

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 2GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

/kaggle/input/cristiano-ronaldo-motivational-speech/Cristiano Ronaldo Motivational Speech.mp4
/kaggle/input/ronaldos-tweets-insight-engagement-and-trends/cr_tweets.csv
/kaggle/input/iltv-news-flash-war-day-146-february-29-2024/ILT News Flash - War Day 146 February 29 2024.mp4
/kaggle/input/food-review-web-scraped-data/google.csv

+ Code + Markdown

```
[ ]:
import pandas as pd
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from gensim import corpora
from gensim.models import LdaModel
from gensim.models import CoherenceModel
import matplotlib.pyplot as plt
```

[9]:

```
df = pd.read_csv("/kaggle/input/food-review-web-scraped-data/google.csv")
df.head()
```

	IDY1rd src	TSubDb	RvU3D href	A503be	dehysf	dmZlBb	PV7e7	review-full-text	k8MTF	...	k8MTF 42	k8MTF 43	k8MTF 46	k8MTF 47	k8MTF 48	k8MTF 49	k8MTF 51	QW0dijf 19	pi6uoE 17	d6SClc 9
googleusercontent.com/a-/ALV-UjVd7...	Ahad Garrix	https://www.google.com/local/review/rap/report...	2 reviews	a day ago	New	Delivery		Ridiculous services of Dominos following up fo...	: 1/5 ...		: 1/5	Service	Atmosphere	: 1/5	Food	Service	0 months ago	5	Hi, you are a big part of our journey and we d...	
googleusercontent.com/a-/ALV-UjXhG...	Ghanshyam Parmar	https://www.google.com/local/review/rap/report...	17 reviews	6 months ago	NaN	Delivery		Why does it mention about the 30 MINUTES DELI...	: 4/5 ...		: 1/5	Service	Atmosphere		Food	: 1/5 Service	0 months ago	7	Hi Navin, our guest is everything to us! We va...	
googleusercontent.com/a-/ALV-UjVNr...	Usernumber 255	https://www.google.com/local/review/rap/report...	1 review	9 months ago	NaN	Delivery		They cheated. It's a fraud. I ordered a pizza...	: 1/5 ...		NaN	NaN	NaN	NaN	NaN	NaN	2	NaN	NaN	
gs://lh3.googleusercontent.com/a-/ALV-UjUm6...	Siddhesh Dingankar	https://www.google.com/local/review/rap/report...	27 reviews	6 years ago	NaN	Delivery		Had great fun at my friend home with Domino's ...	: 5/5 ...		NaN	Atmosphere		NaN	NaN	NaN	NaN	0	NaN	NaN
googleusercontent.com/a-/ACg8ocjRE2...	Deepti Kumar	https://www.google.com/local/review/rap/report...	2 reviews	a year ago	NaN	Take out		1) great outlet(s)2) great menu3) self service4...	: 4/5 ...		NaN	NaN	NaN	NaN	NaN	NaN	3	NaN	NaN	

[10]:

```
# List of columns to remove
# columns_to_remove = ['WBnJf href', 'IDY1rd src', 'TSubDb', 'RvU3D href', 'A503be', 'dehysf', 'dmZlBb', 'PV7e7', 'k8MTF', 'k8MTF 42', 'k8MTF 43', 'k8MTF 46', 'k8MTF 47', 'k8MTF 48', 'k8MTF 49', 'k8MTF 51']

# Drop specified columns
# df = df.drop(columns=columns_to_remove)
df = df[["review-full-text"]]

df.head()
```

	review-full-text
0	Ridiculous services of Dominos following up fo...
1	Why does it mention about the 30 MINUTES DELI...
2	They cheated. It's a fraud. I ordered a pizza...
3	Had great fun at my friend home with Domino's ...
4	1) great outlet(s)2) great menu3) self service4...

[11]:

```
# df = df.rename(columns={
#     'NBa7we src': 'source',
#     'd4r55': 'reviewer_name',
#     'RfnDt': 'review_rating',
#     'eal oGF': 'review_date'}
```

```
# # 'hCjke': 'review_text',
# # 'rsqaWe': 'review_likes',
# # 'wiI7pd': 'response_text',
# # 'w8nwRe': 'response_date',
# # 'dS1Jg': 'response_likes',
# # 'znYl0': 'response_dislikes',
# # 'dS1Jg 2': 'response2_likes',
# # 'znYl0 2': 'response2_dislikes',
# # 'nM6z2c': 'response2_text',
# # 'DZSIDd': 'response2_date',
# # 'wiI7pd 2': 'response2_date',
# # 'W8gobe': 'response2_text'
# })
```

[13]: df.dropna(inplace=True) # Drop rows with missing values

```
from textblob import TextBlob

def get_sentiment(text):
    analysis = TextBlob(text)
    sentiment_score = analysis.sentiment.polarity
    sentiment_category = "positive" if sentiment_score > 0 else "negative" if sentiment_score < 0 else "neutral"
    return sentiment_score, sentiment_category

df[["sentiment_score"], df[["sentiment_category"]]] = zip(*df[["review-full-text"].apply(get_sentiment)])
df.head()
```

[14]:

	review-full-text	sentiment_score	sentiment_category
0	Ridiculous services of Dominos following up fo...	-0.119048	negative
1	Why does it mention about the 30 MINUTES DELI...	-0.087500	negative
2	They cheated . It's a fraud. I ordered a pizza...	-0.300000	negative
3	Had great fun at my friend home with Domino's ...	0.481250	positive
4	1) great outlet/s2) great menu3) self service4...	0.276190	positive

[15]:

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
import numpy as np

# Combine all the text in the 'text' column
all_text = df['review-full-text'].tolist()

# Vectorize the text using CountVectorizer
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(all_text)

# Perform Latent Dirichlet Allocation
num_topics = 5 # You can adjust the number of topics based on your requirements
lda = LatentDirichletAllocation(n_components=num_topics, random_state=42)
lda.fit(X)

# Function to get the top words for each topic
def get_top_words(model, feature_names, n_words=10):
    topics = []
    for topic_idx, topic in enumerate(model.components_):
        top_words = [feature_names[i] for i in topic.argsort()[:-n_words - 1:-1]]
        topics.append(top_words)
    return topics

# Get the feature names from the vectorizer
feature_names = vectorizer.get_feature_names_out()

# Get the top words for each topic
topics = get_top_words(lda, feature_names)

# Print the topics
for i, topic in enumerate(topics):
    print(f"Topic {i + 1}: {', '.join(topic)}")

# Assign topics to a new column 'topics_LDA'
df[['topics_LDA']] = [topics[np.argmax(doc)] for doc in lda.transform(X)]

# Print the DataFrame with the new 'topics_LDA' column
print(df[['review-full-text', 'topics_LDA']])
```

Topic 1: send, cold, realised, arrived, showed, cheated, review, took, late, arrives
Topic 2: order, store, dominos, mentioned, calls, services, following, guy, did, customer
Topic 3: time, delivery, service, ll, coming, ordered, pretty, period, number, minutes
Topic 4: restrictions, wheelchair, everrrr, options, dietary, saddest, kid, parking, vegetarian, accessibility
Topic 5: great, pizza, domino, home, love, comparison, fun, paneer, moment, margarita
Topic 6: great, pizza, domino, home, love, comparison, fun, paneer, moment, margarita
Topic 7: review, took, late, arrives, review-full-text
0 Ridiculous services of Dominos following up fo...
1 Why does it mention about the 30 MINUTES DELI...
2 They cheated . It's a fraud. I ordered a pizza...
3 Had great fun at my friend home with Domino's ...
4 1) great outlet/s2) great menu3) self service4...
5 Saddest Pizza everrrr!!Vegetarian options: ...
6 Saddest Pizza everrrr!!Vegetarian options: ...
7 review, took, late, arrives, review-full-text

[16]: print("Topic_LDA for Row 3:", df.loc[2, 'topics_LDA'])

Topic_LDA for Row 3: ['send', 'cold', 'realised', 'arrived', 'showed', 'cheated', 'review', 'took', 'late', 'arrives']

```
[2]:
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from nltk import FreqDist
from nltk import bigrams
import seaborn as sns
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import nltk
import re

# Download NLTK resources
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('vader_lexicon')

/opt/conda/lib/python3.10/site-packages/nltk/twitter/_init_.py:20: UserWarning: The twython library has not been installed. Some functionality from the twitter package will not be available.
warnings.warn("The twython library has not been installed")
[nltk_data] Downloading package stopwords to /usr/share/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /usr/share/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package vader_lexicon to
[nltk_data]   /usr/share/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
[2]: True
```

