```
#Instalar as bibliotecas necessárias (caso não estejam instaladas)
!pip install transformers
!pip install openpyxl
!pip install torch
```

Mostrar saída oculta

```
import torch
from transformers import XLMRobertaTokenizer, XLMRobertaForSequenceClassification, pipeline
import pandas as pd


#Subir arquivo diretamente no Colab
from google.colab import files
uploaded = files.upload()
```

Escolher Arquivos Nenhum arquivo escolhido Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
#Carregar o dataset
file_path = list(uploaded.keys())[0]  # Obtém o nome do arquivo enviado
df = pd.read_excel(file_path)


#Carregar modelo pré-treinado da Hugging Face
model_name = 'cardiffnlp/twitter-xlm-roberta-base-sentiment'


#Usar o tokenizer e modelo corretos para XLM-RoBERTa
tokenizer = XLMRobertaTokenizer.from_pretrained(model_name)
model = XLMRobertaForSequenceClassification.from_pretrained(model_name)
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
sentencepiece.bpe.model: 100%  5.07M/5.07M [00:00<00:00, 14.8MB/s]
special_tokens_map.json: 100%  150/150 [00:00<00:00, 6.59kB/s]
config.json: 100%  841/841 [00:00<00:00, 13.8kB/s]
pytorch_model.bin: 100%  1.11G/1.11G [00:47<00:00, 23.6MB/s]
```

```
#Criar pipeline de análise de sentimentos
nlp = pipeline('sentiment-analysis', model=model, tokenizer=tokenizer)
```

```
Device set to use cpu
```

```
#Aplicar a análise de sentimentos
results = nlp(df['Comentário'].tolist())


#Mapear os sentimentos para valores numéricos (opcional)
sentiment_mapping = {
    'negative': 1,
    'neutral': 3,
    'positive': 5
}


df['Sentimento'] = [sentiment_mapping.get(result['label'], None) for result in results]


#Salvar os resultados em um novo arquivo Excel
output_file_path = '/content/here_arquivo_ml_text_output.xlsx'
df.to_excel(output_file_path, index=False)


#Oferecer para download o arquivo resultante
files.download(output_file_path)
```

```
#Instalar as bibliotecas necessárias
!pip install nltk wordcloud matplotlib seaborn
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-packages (3.9.1)
Requirement already satisfied: wordcloud in /usr/local/lib/python3.11/dist-packages (1.9.4)
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: click in /usr/local/lib/python3.11/dist-packages (from nltk) (8.1.8)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.11/dist-packages (from nltk) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nltk) (4.67.1)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.11/dist-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-packages (from wordcloud) (11.1.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.11/dist-packages (from seaborn) (2.2.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.2->seaborn) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.2->seaborn) (2025.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
```

```
#Importar as bibliotecas
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from collections import Counter
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import string
```

```
#Carregar o arquivo com os comentários e sentimentos (certifique-se de fazer o upload do arquivo)
from google.colab import files
uploaded = files.upload()
```

