2023

Relatório do Projeto da UC de Análise e Visualização de Dados

DOCENTES: ALVARO IRIARTE | JOSÉ JOÃO

Realizado por: Inês Catarina Teixeira Leão | PG50855

Mestrado em Humanidades Digitais

Conteúdo

Introdução

Este relatório apresenta os resultados de um trabalho de análise e visualização de dados realizado com o objetivo de extrair informações pertinentes e insights significativos a partir de conjuntos de dados mais complexos. Neste projeto, adotou-se uma abordagem abrangente para explorar os conjuntos de dados disponíveis, utilizando técnicas avançadas de análise estatística, mineração de dados e visualização interativa. Ao longo deste relatório, serão apresentadas as etapas do processo de análise e visualização de dados, descrevendo as técnicas e ferramentas utilizadas, bem como os principais resultados obtidos. As obras escolhidas foram: "O Romance de um Homem Rico", "A Mulher Fatal", "Amor de Salvação" e "Onde Está a Felicidade?".

Importante referir que as obras se encontram em formato de *one sentence per line*, todos os programas criados.

Sumário Executivo

Este relatório apresenta os resultados de um trabalho de análise e visualização de dados sobre quatro obras de Camilo Castelo Branco: "O Romance de um Homem Rico", "A Mulher Fatal", "Amor de Salvação" e "Onde Está a Felicidade?". O objetivo do estudo foi analisar diversos aspetos das obras, nomeadamente a extração de lemas, identificação de entidades, análise de adjetivos e avaliação de sentimentos em cada capítulo.

Utilizando técnicas de processamento de linguagem natural e programação em Python, foram aplicadas bibliotecas como *SpaCy* e *NLTK* para realizar a análise dos dados.

obra, como tristeza, alegria, melancolia ou raiva, contribuindo para uma análise mais abrangente do tom emocional presente nos textos.

Estas análises e visualizações dos dados fornecem uma visão detalhada das características linguísticas, temáticas e emocionais presentes nas obras de Camilo Castelo Branco, aqui trabalhadas. Os resultados obtidos podem ser usados para *insights* adicionais sobre o estilo de escrita do autor, por exemplo, a evolução dos personagens ao longo das histórias e os temas abordados nas obras.

Ademais, note-se que a compreensão destes elementos permite uma apreciação mais profunda das obras de Camilo Castelo Branco, bem como o enriquecimento da crítica literária e estudos comparativos dos seus trabalhos. Além disso, essas análises e visualizações podem servir de base para futuras pesquisas e estudos relacionados à obra do autor. Este relatório além de apresentar os resultados relevantes da análise, acaba por destacar por si só a importância da análise e visualização de dados para a compreensão e apreciação da literatura.

Objeto do relatório

O objeto do relatório é fornecer uma análise detalhada e abrangente das quatro obras de Camilo Castelo Branco: "O Romance de um Homem Rico", "A Mulher Fatal", "Amor de Salvação" e "Onde Está a Felicidade?". O relatório tem como objetivo principal investigar e extrair informações relevantes a partir dos dados contidos nessas obras, utilizando técnicas de análise de texto, processamento de linguagem natural e análise de sentimentos.

Caderno de campo

O caderno de campo desempenha um papel fundamental no trabalho de Análise e Visualização de Dados (AVD) realizado com as obras de Camilo Castelo Branco. Este documento serve como um registo detalhado das atividades, observações e descobertas feitas durante a análise dos dados extraídos das obras selecionadas: "O Romance de um Homem Rico", "A Mulher Fatal", "Amor de Salvação" e "Onde Está a Felicidade?". Este documenta as técnicas e ferramentas utilizadas para a extração dados presentes em cada obra. Ao longo deste documento, se registadas as decisões metodológicas, as dificuldades encontradas e as observações relevantes que surgirem durante a análise dos dados.

1. A Mulher Fatal

Para esta obra, foi analisado e extraído as entidades do texto, as localidades, os lemas e feita uma análise de sentimento por cada capítulo do livro. O documento de texto que contem a obra, foi utilizado sempre em formato de *one sentence per line.*

1.1 Entidades

Para extrair as entidades da obra *A mulher fatal*, utilizei o seguinte programa *Python*, utilizando a biblioteca *spaCy*

```python
import spacy
import os
import csv

with open('a_mulher_fatal1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

characters = []

for ent in doc.ents:
    if ent.label_ == "PER":
        characters.append(ent.text)

from collections import Counter

characters_counts = Counter(characters)

top30_characters = characters_counts.most_common(30)

print(top30_characters)

with open('Entidades_amulherfatal.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Personagem', 'Contagem'])  # Adiciona o cabeçalho das colunas

    for character, count in top30_characters:
        writer.writerow([character, count])
```