```
[4]:
# Load the CSV file
file_path = "/kaggle/input/ronaldos-tweets-insight-engagement-and-trends/cr_tweets.csv"
df = pd.read_csv(file_path)

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 794 entries, 0 to 793
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   id          794 non-null    int64  
 1   createdAt   794 non-null    object 
 2   text        794 non-null    object 
 3   retweetCount 794 non-null    int64  
 4   replyCount  794 non-null    int64  
 5   likeCount   794 non-null    int64  
 6   quoteCount  794 non-null    int64  
 7   bookmarkCount 794 non-null    int64  
dtypes: int64(6), object(2)
memory usage: 49.8+ KB
```

```
[5]: df.head()
```

	id	createdAt	text	retweetCount	replyCount	likeCount	quoteCount	bookmarkCount
0	1750522838283649232	Thu Jan 25 14:15:42 +0000 2024	Had an amazing time with @Binance, creating th...	5657	3676	50466	141	186
1	174070733958825311	Fri Dec 29 12:12:25 +0000 2023	With the best fireworks show in the world - me...	9365	8766	99551	310	364
2	1736824246070890705	Mon Dec 18 19:02:23 +0000 2023	Great memory with my SIXPAD family from the Co...	11088	6263	130247	374	753
3	1736404837649023475	Sun Dec 17 15:15:48 +0000 2023	Grateful to be honored as the most searched at...	21526	7213	194912	1372	1186
4	1732473555923644630	Wed Dec 06 18:54:18 +0000 2023	Funchal is the capital and tourist centre of M...	14517	6903	165795	404	766

```
[6]:
# Remove specified columns
columns_to_remove = ['id', 'createdAt', 'retweetCount', 'replyCount', 'likeCount', 'quoteCount', 'bookmarkCount']
df = df.drop(columns_to_remove, axis=1)
```

```
[7]: df.tail()
```

	text
789	Catch my CR7 Fragance now 🇵🇹 https://t.co/NhOav...
790	Dupla vencedora 🎉 https://t.co/G1suQwIX7j
791	Comment 🍀 if you're planning on coming to a @P...
792	👉 https://t.co/VbcYZBV6AO
793	👉 🎉 https://t.co/2XbtARP6n0

```
[8]:
print("Null Values Before Cleaning:", df.isnull().sum())
df['text'] = df['text'].str.lower()
df = df.dropna()
```

```
Null Values Before Cleaning: text    0
dtype: int64
```

```
[9]: df.head()
```

	text
0	had an amazing time with @binance, creating th...
1	with the best fireworks show in the world - me...
2	great memory with my sixpad family from the co...
3	grateful to be honored as the most searched at...
4	funchal is the capital and tourist centre of m...

```
[10]:
# Remove stop words and special characters
stop_words = set(stopwords.words('english') + ['https', 'de', 'e', 'um'])
df['text'] = df['text'].apply(lambda x: ' '.join([word for word in word_tokenize(x) if word.lower() not in stop_words]))
df['text'] = df['text'].apply(lambda x: re.sub(r'^\w+', '', x))
```

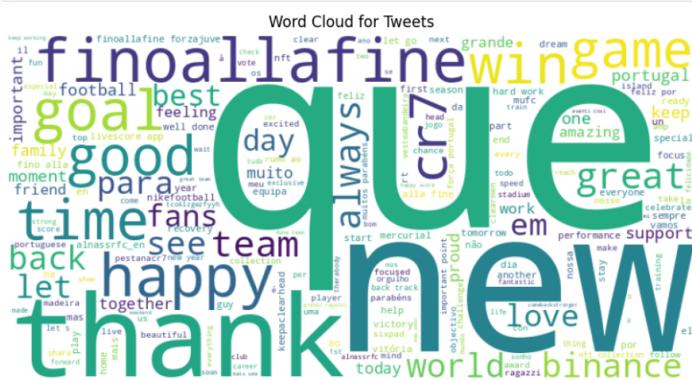
```
[11]: # Sentiment analysis using NLTK's Vader  
sid = SentimentIntensityAnalyzer()  
df['compound'] = df['text'].apply(lambda x: sid.polarity_scores(x)['compound'])  
df['sentiment_label'] = df['compound'].apply(lambda x: 'positive' if x > 0 else 'negative' if x < 0 else 'neutral')  
  
df.head()
```

		text	compound	sentiment_label
0	amazing time binance creating next level fan...	0.8360	positive	
1	best fireworks show world mentioned guinness ...	0.7717	positive	
2	great memory sixpad family core belt event jap...	0.6249	positive	
3	grateful honored searched athlete google hist...	0.7783	positive	
4	funchal capital tourist centre madeira vibran...	0.7430	positive	

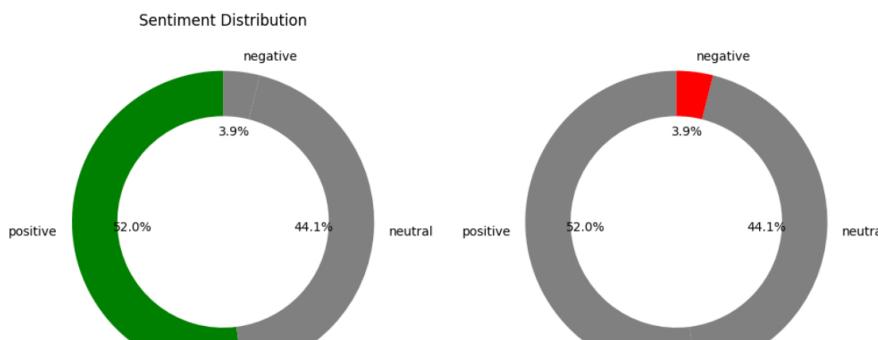
```
[12]: df['sentiment_label'].unique()
```

```
[12]: array(['positive', 'neutral', 'negative'], dtype=object)
```

```
[13]:  
from wordcloud import WordCloud  
import matplotlib.pyplot as plt  
  
# Combine all the text in the 'text' column  
all_text = ' '.join(df['text'])  
  
# Generate a word cloud  
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_text)  
  
# Plot the WordCloud image  
plt.figure(figsize=(10, 5))  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis('off')  
plt.title('Word Cloud for Tweets')  
plt.show()
```



```
[14]:  
sentiment_counts = df['sentiment_label'].value_counts()  
  
# Create a figure and two subplots  
fig, axs = plt.subplots(1, 2, figsize=(12, 6))  
  
# Plot the positive sentiment pie chart  
axs[0].pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%', startangle=90, colors=['green', 'gray', 'gray'], wedgeprops=dict(width=0.3))  
axs[0].set_title('Sentiment Distribution')  
  
# Plot the negative sentiment pie chart  
axs[1].pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%', startangle=90, colors=['gray', 'gray', 'red'], wedgeprops=dict(width=0.3))  
  
# Set the aspect ratio to be equal to ensure the pie charts are circular  
axs[0].set_aspect('equal')  
axs[1].set_aspect('equal')  
  
# Hide the y-axis on the second subplot  
axs[1].set_yticks([])  
  
plt.show()
```



```
[15]:
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation

# Combine all the text in the 'text' column
all_text = df['text'].tolist()

# Vectorize the text using CountVectorizer
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(all_text)

# Perform Latent Dirichlet Allocation
num_topics = 5 # You can adjust the number of topics based on your requirements
lda = LatentDirichletAllocation(n_components=num_topics, random_state=42)
lda.fit(X)

# Function to get the top words for each topic
def get_top_words(model, feature_names, n_words=10):
    topics = []
    for topic_idx, topic in enumerate(model.components_):
        top_words = [feature_names[i] for i in topic.argsort()[:-n_words - 1:-1]]
        topics.append(top_words)
    return topics