```
Escolher Arquivos Nenhum arquivo escolhido Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
```

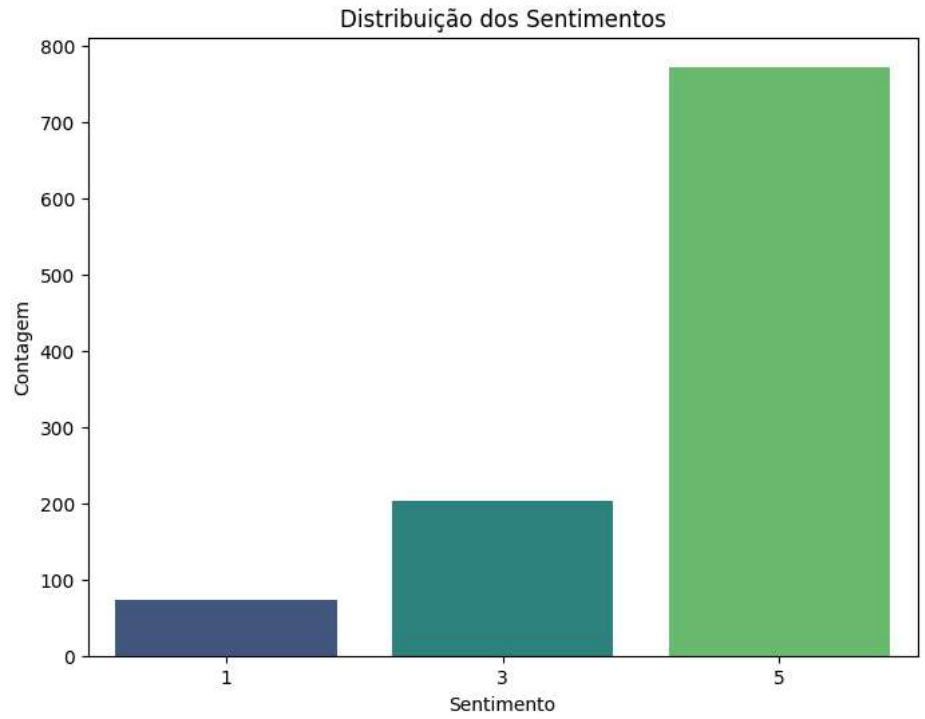
```
#Carregar o dataset
file_path = 'here_arquivo_ml_text_output.xlsx'  # Use o nome do arquivo que você fez o upload
df = pd.read_excel(file_path)
```

```
#Exibição gráfica da distribuição dos sentimentos
plt.figure(figsize=(8, 6))
sns.countplot(x='Sentimento', data=df, palette='viridis')
plt.title('Distribuição dos Sentimentos')
plt.xlabel('Sentimento')
plt.ylabel('Contagem')
plt.show()
```

```
<ipython-input-17-01bd8509561c>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.countplot(x='Sentimento', data=df, palette='viridis')
```




```
#Estatísticas descritivas sobre os sentimentos
sentimento_stats = df['Sentimento'].describe()
print("Estatísticas Descritivas sobre os Sentimentos:")
print(sentiment_stats)
```

```
Estatísticas Descritivas sobre os Sentimentos:
count    1050.000000
mean        4.333333
std         1.202687
min         1.000000
25%         3.000000
50%         5.000000
75%         5.000000
max         5.000000
Name: Sentimento, dtype: float64
```

```
#Análise qualitativa: Frequência de palavras
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
True
```

```
#Preparação do texto para análise de frequência
all_comments = ' '.join(df['Comentário'].dropna().tolist())
```

```
#Tokenização
tokens = word_tokenize(all_comments.lower())
```

```
#Remover pontuação e stopwords
stop_words = set(stopwords.words('portuguese'))  # Definindo as stopwords em português
punctuation = set(string.punctuation)  # Definindo os sinais de pontuação
filtered_tokens = [word for word in tokens if word.isalnum() and word not in stop_words and word not in punctuation]
```

```
#Lematização: para reduzir as palavras à sua raiz (opcional, mas melhora o processo)
lemmatizer = WordNetLemmatizer()
lemmatized_tokens = [lemmatizer.lemmatize(word) for word in filtered_tokens]


#Frequência das palavras
word_counts = Counter(lemmatized_tokens)
most_common_words = word_counts.most_common(10)


#Exibir as 10 palavras mais comuns
print("\n10 Palavras mais comuns:")
for word, count in most_common_words:
    print(f"{word}: {count}")
```

```
10 Palavras mais comuns:
matemática: 150
parabéns: 127
mm: 111
seminário: 97
evento: 79
obrigada: 65
todos: 65
ter: 62
bom: 61
gostaria: 60
```

```
#Gerar uma nuvem de palavras
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(word_counts)



#Exibir a nuvem de palavras
plt.figure(figsize=(10, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Nuvem de Palavras')
plt.show()
```