Este programa extrairá um Top 30 personagens, juntamente com o número de vezes que cada uma surge. O programa cria um documento *csv* com os resultados:

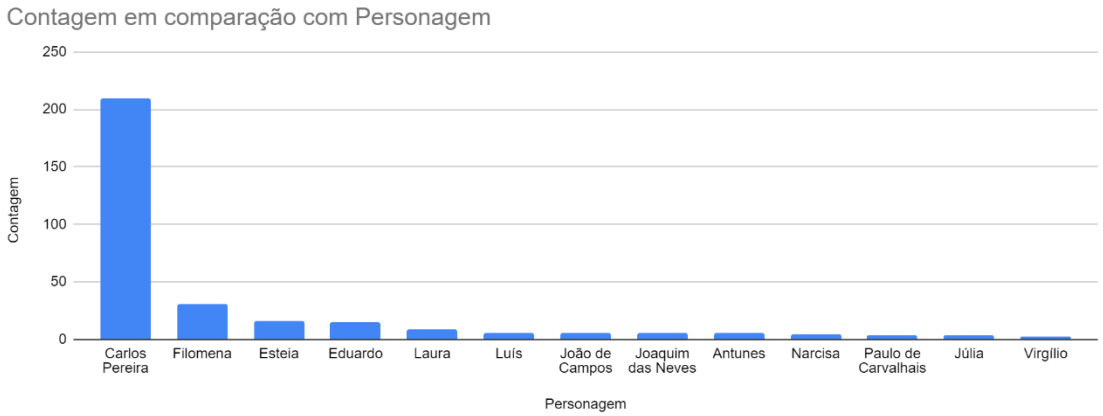
	Personagem	Contagem
2	Carlos	172
3	Carlos Pereira	34
4	Vossa Excelênci	32
5	Filomena	31
6	Deus	22
7	Esteia	16
8	Eduardo	15
9	Vossa Senhoria	10
10	Laura	9
11	Olhe	8
12	Fui	8
13	Quero	7
14	Luís	5
15	João de Campos	5
16	Tenho	5
17	Joaquim das Ne	5
18	Antunes	5
19	Espero	4

18	Antunes	5
19	Espero	4
20	Narcisa	4
21	Querem	4
22	Voltaire	3
23	Paulo de Carvall	3
24	Júlia	3
25	Ali	3
26	Diga	3
27	Ouvi	3
28	Carlos Pereira	3
29	conde de ***	3
30	Virgílio	2
31	Satã	2
32		
33		

Em seguida, prossegui com uma limpeza manual ao documento, deixando apenas os resultados relevantes para o que são as personagens do livro. Assim:

	A	B
1	Personagem	Contagem
2	Carlos Pereira	209
3	Filomena	31
4	Deus	22
5	Esteia	16
6	Eduardo	15
7	Laura	9
8	Luís	5
9	João de Campos	5
10	Joaquim das Ne	5
11	Antunes	5
12	Narcisa	4
13	Voltaire	3
14	Paulo de Carvall	3
15	Júlia	3
16	Virgílio	2
17		
18		
19		

Por fim, criei um gráfico de barras como proposta de visualização. Na figura abaixo:



1.2 Análise de Sentimento

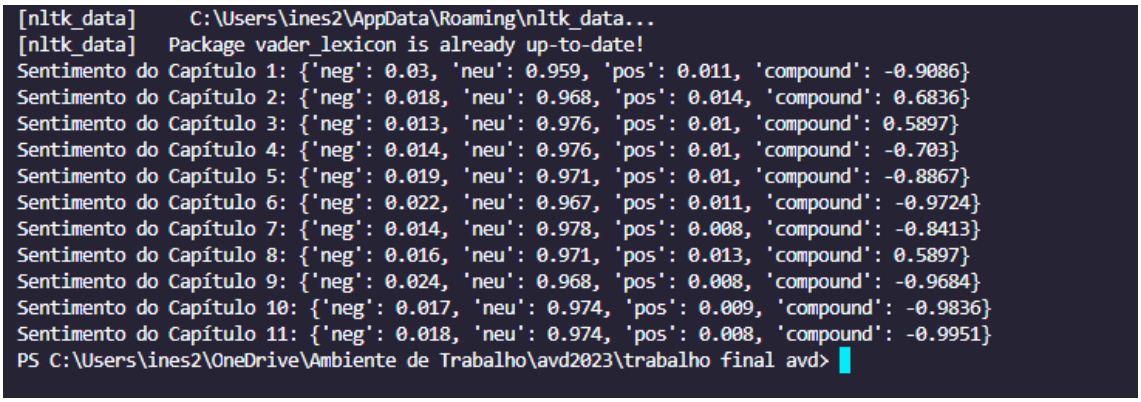
Para fazer a analise de sentimento de cada capítulo, criei o seguinte programa *Python,* que utiliza a biblioteca *NLTK (Natural Language Toolkit*) para realizar uma análise de sentimentos num dado texto. Ele utiliza o módulo `*SentimentIntensityAnalyzer*` do *NLTK*, que é responsável por atribuir pontuações de sentimento a diferentes partes do texto.

```python
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer

nltk.download('vader_lexicon')

sia = SentimentIntensityAnalyzer()

caminho_arquivo = "a_mulher_fatal1spl.txt"

with open(caminho_arquivo, 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

capitulos = texto.split('# Capítulo')

for i, capitulo in enumerate(capitulos[1:]):
    sentiment = sia.polarity_scores(capitulo)
    print(f"Sentimento do Capítulo {i+1}: {sentiment}")
```