# Get the feature names from the vectorizer
feature_names = vectorizer.get_feature_names_out()

# Get the top words for each topic
topics = get_top_words(lda, feature_names)

# Print the topics
for i, topic in enumerate(topics):
    print(f"Topic {i + 1}: {', '.join(topic)}")

# Assign topics to a new column 'topics_LDA'
df['topics_LDA'] = [topics[np.argmax(doc)] for doc in lda.transform(X)]

# Print the DataFrame with the new 'topics_LDA' column
print(df[['text', 'topics_LDA']])


Topic 1: new, mercurial, nikefootball, best, good, dreams, year, love, island, cr7
Topic 2: work, hard, let, game, great, cr7, binance, goals, working, team
Topic 3: que, portugal, mais, para, por, uma, livescore, ao, com, sempre
Topic 4: team, good, happy, support, let, amazing, que, fans, feeling, win
Topic 5: finoallafine, forzajuve, important, team, vittoria, great, ragazzi, avanti, victory, let
text \
0   amazing time binance creating next level fan...
1   best fireworks show world mentioned guinness ...
2   great memory sixpad family core belt event jap...
3   grateful honored searched athlete google hist...
4   funchal capital tourist centre madeira vibran...
...
789  catch cr7 fragance tconhavokinc cr7 crist...
790  dupla vencedora tcogisunquixj
791  comment if planning coming pestanac7 hotel ...
792  tcovbcybv6ao
793  tco2xbtarp6n0

      topics_LDA
0  [team, good, happy, support, let, amazing, que...
1  [new, mercurial, nikefootball, best, good, dre...
2  [team, good, happy, support, let, amazing, que...
3  [team, good, happy, support, let, amazing, que...
4  [finoallafine, forzajuve, important, team, vit...
...
789  [team, good, happy, support, let, amazing, que...
790  [finoallafine, forzajuve, important, team, vit...
791  [team, good, happy, support, let, amazing, que...
792  [work, hard, let, game, great, cr7, binance, g...
793  [team, good, happy, support, let, amazing, que...

[794 rows x 2 columns]
```

```
[16]:
df.head(5)

[16]:
```

	text	compound	sentiment_label	topics_LDA
0	amazing time binance creating next level fan...	0.8360	positive	[team, good, happy, support, let, amazing, que...
1	best fireworks show world mentioned guinness ...	0.7717	positive	[new, mercurial, nikefootball, best, good, dre...
2	great memory sixpad family core belt event jap...	0.6249	positive	[team, good, happy, support, let, amazing, que...
3	grateful honored searched athlete google hist...	0.7783	positive	[team, good, happy, support, let, amazing, que...
4	funchal capital tourist centre madeira vibran...	0.7430	positive	[finoallafine, forzajuve, important, team, vit...

```
[17]:
# Print the 'topics_LDA' column for Row 3
print("Topic_LDA for Row 3:", df.loc[2, 'topics_LDA'])

Topic_LDA for Row 3: ['team', 'good', 'happy', 'support', 'let', 'amazing', 'que', 'fans', 'feeling', 'win']
```

Audio Processing : Librosa

```
[60]:
pip install librosa pydub SpeechRecognition

Requirement already satisfied: librosa in /opt/conda/lib/python3.10/site-packages (0.10.1)
Requirement already satisfied: pydub in /opt/conda/lib/python3.10/site-packages (0.25.1)
Requirement already satisfied: SpeechRecognition in /opt/conda/lib/python3.10/site-packages (3.10.1)
Requirement already satisfied: audioread>=2.1.9 in /opt/conda/lib/python3.10/site-packages (from librosa) (3.0.1)
Requirement already satisfied: numpy>=1.22.0,>=1.22.1,>=1.22.2,>=1.20.3 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.24.4)
Requirement already satisfied: scipy>=1.2.0 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.11.4)
Requirement already satisfied: scikit-learn>=0.20.0 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.2.2)
Requirement already satisfied: joblib>=0.14 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.3.2)
Requirement already satisfied: decorator>=4.3.0 in /opt/conda/lib/python3.10/site-packages (from librosa) (4.4.2)
```

```

requirement already satisfied: numpy>=1.19 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.20.1)
Requirement already satisfied: soundfile>=0.12.1 in /opt/conda/lib/python3.10/site-packages (from librosa) (0.12.1)
Requirement already satisfied: pooch>=1.0 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.8.0)
Requirement already satisfied: sox>=0.3.2 in /opt/conda/lib/python3.10/site-packages (from librosa) (0.3.7)
Requirement already satisfied: typing-extensions>=4.1.1 in /opt/conda/lib/python3.10/site-packages (from librosa) (4.9.0)
Requirement already satisfied: lazy-loader>=0.1 in /opt/conda/lib/python3.10/site-packages (from librosa) (0.3)
Requirement already satisfied: msgpack>=1.0 in /opt/conda/lib/python3.10/site-packages (from librosa) (1.0.7)
Requirement already satisfied: requests>=2.26.1 in /opt/conda/lib/python3.10/site-packages (from SpeechRecognition) (2.31.0)
Requirement already satisfied: llvmlite<0.42,>=0.41.0dev0 in /opt/conda/lib/python3.10/site-packages (from numba>=0.51.0->librosa) (0.41.1)
Requirement already satisfied: platformdirs>=2.5.0 in /opt/conda/lib/python3.10/site-packages (from pooch>=1.0->librosa) (4.1.0)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.10/site-packages (from pooch>=1.0->librosa) (21.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/python3.10/site-packages (from requests>=2.26.0->SpeechRecognition) (3.3.2)
Requirement already satisfied: idna>=2.5 in /opt/conda/lib/python3.10/site-packages (from requests>=2.26.0->SpeechRecognition) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.10/site-packages (from requests>=2.26.0->SpeechRecognition) (1.26.18)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests>=2.26.0->SpeechRecognition) (2023.11.17)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/lib/python3.10/site-packages (from scikit-learn>=0.20.0->librosa) (3.2.0)
Requirement already satisfied: cffi>=1.0 in /opt/conda/lib/python3.10/site-packages (from soundfile>=0.12.1->librosa) (1.16.0)
Requirement already satisfied: pyparsing in /opt/conda/lib/python3.10/site-packages (from packaging>=20.0->pooch>=1.0->librosa) (2.2.1)
Requirement already satisfied: pyParsing>=3.0.5,>=2.0.2 in /opt/conda/lib/python3.10/site-packages (from packaging>=20.0->pooch>=1.0->librosa) (3.1.1)
Note: you may need to restart the kernel to use updated packages.

```

```
[72]:
import librosa
import librosa.display
import speech_recognition as sr
from pydub import AudioSegment
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
import urllib.request

audio_url = "https://github.com/AtharvaPawar456/SEM-8-Comps-FRCRCE/raw/main/Social-Media-Analytics-SMA/Labs/exp%20-%205/Cristiano%20Ronaldo%20Motivational%20Speech%20CR7%20bes"
# audio_url = "https://github.com/AtharvaPawar456/SEM-8-Comps-FRCRCE/blob/main/Social-Media-Analytics-SMA/Labs/exp%20-%205/ILTV%20News%20Flash%20-%20War%20day%20146%20February%202024.mp4"
output_file = "audio.mp4"

# Download the file
urllib.request.urlretrieve(audio_url, output_file)

print("Audio file downloaded successfully.")


```

Audio file downloaded successfully.

```
[74]:
# Load the audio file using librosa
audio_file_path = "/kaggle/input/iltv-news-flash-war-day-146-february-29-2024/ILTV News Flash - War Day 146 February 29 2024.mp4"
audio, sr = librosa.load(audio_file_path, sr=None)
```

```
[75]:
audio, sr

[75]: (array([0.          , 0.          , 0.          , ..., 0.00094604, 0.00085449,
       44100]),
```

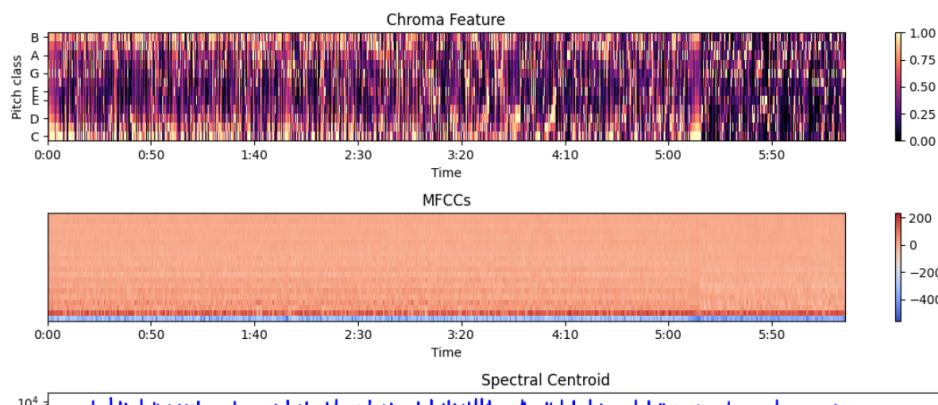
```
[76]:
# Audio Feature Extraction
chroma_stft = librosa.feature.chroma_stft(y=audio, sr=sr)
mfccs = librosa.feature.mfcc(y=audio, sr=sr)
spectral_centroid = librosa.feature.spectral_centroid(y=audio, sr=sr)[0]
spectral_rolloff = librosa.feature.spectral_rolloff(y=audio, sr=sr)[0]
```

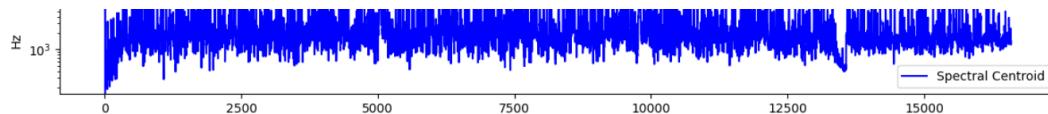
```
[77]:
# Plot Chroma Feature
plt.figure(figsize=(12, 6))
plt.subplot(3, 1, 1)
librosa.display.specshow(chroma_stft, y_axis='chroma', x_axis='time')
plt.title('Chroma Feature')
plt.colorbar()