```
from nltk import ngrams
from collections import Counter


#Gerar bigramas (duas palavras consecutivas)
bigrams = ngrams(filtered_tokens, 2)
bigram_counts = Counter(bigrams)


#Exibir os 10 bigramas mais comuns
print("\n10 Bigramas mais comuns:")
for bigram, count in bigram_counts.most_common(10):
    print(f"{' '.join(bigram)}: {count}")
```

```
10 Bigramas mais comuns:
mentalidades matemáticas: 43
sala aula: 24
parabéns evento: 13
fazer parte: 12
ensino matemática: 12
```

```
mentalidade matemática: 12
quero agradecer: 11
agradecer oportunidade: 11
parabéns todos: 9
gostaria agradecer: 8
```

```
#Gerar trigramas (três palavras consecutivas)
trigrams = ngrams(filtered_tokens, 3)
trigram_counts = Counter(trigrams)


#Exibir os 10 trigramas mais comuns
print("\n10 Trigramas mais comuns:")
for trigram, count in trigram_counts.most_common(10):
    print(f"{' '.join(trigram)}: {count}")
```

```
10 Trigramas mais comuns:
sobre mentalidades matemáticas: 5
saber sobre maleta: 5
primeiro dia seminário: 4
quero agradecer oportunidade: 4
parabéns toda equipe: 4
ensinar aprender matemática: 3
gostaria saber sobre: 3
bom participar desse: 3
participar desse seminário: 3
parabéns equipe mm: 3
```

```
from nltk import ngrams
from collections import defaultdict

# Análise de sentimentos para bigramas
bigram_sentiment = defaultdict(lambda: {'positive': 0, 'neutral': 0, 'negative': 0})

# Percorrer os comentários e os bigramas para classificar sentimentos
for comment, sentiment in zip(df['Comentário'], df['Sentimento']):
    tokens = word_tokenize(comment.lower())
    bigrams_in_comment = ngrams(tokens, 2)
    for bigram in bigrams_in_comment:
        sentiment_label = 'positive' if sentiment > 3 else 'negative' if sentiment < 3 else 'neutral'
        bigram_sentiment[bigram][sentiment_label] += 1

# Exibir os bigramas com maior associação a sentimentos
for bigram, sentiment_counts in bigram_sentiment.items():
    print(f"Bigram: {' '.join(bigram)} | Sentimentos: {sentiment_counts}")
```

Mostrar saída oculta

```
import spacy
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

!python -m spacy download pt_core_news_lg
```

Mostrar saída oculta

```
import spacy
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

!python -m spacy download pt_core_news_sm # download the correct model
# %%
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import pandas as pd
import spacy

# Carregar modelo de linguagem do spaCy para português
nlp = spacy.load("pt_core_news_sm") # load the correct model

# Função para pré-processar o texto
def preprocess_text(text):
    # Processar o texto com spaCy
    doc = nlp(text)

    # Filtrar tokens: remover stopwords, pontuações e espaços em branco
    filtered_tokens = [
        token.text.lower() for token in doc
        if not token.is_stop and not token.is_punct and not token.is_space
    ]
```

```
        # Juntar os tokens em uma única string
        return " ".join(filtered_tokens)

    # Filtrar os comentários para sentimentos específicos
    positive_comments = df[df['Sentimento'] > 3]['Comentário'].dropna()
    negative_comments = df[df['Sentimento'] < 3]['Comentário'].dropna()

    # Pré-processar os comentários
    positive_comments_cleaned = positive_comments.apply(preprocess_text)
    negative_comments_cleaned = negative_comments.apply(preprocess_text)

    # Gerar nuvem de palavras para sentimentos positivos
    positive_text = ' '.join(positive_comments_cleaned)
    positive_wordcloud = WordCloud(width=800, height=400, background_color='white').generate(positive_text)

    # Gerar nuvem de palavras para sentimentos negativos
    negative_text = ' '.join(negative_comments_cleaned)
    negative_wordcloud = WordCloud(width=800, height=400, background_color='white').generate(negative_text)

