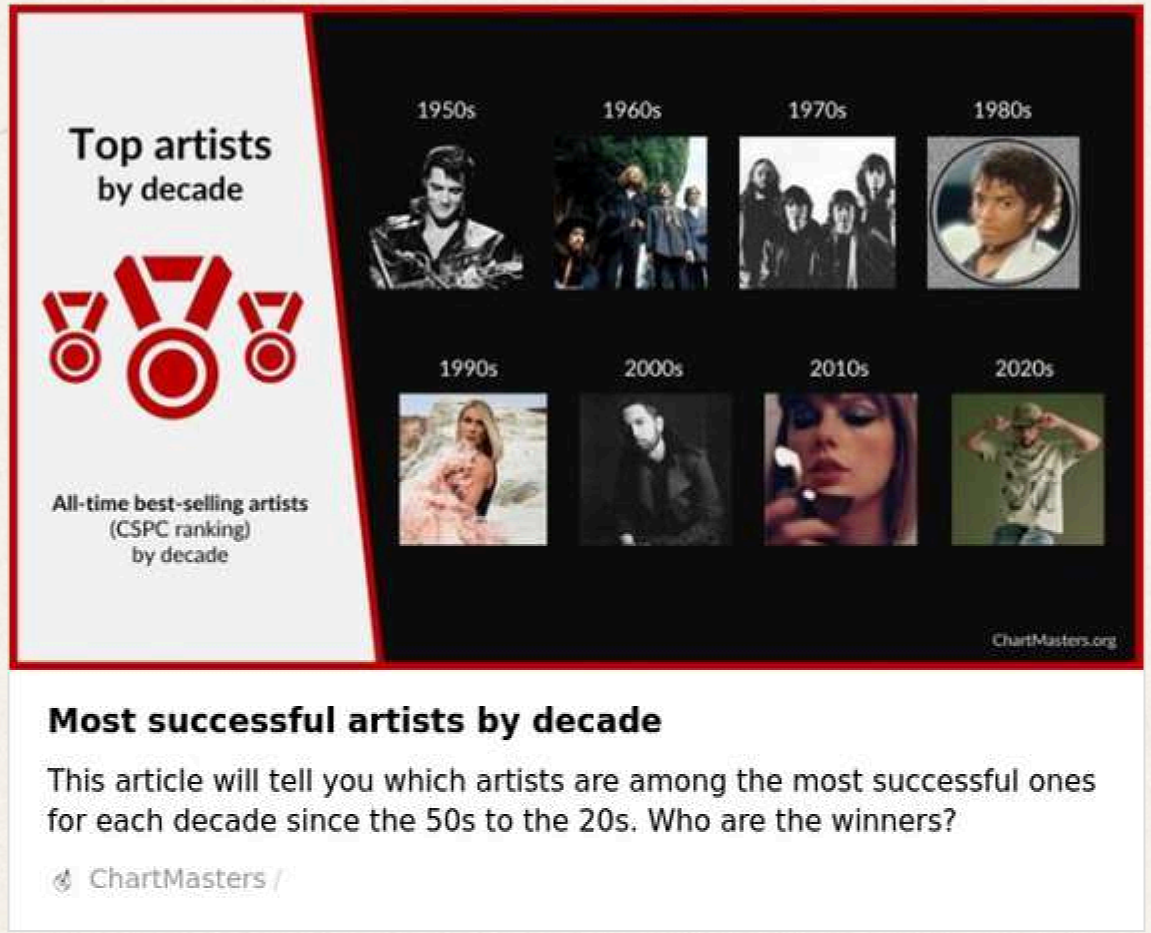
https://medium.com/@cd_24/lyrics-analysis-with-nlp-techniques-4-sentiment-analysis-on-albums-88363eac33fb



cardiffnlp/twitter-roberta-base-sentiment-latest · Hugging Face

We're on a journey to advance and democratize artificial intelligence through open source and open science.

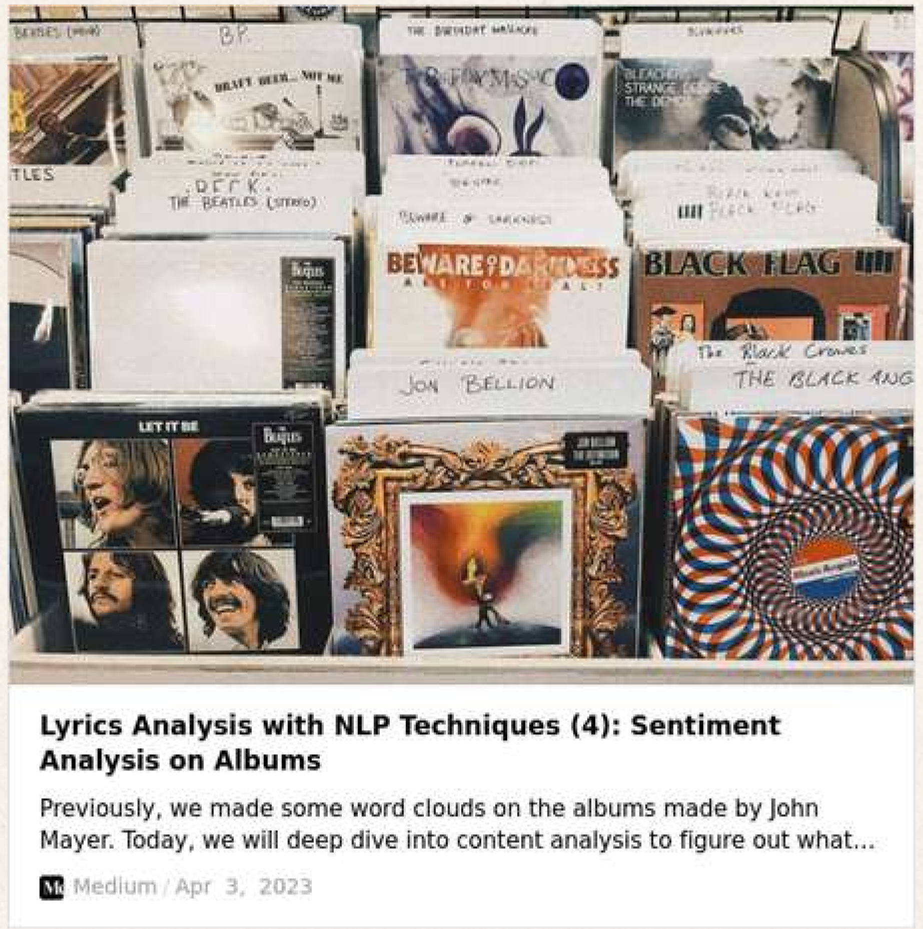
huggingface



Most successful artists by decade

This article will tell you which artists are among the most successful ones for each decade since the 50s to the 20s. Who are the winners?

ChartMasters /



Lyrics Analysis with NLP Techniques (4): Sentiment Analysis on Albums

Previously, we made some word clouds on the albums made by John Mayer. Today, we will deep dive into content analysis to figure out what...

Medium / Apr 3, 2023