# Plot MFCCs
plt.subplot(3, 1, 2)
librosa.display.specshow(mfccs, x_axis='time')
plt.title('MFCCs')
plt.colorbar()

# Plot Spectral Centroid
plt.subplot(3, 1, 3)
plt.semilogy(spectral_centroid, label='Spectral Centroid', color='b')
plt.ylabel('Hz')
plt.title('Spectral Centroid')
plt.legend()

plt.tight_layout()
plt.show()
```





```
[78]: # Beat and Tempo Detection
tempo, beat_frames = librosa.beat.beat_track(y=audio, sr=sr)
```

```
[79]: import numpy as np

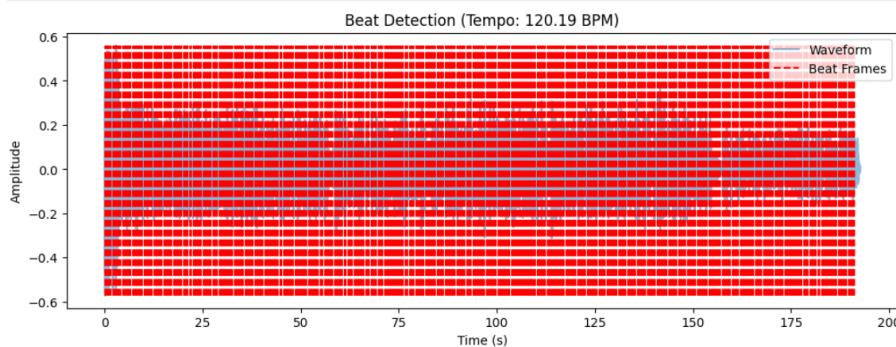
# Plot Beat and Tempo
beat_times = librosa.frames_to_time(beat_frames, sr=sr)

plt.figure(figsize=(12, 4))

# Plot the waveform
plt.plot(np.arange(len(audio)) / sr, audio, label='Waveform', alpha=0.5)

# Plot the beat Frames
plt.vlines(beat_times, ymin=min(audio), ymax=max(audio), color='r', linestyle='dashed', label='Beat Frames')

plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.title(f'Beat Detection (Tempo: {tempo:.2f} BPM)')
plt.legend()
plt.show()
```



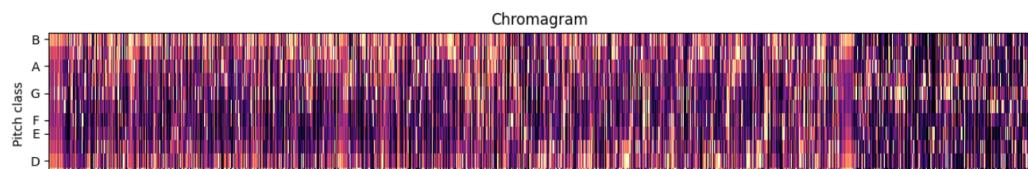
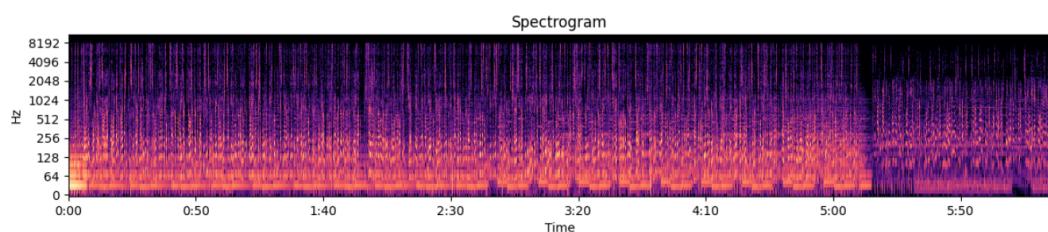
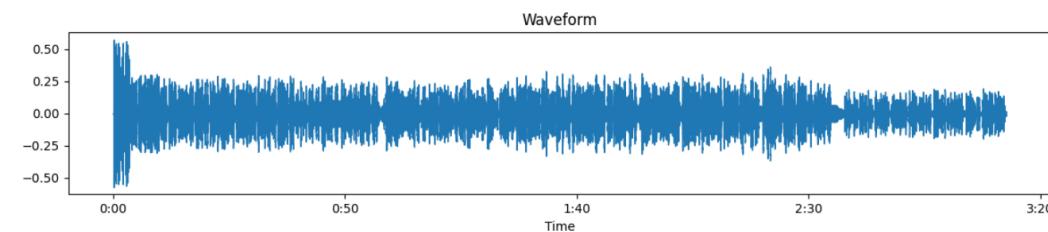
```
[80]: # Music Visualization
plt.figure(figsize=(12, 8))

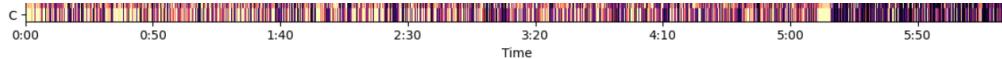
plt.subplot(3, 1, 1)
librosa.display.waveshow(audio, sr=sr)
plt.title('Waveform')

plt.subplot(3, 1, 2)
librosa.display.specshow(librosa.amplitude_to_db(np.abs(librosa.stft(audio)), ref=np.max), y_axis='log', x_axis='time')
plt.title('Spectrogram')

plt.subplot(3, 1, 3)
librosa.display.specshow(chroma_stft, y_axis='chroma', x_axis='time')
plt.title('Chromagram')

plt.tight_layout()
plt.show()
```





```
[81]: pip install moviepy

Requirement already satisfied: moviepy in /opt/conda/lib/python3.10/site-packages (1.0.3)
Requirement already satisfied: decorator<5.0,>=4.0.2 in /opt/conda/lib/python3.10/site-packages (from moviepy) (4.4.2)
Requirement already satisfied: tqdm<5.0,>=4.11.2 in /opt/conda/lib/python3.10/site-packages (from moviepy) (4.66.1)
Requirement already satisfied: requests<3.0,>=2.8.1 in /opt/conda/lib/python3.10/site-packages (from moviepy) (2.31.0)
Requirement already satisfied: prolog<1.0.0 in /opt/conda/lib/python3.10/site-packages (from moviepy) (0.1.18)
Requirement already satisfied: numpy<>1.17.3 in /opt/conda/lib/python3.10/site-packages (from moviepy) (1.24.4)
Requirement already satisfied: imageio<3.0,>=2.5 in /opt/conda/lib/python3.10/site-packages (from moviepy) (2.33.1)
Requirement already satisfied: imageio-ffmpeg<0.2.0 in /opt/conda/lib/python3.10/site-packages (from moviepy) (0.4.9)
Requirement already satisfied: pillow<9.3.2 in /opt/conda/lib/python3.10/site-packages (from imageio-ffmpeg=>0.2.0->moviepy) (9.5.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>=2.8.1->moviepy) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>=2.8.1->moviepy) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>=2.8.1->moviepy) (1.26.18)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>=2.8.1->moviepy) (2023.11.17)
Note: you may need to restart the kernel to use updated packages.
```