Primeiro, o código importa as bibliotecas necessárias e faz o download do léxico *Vader do NLTK*, que é usado para análise de sentimentos. Em seguida, é criada uma instância do `*SentimentIntensityAnalyzer*`.

Depois, o código lê o arquivo *txt* da obra *A mulher fatal* no formato de *one sentence per line* e armazena-o numa variável. O texto é dividido em capítulos usando um marcador específico (no caso, foi colocada um # em cada capítulo no documento de texto). Através de um *loop,* cada capítulo é analisado usando o `*SentimentIntensityAnalyzer*`, que retorna uma pontuação de sentimento para aquele capítulo.

Em resumo, o código realiza a análise de sentimentos no texto, dividindo-o em capítulos e calculando a pontuação de sentimento para cada um deles. Os resultados no terminal são:



Nos resultados exibidos, cada capítulo possui um dicionário de pontuações que indica a intensidade do sentimento positivo, negativo e neutro. Além disso, há o valor do "*compound*".

- *'neg'*: pontuação de negatividade do capítulo.

- *'neu'*: pontuação de neutralidade do capítulo.

- *'pos'*: pontuação de positividade do capítulo.

- *'compound'*: pontuação geral do sentimento do capítulo, que é calculada com base nas pontuações anteriores e representa uma medida agregada do sentimento do texto. O valor do *compound* varia de -1 a 1, onde -1 indica um sentimento extremamente negativo, 1 indica um sentimento extremamente positivo e 0 indica neutralidade.

Neste caso, os valores para *'compound'* variam entre -0.9951 e 0.6836. Os valores mais próximos de -1 indicam sentimentos negativos, os valores mais próximos de 1 indicam sentimentos positivos e valores próximos de 0 indicam sentimentos neutros.

Portanto, analisando os resultados, podemos inferir que o Capítulo 1 possui um sentimento negativo mais forte (-0.9086), enquanto o Capítulo 2 possui um sentimento positivo moderado (0.6836). Já o Capítulo 3 também tem um sentimento positivo, porém menos intenso (0.5897).

Para criar uma proposta de visualização, criei o seguinte programa:

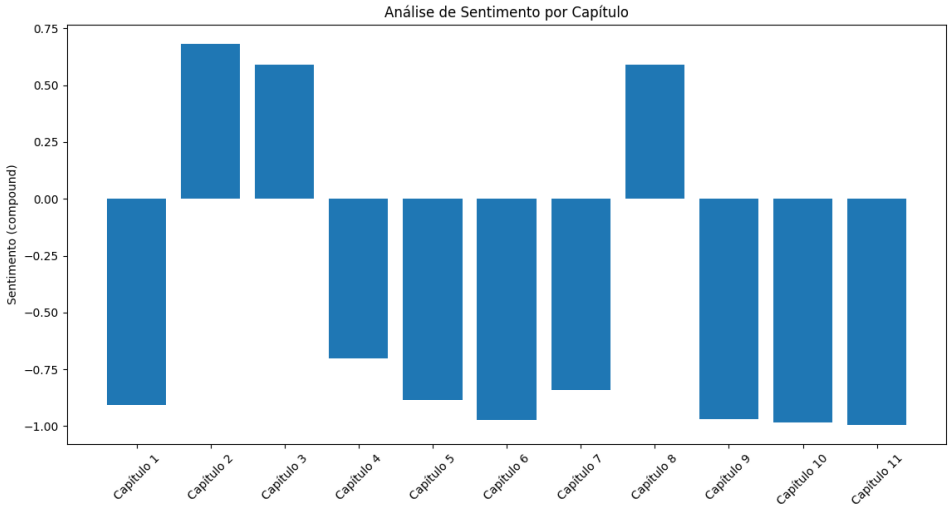
```python
import matplotlib.pyplot as plt

pontuacoes = [
    {'neg': 0.03, 'neu': 0.959, 'pos': 0.011, 'compound': -0.9086},
    {'neg': 0.018, 'neu': 0.968, 'pos': 0.014, 'compound': 0.6836},
    {'neg': 0.013, 'neu': 0.976, 'pos': 0.01, 'compound': 0.5897},
    {'neg': 0.014, 'neu': 0.976, 'pos': 0.01, 'compound': -0.703},
    {'neg': 0.019, 'neu': 0.971, 'pos': 0.01, 'compound': -0.8867},
    {'neg': 0.022, 'neu': 0.967, 'pos': 0.011, 'compound': -0.9724},
    {'neg': 0.014, 'neu': 0.978, 'pos': 0.008, 'compound': -0.8413},
    {'neg': 0.016, 'neu': 0.971, 'pos': 0.013, 'compound': 0.5897},
    {'neg': 0.024, 'neu': 0.968, 'pos': 0.008, 'compound': -0.9684},
    {'neg': 0.017, 'neu': 0.974, 'pos': 0.009, 'compound': -0.9836},
    {'neg': 0.018, 'neu': 0.974, 'pos': 0.008, 'compound': -0.9951}
]

valores_compound = [sentimento['compound'] for sentimento in pontuacoes]

rotulos = [f"Capítulo {i+1}" for i in range(len(pontuacoes))]

plt.figure(figsize=(10, 6))
plt.bar(rotulos, valores_compound)
plt.xlabel('Capítulo')
plt.ylabel('Sentimento (compound)')
plt.title('Análise de Sentimento por Capítulo')
plt.xticks(rotation=45)
plt.show()
```