    # Exibir nuvem de palavras
    plt.figure(figsize=(10, 8))
    plt.imshow(positive_wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title("Nuvem de Palavras - Sentimentos Positivos")
    plt.show()

    plt.figure(figsize=(10, 8))
    plt.imshow(negative_wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title("Nuvem de Palavras - Sentimentos Negativos")
    plt.show()
```

```
Collecting pt-core-news-sm==3.7.0
  Downloading https://github.com/explosion/spacy-models/releases/download/pt_core_news_sm-3.7.0/pt_core_news_sm-3.7.0-py3-none-any
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.0/13.0 MB 32.2 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.8.0,>=3.7.0 in /usr/local/lib/python3.11/dist-packages (from pt-core-news-sm==3.7.0) (3.7.
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core
Requirement already satisfied: thinc<8.3.0,>=8.2.2 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-c
Requirement already satisfied: weasel<0.5.0,>=0.1.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core
Requirement already satisfied: typer<1.0.0,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-c
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-news-sm==3.7.
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-news-sm==
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-news
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-c
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.11/dist-packages (from spacy<3.8.0,>=3.7.0->pt-core-news-sm
Requirement already satisfied: language-data>=1.2 in /usr/local/lib/python3.11/dist-packages (from langcodes<4.0.0,>=3.2.0->spacy
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic!=1.8,!=1.8.1,<3.0
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-packages (from pydantic!=1.8,!=1.8.1,<3.0.
Requirement already satisfied: typing-extensions>=4.12.2 in /usr/local/lib/python3.11/dist-packages (from pydantic!=1.8,!=1.8.1,<
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.13.0-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.13.0->spacy<3.8.0
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.11/dist-packages (from thinc<8.3.0,>=8.2.2->spacy<3.8
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.11/dist-packages (from thinc<8.3.0,>=8.2.2->spa
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0.0,>=0.3.0->spacy<3.8.0,>=3
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0.0,>=0.3.0->spacy<3.8
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0.0,>=0.3.0->spacy<3.8.0,>=
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.11/dist-packages (from weasel<0.5.0,>=0.1.0->s
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.11/dist-packages (from weasel<0.5.0,>=0.1.0->spa
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from jinja2->spacy<3.8.0,>=3.7.0->pt-c
Requirement already satisfied: marisa-trie>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from language-data>=1.2->langcodes<4
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0.0,
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0.0
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-packages (from smart-open<8.0.0,>=5.2.1->weasel<0.5.0,>=0.1
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0->t
Installing collected packages: pt-core-news-sm
Successfully installed pt-core-news-sm-3.7.0
✓ Download and installation successful
You can now load the package via spacy.load('pt_core_news_sm')
⚠ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python in
order to load all the package's dependencies. You can do this by selecting the
'Restart kernel' or 'Restart runtime' option.
```

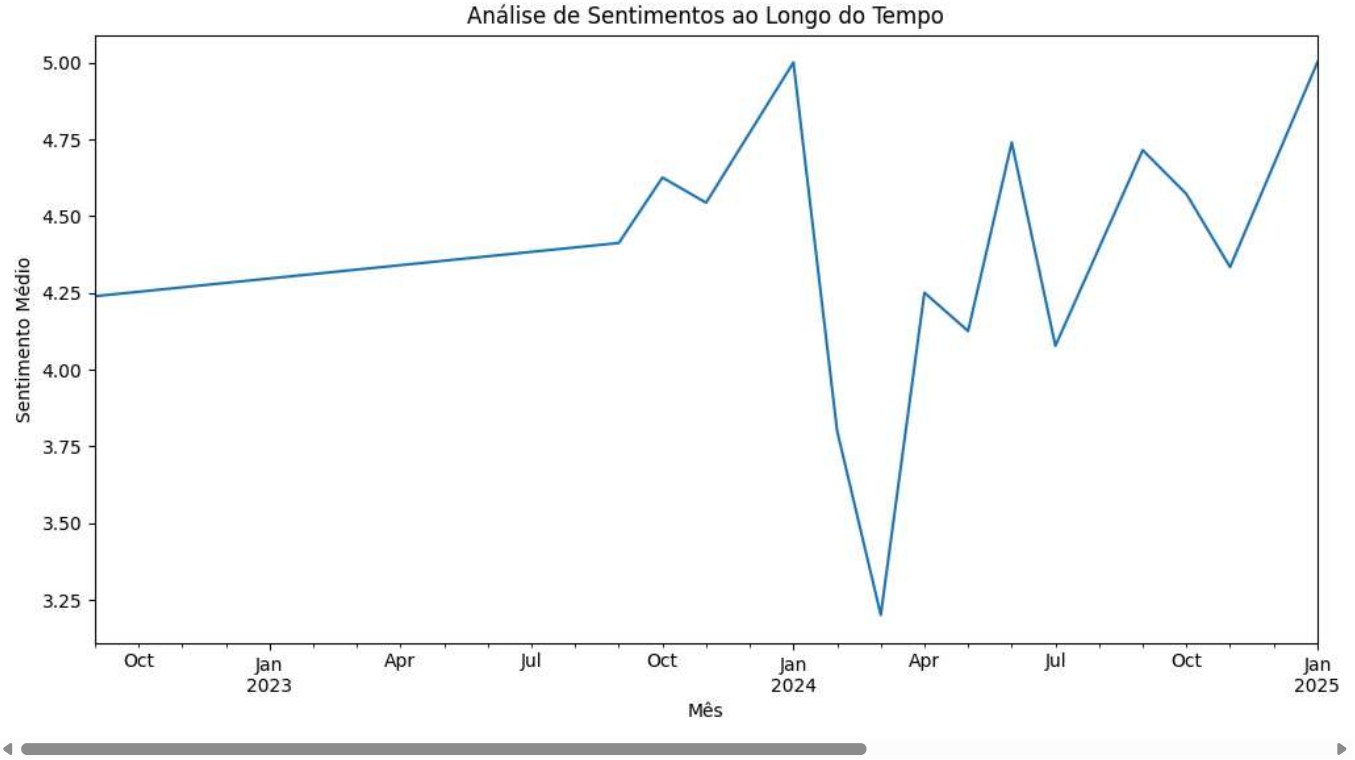
Nuvem de Palavras - Sentimentos Positivos

Nuvem de Palavras - Sentimentos Negativos