```
[82]: from pydub import AudioSegment
import speech_recognition as sr
import os

def convert_to_wav(input_file, output_file):
    audio = AudioSegment.from_file(input_file, format="mp4")
    audio.export(output_file, format="wav")

def split_audio(input_file, output_directory, chunk_size=7000):
    audio = AudioSegment.from_wav(input_file)
    num_chunks = len(audio) // chunk_size + 1

    for i in range(num_chunks):
        start_time = i * chunk_size
        end_time = (i + 1) * chunk_size
        chunk = audio[start_time:end_time]
        chunk.export(os.path.join(output_directory, f"chunk_{i+1}.wav"), format="wav")

def recognize_text_from_audio(input_file):
    recognizer = sr.Recognizer()

    with sr.AudioFile(input_file) as source:
        audio_data = recognizer.record(source)
        try:
            text = recognizer.recognize_google(audio_data)
            return text
        except sr.UnknownValueError:
            return "Speech Recognition could not understand audio"
        except sr.RequestError as e:
            return f"Could not request results from Google Speech Recognition service; {e}"

# Convert to WAV
audio_file_path_wav = "/kaggle/working/audio.wav"
convert_to_wav(audio_file_path, audio_file_path_wav)

# Split into chunks
output_directory = "/kaggle/working/chunks"
os.makedirs(output_directory, exist_ok=True)
split_audio(audio_file_path_wav, output_directory)

final_text = ""
final_text_list = []

# Recognize text from each chunk
for i, chunk_file in enumerate(os.listdir(output_directory)):
    chunk_path = os.path.join(output_directory, chunk_file)
    text = recognize_text_from_audio(chunk_path)
    print(f"Text from Chunk {i + 1}:\n{text}\n")
    final_text += " " + text
    final_text_list.append(text)
```

Text from Chunk 1:
I looked to myself and I said what you can do it better to improve your level to up the team yet

Text from Chunk 2:
spell mediators have proposed to free Israeli hostages in exchange for an extended poisoning fighting the US

Text from Chunk 3:
we have Saturday a big game that we have to drive or just

Text from Chunk 4:
boss is still trying to keep firing but each time the launch site is quickly attacked by the Army troops and communist

Text from Chunk 5:
to Jerusalem the security cabinet is meeting today in the efforts to advance

Text from Chunk 6:
next on Israel in a political development defense minister Gallant dropped a bombshell with a call for an end

Text from Chunk 7:
list of Gaza struck multiple targets after Rockets were fired at Escalon hamas's ability to fire Rockets

Text from Chunk 8:
Larry Burton his comment was quickly met by threats from haredi politicians to bring down the government Gallant said he would only

Text from Chunk 9:
operations targeting Tara tunnels and infrastructure the 7th Brigade rated buildings and recovered weapons in military

Text from Chunk 10:
play bluegrass Hamas Chief is Melania said that the terror group is showing flexibility in the talks but is also prepared

Text from Chunk 11:
and I'm ready when you suffer when you struggle in a game you know to be compacts

Text from Chunk 12:
what's the proposal reportedly provides for a 6-week poison fighting during which some 40 hostages would be freed in exchange

Text from Chunk 13:
day 146 of the war IDF forces in the

Text from Chunk 14:
reach for ceasefire saying that in the coming days we will reach a ceasefire and there will be relief real relief for the

Text from Chunk 15:
Speech Recognition could not understand audio

Text from Chunk 16:
the United Arab Emirates and the United States to airdrop various humanitarian supplies to guys and residents

Text from Chunk 17:
play some 400 Palestinian security prisoners Egyptian president LCC expressed hope that there may soon be

Text from Chunk 18:
how many gallons position along with growing pressure on the government to reach a resolution on the long deferred political hot

Text from Chunk 19:
the coordinated effort by the state's or supplies such as food and medical equipment airdrop to residents of Southern Gaza and

Text from Chunk 20:
Speech Recognition could not understand audio

Text from Chunk 21:
Jordanian field Hospital the families of Israeli hostages joined by a growing number of supporters

Text from Chunk 22:
military draft exemptions for ultra-orthodox Jews Gallant said that the ultra-orthodox community must share the

Text from Chunk 23:
can I look in the future I'm looking to present the present is now the moment is

Text from Chunk 24:
we embarked on a mission to unveil the real Israel dismantling the Web of Lies and misinformation that Sun

Text from Chunk 25:
...the name of Taylor (11, 12, TV, and 16).

experience the power of truth with II TV News II

continue to fight at the same time on here also called on the so-called axis of resistance to step up

legislation settling the matter if it's endorsed by Centrist ministers then against and Gotti eisenhardt

Text from Chunk 2b: amplify our message creating a ripple effect that carries the truth far and wide subscript

I leave the presents the present for me is the most important in the future we will

Text from Chunk 31:

Text from Chunk 32:

volume 40% and beat him every weekend is a challenge I'm not
Text from Chunk 33:

Text from Chunk 34:

play The Narrative available on the web Android and Apple
Text from Chunk 35:

Text from Chunk 36:

you're looking for quality content and captivating visuals join our news community and be part of the conversation.

Text from Chunk 38:
all the places different the custom the stadium you have to be sensible

Text from Chunk 39:
reporting on Israel by subscribing to ity news you will not only have access to the latest updates but you will also

```
[83]: print("Speech length : ", len(final_text))  
      print("\nSpeech text   :\n",final_text)
```

Speech length : 3730

Speech text :
I looked to myself and I said what you can do it better to improve your level to up the team yet spell mediators have proposed to free Israeli hostages in exchange for an extended poisoning fighting the US we have Saturday a big game that we have to drive or just boss is still trying to keep firing but each time the launch site is quickly attacked by the Army troops and communist to Jerusalem the security cabinet is meeting Saturday in the efforts to advance next on Israel in a political development defense minister Gallant dropped a bombshell with a call for an end list of Gaza struck multiple targets after Rockets were fired at Escalon hamas's ability to fire Rockets Larry Burton his comment was quickly met by threats from haredi politicians to bring down the government Gallant said he would only operations targeting Tara tunnels and in fracture the 7th Brigade rated buildings and recovered weapons in military play bluegrass Hama Chief is Melania said that the terror group is showing flexibility in the talks but is also prepared and I'm ready when you suffer when you struggle in a game you know to be compacts what's the proposal reportedly provides for a 6-week poison fighting during which some 40 hostages would be freed in exchange day 146 of the war IDF forces in the reach for ceasefire saying that in the coming days we will reach a ceasefire and there will be relief real relief for the Speech Recognition could not understand audio the United Arab Emirates and the United States to airdrop various humanitarian supplies to guys and residents play some 400 Palestinian security prisoners Egyptian president LCC expressed hope that there may soon be how many gallons position along with growing pressure on the government to reach a resolution on the long deferred political hot the coordinated effort by the state's or supplies such as food and medical equipment airdrop to residents of Southern Gaza and Speech Recognition could not understand audio Jordanian families the families of Israeli hostages joined by a growing number of supporters military draft exemptions for ultra-orthodox Jews Gallant said that the ultra-orthodox community must share the pain I look in the future I'm looking to present the present is now the moment we are embarked on a mission to unveil the real Israeli dismantling of the ultra-orthodox community the power of the law and if I need if I continued to fight in the same time period on the so-called responsibility to stand up legislation representing the matter if it's endorsed by Minister then again and Gotts eigentum amplify the existing ripple effect that presents the present for me is the most important in the future we will come 90 minutes if something happened to help my teammates when you suffer or he's tired of course and step by step as I mentioned before we're going to improve then by game volume 40% and beat him every weekend is a challenge I'm not could set up a Do or Die clash with our lady partners key to the survival of prime minister Netanyahu play The Narrative available on the web Android and Apple from Gaza has been reduced to a tricke since the start of the war as the idea of gained control of most of Gaza you're looking for Quality content and captivating visuals join our new community and become an integral part of our team meanwhile the IDF announced a joint coordinated effort between Israel Jordan Egypt France all the places different the system the stadiums you have to be capable reporting on Israel by subscribing to iiltv news you will not only have access to the latest updates but you will also

```
[84]: # Generate Word Cloud  
wordcloud = WordCloud(width=800, height=400, random_state=42, max_font_size=100, background_color='white').generate(final_text)  
  
# Display the Word Cloud using matplotlib  
plt.figure(figsize=(10, 6))  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis('off')  
plt.title('Word Cloud for Final Text')  
plt.show()
```



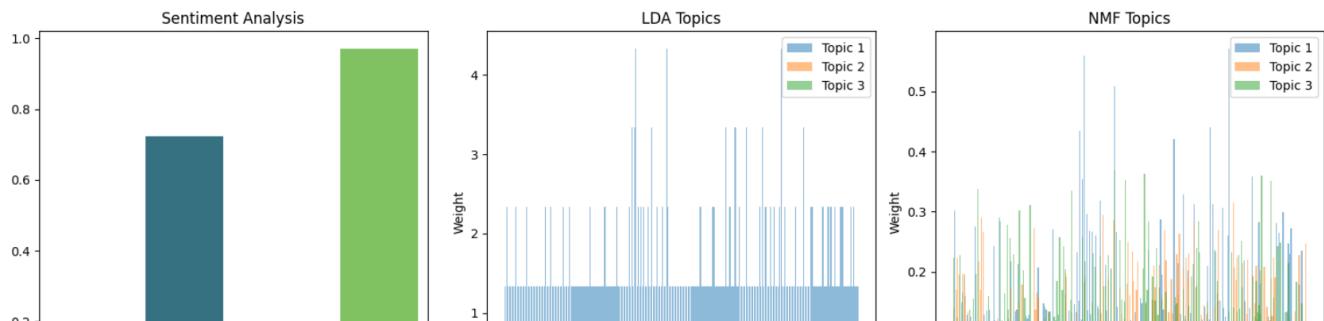
```
[85]:  
import nltk  
from nltk.sentiment.vader import SentimentIntensityAnalyzer  
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer  
from sklearn.decomposition import LatentDirichletAllocation, NMF  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Download NLTK resources  
nltk.download('vader_lexicon')  
  
# Example text for demonstration  
# final_text = "Your final text goes here. It can be a combination of sentences or paragraphs."  
  
# Sentiment Analysis using NLTK's Vader  
sid = SentimentIntensityAnalyzer()  
sentiment_scores = sid.polarity_scores(final_text)  
sentiment_label = 'positive' if sentiment_scores['compound'] > 0 else 'negative' if sentiment_scores['compound'] < 0 else 'neutral'  
  
print("Sentiment Analysis:")  
print(f"Compound Sentiment Score: {sentiment_scores['compound']}")  
print(f"Sentiment Label: {sentiment_label}\n")  
  
[nltk_data] Downloading package vader_lexicon to  
[nltk_data]   /usr/share/nltk_data...  
[nltk_data]   Package vader_lexicon is already up-to-date!  
Sentiment Analysis:  
Compound Sentiment Score: 0.9714  
Sentiment Label: positive
```

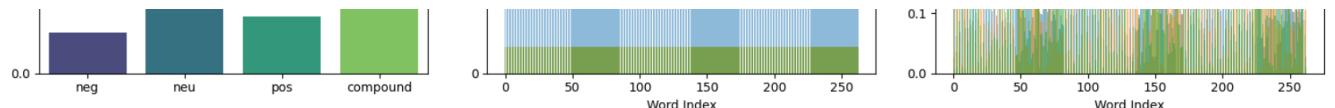
```
[86]:  
# Topic Modeling using Latent Dirichlet Allocation (LDA)  
vectorizer = CountVectorizer(stop_words='english')  
X = vectorizer.fit_transform([final_text])  
  
num_topics_lda = 3 # Adjust the number of topics as needed  
lda = LatentDirichletAllocation(n_components=num_topics_lda, random_state=42)  
lda.fit(X)  
  
[86]:  
LatentDirichletAllocation  
LatentDirichletAllocation(n_components=3, random_state=42)
```

```
[87]:  
# Topic Modeling using Non-Negative Matrix Factorization (NMF)  
vectorizer_tfidf = TfidfVectorizer(stop_words='english')  
X_tfidf = vectorizer_tfidf.fit_transform([final_text])  
  
num_topics_nmf = 3 # Adjust the number of topics as needed  
nmf = NMF(n_components=num_topics_nmf, random_state=42)  
nmf.fit(X_tfidf)
```

```
[87]:  
NMF  
NMF(n_components=3, random_state=42)
```

```
[88]:  
# Convert sentiment scores to a Pandas DataFrame  
sentiment_df = pd.DataFrame(sentiment_scores, index=[0])  
  
# Visualization  
plt.figure(figsize=(15, 5))  
  
# Plot Sentiment Analysis  
plt.subplot(1, 3, 1)  
sns.barplot(data=sentiment_df, palette="viridis")  
plt.title('Sentiment Analysis')  
  
# Plot LDA Topics  
plt.subplot(1, 3, 2)  
for i, topic in enumerate(lda.components_):  
    plt.bar(range(len(vectorizer.get_feature_names_out())), topic, label=f'Topic {i + 1}', alpha=0.5)  
plt.xlabel('Word Index')  
plt.ylabel('Weight')  
plt.title('LDA Topics')  
plt.legend()  
  
# Plot NMF Topics  
plt.subplot(1, 3, 3)  
for i, topic in enumerate(nmf.components_):  
    plt.bar(range(len(vectorizer_tfidf.get_feature_names_out())), topic, label=f'Topic {i + 1}', alpha=0.5)  
plt.xlabel('Word Index')  
plt.ylabel('Weight')  
plt.title('NMF Topics')  
plt.legend()  
  
plt.tight_layout()  
plt.show()
```





[89]: pip install gensim scikit-learn

```
Requirement already satisfied: gensim in /opt/conda/lib/python3.10/site-packages (4.3.2)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.10/site-packages (1.2.2)
Requirement already satisfied: numpy>=1.18.5 in /opt/conda/lib/python3.10/site-packages (from gensim) (1.24.4)
Requirement already satisfied: scipy>=1.7.0 in /opt/conda/lib/python3.10/site-packages (from gensim) (1.11.4)
Requirement already satisfied: smallopopen>=1.8.1 in /opt/conda/lib/python3.10/site-packages (from gensim) (6.4.0)
Requirement already satisfied: joblib>=1.1.1 in /opt/conda/lib/python3.10/site-packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>2.0.0 in /opt/conda/lib/python3.10/site-packages (from scikit-learn) (3.2.0)
Note: you may need to restart the kernel to use updated packages.
```

```
[94]: from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation, NMF
from gensim.corpora.dictionary import Dictionary
from gensim.models.ldamodel import LdaModel
from gensim.models.nmf import Nmf
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import pandas as pd

# Example final_text
# final_text = "Your final text goes here. It can be a combination of sentences or paragraphs."

# Function to extract topics using Latent Dirichlet Allocation (LDA)
def get_lda_topics(text, num_topics=5):
    vectorizer = CountVectorizer(stop_words='english')
    X = vectorizer.fit_transform([text])

    lda = LatentDirichletAllocation(n_components=num_topics, random_state=42)
    lda.fit(X)

    words = vectorizer.get_feature_names_out()
    topics = {}

    for i, topic in enumerate(lda.components_):
        top_words_idx = topic.argsort()[-6:-1] # Get indices of top 5 words for each topic
        top_words = [words[idx] for idx in top_words_idx]
        topics[f"Topic {i + 1}"] = {
            "Top Words": top_words,
            "Score": topic.sum() # You can customize the score calculation as needed
        }

    return topics

# Function to extract topics using Non-Negative Matrix Factorization (NMF)
def get_nmf_topics(text, num_topics=5):
    vectorizer_tfidf = TfidfVectorizer(stop_words='english')
    X_tfidf = vectorizer_tfidf.fit_transform([text])

    nmf = NMF(n_components=num_topics, random_state=42)
    nmf.fit(X_tfidf)

    words = vectorizer_tfidf.get_feature_names_out()
    topics = {}

    for i, topic in enumerate(nmf.components_):
        top_words_idx = topic.argsort()[-6:-1] # Get indices of top 5 words for each topic
        top_words = [words[idx] for idx in top_words_idx]
        topics[f"Topic {i + 1}"] = {
            "Top Words": top_words,
            "Score": topic.sum() # You can customize the score calculation as needed
        }

    return topics

# Get topics using LDA
lda_topics = get_lda_topics(final_text)

# Get topics using NMF
nmf_topics = get_nmf_topics(final_text)

# Combine topics from LDA and NMF
combined_topics = {*lda_topics, **nmf_topics}

# Sort topics by score and print the top 5
top_5_topics = sorted(combined_topics.items(), key=lambda x: x[1]['Score'], reverse=True)[:5]

topic, details = top_5_topics[0]
print(f'{topic}: {details["Top Words"]}, Score: {details["Score"]}\n\n')

# Print top 5 topics
print("Top 5 Topics:")
for topic, details in top_5_topics:
    print(f"\t--- {topic}: {details['Top Words']}, Score: {details['Score']}")
```

Topic 1: ['said', 'gaza', 'israel', 'reach', 'gallant'], Score: 23.84661513481874

```
Top 5 Topics:
--- Topic 1: ['said', 'gaza', 'israel', 'reach', 'gallant'], Score: 23.84661513481874
--- Topic 5: ['course', 'amplify', 'president', 'hospital', 'threats'], Score: 22.22702820767053
--- Topic 3: ['israel', 'long', 'supporters', 'jordanian', 'team'], Score: 22.20540143231875
--- Topic 4: ['freed', 'dismantling', 'quality', 'carries', 'looked'], Score: 21.530376309024646
--- Topic 2: ['security', 'present', 'relief', 'big', 'key'], Score: 20.189008662615237
```

Video Processing

```
[95]: import urllib.request
audio_url = "https://github.com/AtharvaPawar456/SEM-B-Comps-FRCRCE/blob/main/Social-Media-Analytics-SMA/Labs/exp%20-%205/Cristiano%20Ronaldo%20Motivational%20Speech.mp4"
output_file = "video.mp4"

# Download the file
urllib.request.urlretrieve(audio_url, output_file)

print("Audio file downloaded successfully.")