```
# Adicionar uma coluna de data se necessário
# df['data'] = pd.to_datetime(df['data'])

#Calcular a média dos sentimentos por mês
df['month'] = df['Data'].dt.to_period('M')
sentiment_over_time = df.groupby('month')['Sentimento'].mean()

#Exibir a evolução do sentimento ao longo do tempo
sentiment_over_time.plot(figsize=(12, 6))
plt.title('Análise de Sentimentos ao Longo do Tempo')
plt.xlabel('Mês')
plt.ylabel('Sentimento Médio')
plt.show()
```




```
#Instalar pyLDAvis (se ainda não estiver instalado)
!pip install pyLDAvis

#Importar bibliotecas necessárias
import pyLDAvis
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
import nltk
import pandas as pd
from google.colab import files  # Para carregar arquivos no Google Colab

# Baixar stopwords em português (se ainda não estiverem baixadas)
nltk.download('stopwords')

# Carregar stopwords em português
stop_words = nltk.corpus.stopwords.words('portuguese')

# Carregar o arquivo Excel
uploaded = files.upload()  # Abre a interface para upload de arquivos
file_path = list(uploaded.keys())[0]  # Obtém o nome do arquivo enviado
df = pd.read_excel(file_path)  # Carrega o arquivo Excel em um DataFrame

# Criar o vetorizador com stopwords em português
vectorizer = CountVectorizer(stop_words=stop_words)
```