Audio file downloaded successfully.
```

```
[96]: pip install mediapipe opencv-python moviepy

Requirement already satisfied: mediapipe in /opt/conda/lib/python3.10/site-packages (0.10.10)
Requirement already satisfied: opencv-python in /opt/conda/lib/python3.10/site-packages (4.9.0.80)
Requirement already satisfied: moviepy in /opt/conda/lib/python3.10/site-packages (1.0.3)
Requirement already satisfied: absl-py in /opt/conda/lib/python3.10/site-packages (from mediapipe) (1.4.0)
Requirement already satisfied: attrs>=19.1.0 in /opt/conda/lib/python3.10/site-packages (from mediapipe) (23.2.0)
Requirement already satisfied: flatbuffers>2.0 in /opt/conda/lib/python3.10/site-packages (from mediapipe) (23.5.26)
Requirement already satisfied: jax in /opt/conda/lib/python3.10/site-packages (from mediapipe) (0.4.23)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.10/site-packages (from mediapipe) (3.7.4)
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages (from mediapipe) (1.24.4)
Requirement already satisfied: protobuf<4>; python<3.11 in /opt/conda/lib/python3.10/site-packages (from mediapipe) (3.20.3)
Requirement already satisfied: sounddevice>0.4.4 in /opt/conda/lib/python3.10/site-packages (from mediapipe) (0.4.6)
Requirement already satisfied: decorator<5.0,>4.0.2 in /opt/conda/lib/python3.10/site-packages (from moviepy) (4.4.2)
Requirement already satisfied: tqdm<5.0,>4.11.2 in /opt/conda/lib/python3.10/site-packages (from moviepy) (4.66.1)
Requirement already satisfied: requests<3.0,>2.8.1 in /opt/conda/lib/python3.10/site-packages (from moviepy) (2.31.0)
Requirement already satisfied: prolog<1.0.0 in /opt/conda/lib/python3.10/site-packages (from moviepy) (0.1.10)
Requirement already satisfied: imageio<3.0,>2.5 in /opt/conda/lib/python3.10/site-packages (from moviepy) (2.33.1)
Requirement already satisfied: imageio-ffmpeg>0.2.0 in /opt/conda/lib/python3.10/site-packages (from moviepy) (0.4.9)
Requirement already satisfied: pillow<8.3.2,>8.0.0 in /opt/conda/lib/python3.10/site-packages (from imageio<3.0,>2.5->moviepy) (0.5.0)
Requirement already satisfied: asturio<0.3.2,>0.3.1 in /opt/conda/lib/python3.10/site-packages (from imageio-ffmpeg>0.2.0->moviepy) (69.0.3)
Requirement already satisfied: charset-normalizer<4,>2 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>2.8.1->moviepy) (3.3.2)
Requirement already satisfied: idna<4,>2.5 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>2.8.1->moviepy) (3.6)
Requirement already satisfied: urllib3<2.1.1,>1.26.1 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>2.8.1->moviepy) (1.26.18)
Requirement already satisfied: certifi<2017.4.17,>2023.11.17 in /opt/conda/lib/python3.10/site-packages (from requests<3.0,>2.8.1->moviepy) (2023.11.17)
Requirement already satisfied: CFFI>1.0 in /opt/conda/lib/python3.10/site-packages (from sounddevice>0.4.4->mediapipe) (1.16.0)
Requirement already satisfied: m1-dtypes>0.2.0 in /opt/conda/lib/python3.10/site-packages (from jax->mediapipe) (0.2.0)
Requirement already satisfied: opt-einsum in /opt/conda/lib/python3.10/site-packages (from jax->mediapipe) (3.3.0)
Requirement already satisfied: scipy>1.9 in /opt/conda/lib/python3.10/site-packages (from jax->mediapipe) (1.11.4)
Requirement already satisfied: contourpy>1.0.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (1.2.0)
Requirement already satisfied: fonttools>4.22.0 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (4.47.0)
Requirement already satisfied: KdWISolver>1.0.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (1.4.5)
Requirement already satisfied: packaging>20.0 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (21.3)
Requirement already satisfied: pyParsing>2.3.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (3.11)
Requirement already satisfied: python-dateutil>2.7 in /opt/conda/lib/python3.10/site-packages (from matplotlib->mediapipe) (2.8.2)
Requirement already satisfied: pycparser in /opt/conda/lib/python3.10/site-packages (from CFFI>1.0->sounddevice>0.4.4->mediapipe) (2.21)
Requirement already satisfied: six>1.5 in /opt/conda/lib/python3.10/site-packages (from python-dateutil>2.7->matplotlib->mediapipe) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

```
[100]: import cv2
import os

# Function to extract frames from a video
def extract_frames(video_path, frame_interval=10):
    cap = cv2.VideoCapture(video_path)
    frame_rate = cap.get(cv2.CAP_PROP_FPS)

    output_folder = "imgframes"
    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    frame_number = 0
    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        if frame_number % (frame_rate * frame_interval) == 0:
            frame_path = os.path.join(output_folder, f"frame_{frame_number}.jpg")
            cv2.imwrite(frame_path, frame)
            print(f"Frame {frame_number} saved at {frame_path}")

        frame_number += 1

    cap.release()

# Specify the path to the video file
videofile = "/kaggle/input/cristiano-ronaldo-motivational-speech/Cristiano Ronaldo Motivational Speech.mp4"

# Extract frames from the video
extract_frames(videofile)
```

```
Frame 0 saved at imgframes/frame_0.jpg
Frame 250 saved at imgframes/frame_250.jpg
Frame 500 saved at imgframes/frame_500.jpg
Frame 750 saved at imgframes/frame_750.jpg
Frame 1000 saved at imgframes/frame_1000.jpg
Frame 1250 saved at imgframes/frame_1250.jpg
Frame 1500 saved at imgframes/frame_1500.jpg
Frame 1750 saved at imgframes/frame_1750.jpg
Frame 2000 saved at imgframes/frame_2000.jpg
Frame 2250 saved at imgframes/frame_2250.jpg
Frame 2500 saved at imgframes/frame_2500.jpg
Frame 2750 saved at imgframes/frame_2750.jpg
Frame 3000 saved at imgframes/frame_3000.jpg
Frame 3250 saved at imgframes/frame_3250.jpg
Frame 3500 saved at imgframes/frame_3500.jpg
Frame 3750 saved at imgframes/frame_3750.jpg
Frame 4000 saved at imgframes/frame_4000.jpg
Frame 4250 saved at imgframes/frame_4250.jpg
Frame 4500 saved at imgframes/frame_4500.jpg
Frame 4750 saved at imgframes/frame_4750.jpg
Frame 5000 saved at imgframes/frame_5000.jpg
Frame 5250 saved at imgframes/frame_5250.jpg
Frame 5500 saved at imgframes/frame_5500.jpg
Frame 5750 saved at imgframes/frame_5750.jpg
Frame 6000 saved at imgframes/frame_6000.jpg
Frame 6250 saved at imgframes/frame_6250.jpg
Frame 6500 saved at imgframes/frame_6500.jpg
Frame 6750 saved at imgframes/frame_6750.jpg
```

```
[102]: from IPython.display import Image, display
framepathlist = ["/kaggle/working/imgframes/frame_2250.jpg", "/kaggle/working/imgframes/frame_1750.jpg", "/kaggle/working/imgframes/frame_0.jpg", "/kaggle/working/imgframes/frame_6750.jpg"]
```

```
for item in framelist:  
    display(Image(filename=item))
```





They pass through many bad moments



```
[108]:  
import cv2  
import mediapipe as mp  
import os  
  
# Initialize Mediapipe Face Detection  
mp_face_detection = mp.solutions.face_detection  
  
facedetected = []  
  
def detect_faces(image_path, output_dir):  
    # Read the image  
    image = cv2.imread(image_path)  
  
    # Convert the image to RGB  
    rgb_image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)  
  
    # Initialize the face detection model  
    face_detection = mp_face_detection.FaceDetection(min_detection_confidence=0.3)  
  
    # Run face detection on the image  
    results = face_detection.process(rgb_image)  
  
    # Draw bounding boxes around the faces  
    if results.detections:  
        for i, detection in enumerate(results.detections):  
            bboxC = detection.location_data.relative_bounding_box  
            ih, iw, _ = rgb_image.shape  
            bbox = int(bboxC.xmin * iw), int(bboxC.ymin * ih), \  
                  int(bboxC.width * iw), int(bboxC.height * ih)  
  
            cv2.rectangle(image, bbox, (0, 255, 0), 2)  
  
            # Save the image with bounding box  
            output_path = os.path.join(output_dir, f"output_frame_{i}.jpg")  
            cv2.imwrite(output_path, image)  
            facedetected.append("detected")
```

```

# List of frame paths
frame_path_list = framepathlist

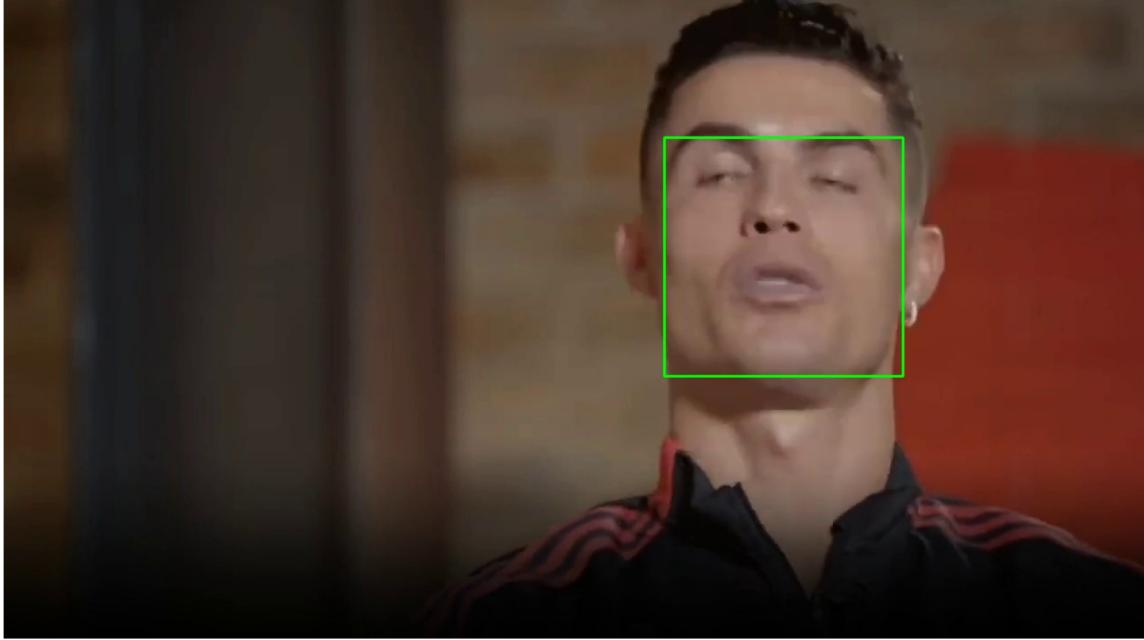
# Output directory for images with bounding boxes
output_directory = "/kaggle/working/output_frames"
if not os.path.exists(output_directory):
    os.makedirs(output_directory)

# Detect faces in each frame and save images with bounding boxes
for frame_path in frame_path_list:
    detect_faces(frame_path, output_directory)

for item in facedetected:
    print(f"Face Detected in Image : {item}")
    display(Image(filename="/kaggle/working/output_frames/output_frame_0.jpg"))

Face Detected in Image : detected
Face Detected in Image : detected

```



```

[112]: import requests

API_URL = "https://api-inference.huggingface.co/models/Salesforce/blip-image-captioning-base"
headers = {"Authorization": "Bearer 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"}

def query(filename):
    with open(filename, "rb") as f:
        data = f.read()
    response = requests.post(API_URL, headers=headers, data=data)
    return response.json()

frameDescription = []
for item in framepathlist:
    output = query(item)
    frameDescription.append(output)

for index, item in enumerate(framepathlist):
    print(frameDescription[index])
    display(Image(filename=item))

```

```
[{'generated_text': "manchester united's new manager, wayne vanderen and manchester united's new manager"}]
```



if the mentality

[{"generated_text": "two men sitting in chairs talking to each other man"}]



because New Year we change the page

[{"generated_text": "crazo's reaction to the referee"}]



[{"generated_text": "a man in a white suit is walking on a field"}]



They pass through many bad moments

```
[{"generated_text": "a man with his eyes closed and a red shirt on"}]
```



[]:



SMA EXP - 5 : Question & Answer

What is the main aim of the work presented by Vadim Moshkin in the paper, and how does it contribute to improving information systems in the context of social media data?

The main aim of the work presented by Vadim Moshkin in the paper is to develop a model of a knowledge base for an information system that collects information from various social networks. This model aims to improve search efficiency and facilitate the unification of data from heterogeneous sources, specifically different social networks [T5].

By developing an ontological model for unifying user data from various social networks, the work helps avoid data redundancy by utilizing graph structures and including contextual information in annotations to ontology relations. This approach is effective in matching information from the profiles of one person across different social networks and when considering historical data. Additionally, the integration of syntactic rules and knowledge engineering methods in forming a semantic tree from text fragments allows for merging the obtained semantic trees into a single subject knowledge base of a specific information resource [T2].

Can you explain the ontological model proposed for the unification of data profiles from different social networks?

The ontological model proposed for the unification of data profiles from different social networks aims to provide a structured framework for organizing and integrating data from heterogeneous sources. Here is an overview of the key components of the ontological model:

1. Nodes and Classes: The model consists of a set of nodes representing objects and classes within the ontology. These nodes encompass users of social networks and internal objects specific to each social network [T5].
2. Ontology Relations: The model includes ontology relations that define the connections between different elements within the ontology. These relations help

establish the relationships and associations between various data entities extracted from social networks [T5].

3. Ontology Interpretation Functions: The model incorporates ontology interpretation functions or axioms that provide additional context and meaning to the relationships between nodes and classes. These functions help in interpreting and understanding the data within the ontology [T5].

4. Translation of Elements: The model includes a translation mechanism that maps elements from different social networks to objects within the ontological model. This translation process helps in standardizing and unifying the data extracted from various sources [T5].

By structuring the data profiles from different social networks using this ontological model, the approach aims to enhance the efficiency of data processing, analysis, and retrieval. It enables the integration of information from diverse sources into a cohesive model, facilitating better understanding and utilization of the collected data for various applications in information systems and social media analysis [T5].

What are the key methods and approaches mentioned in the paper for obtaining representative sets of social network users? Discuss the "width traversal method," the "forest fire" method, and the "Metropolis- Hastings method."

The paper discusses several methods for obtaining representative sets of social network users, which are essential for collecting information from social networks. Here is an overview of the key methods mentioned in the paper:

1. Width Traversal Method: The width traversal method is a technique used to explore social networks by traversing them in a breadth-first manner. This method involves visiting nodes at each level of the network before moving on to the next level. By systematically exploring the network in this way, the width traversal method can provide a comprehensive view of the connections and relationships between users within the network [T5].

2. Forest Fire Method: The forest fire method is another approach for sampling large graphs, including social networks. In this method, a "fire" is started at a selected node, and it spreads through the network based on predefined rules. The nodes that are "burned" by the fire represent the sampled nodes. This method is useful for efficiently sampling large graphs and capturing the local structure of the network [T5].

3. Metropolis-Hastings Method: The Metropolis-Hastings method is a Markov chain Monte Carlo (MCMC) algorithm commonly used for sampling from complex probability distributions. In the context of social networks, the Metropolis-Hastings method can be applied to sample users or nodes based on certain criteria or probabilities. By iteratively sampling nodes and transitioning between states, this method can provide a representative set of users from the social network [T5].

These methods play a crucial role in sampling users from social networks for various research and analysis purposes. Each method offers a unique approach to sampling users and capturing the network's structure, enabling researchers to obtain representative sets of users for further analysis and modeling [T5].