O código em questão utiliza a biblioteca Matplotlib em conjunto com os resultados da análise de sentimento por capítulos para criar um gráfico de barras. O objetivo é visualizar a variação do sentimento ao longo dos capítulos de uma obra. Inicialmente, os resultados da análise de sentimento por capítulo são armazenados numa lista chamada `*pontuacoes*`. Cada elemento dessa lista é um dicionário que contém as pontuações de negatividade (*'neg'*), neutralidade (*'neu'*), positividade (*'pos'*) e a pontuação geral do sentimento (*'compound'*) para um capítulo específico.

O próximo passo consiste em extrair os valores do 'compound' de cada dicionário da lista `pontuacoes`. Esses valores são armazenados numa lista separada chamada `valores_compound`. Esta lista representa a medida agregada do sentimento para cada capítulo. A lista `rotulos` é criada para armazenar os rótulos dos capítulos. Cada rótulo é definido como "Capítulo X", onde X é o número do capítulo correspondente. Com todos os dados necessários preparados, o código utiliza a biblioteca Matplotlib para criar o gráfico de barras. É criada uma figura com um tamanho específico utilizando a função `plt.figure(figsize=(10, 6))`. Em seguida, a função `plt.bar()` é utilizada para criar as barras do gráfico. Os rótulos dos capítulos são exibidos no eixo x e os valores do compound são representados no eixo y. Algumas funções do Matplotlib, como `plt.xlabel()`, `plt.ylabel()`, `plt.title()` e `plt.xticks(rotation=45)`, são utilizadas para adicionar rótulos aos eixos e ao gráfico em geral, além de fazer uma rotação de 45 graus

nos rótulos dos capítulos para facilitar a leitura. Por fim, o gráfico é exibido na tela utilizando a função `plt.show()`.

Assim, o gráfico:



Conclui-se que, na maior parte dos capítulos, o sentimento é negativo.

1.3 Lemas

Para extrair um top 50 lemas da obra, foi criado o seguinte programa:

```python
import nltk
from nltk import FreqDist
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import csv

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

def extract_lemmas(text):
    lemmatizer = WordNetLemmatizer()
    stop_words = set(stopwords.words('english'))

    tokens = nltk.word_tokenize(text)
    lemmas = [lemmatizer.lemmatize(token.lower()) for token in tokens if token.isalnum() and token.lower() not in stop_words]

    return lemmas

def save_lemmas_to_csv(lemmas, output_file):
    fdist = FreqDist(lemmas)
    top_lemmas = fdist.most_common(50)

    with open(output_file, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(['Lemma', 'Frequency'])
        for lemma, freq in top_lemmas:
            writer.writerow([lemma, freq])

input_file = 'a_mulher_fatal1spl.txt'
```

```python
output_file = 'TOP50a_mulher_fatal.csv'

with open(input_file, 'r') as file:
    text = file.read()

lemmas = extract_lemmas(text)

save_lemmas_to_csv(lemmas, output_file)
```

O programa utiliza o NLTK para extrair os lemas da obra, carregada em formato de texto, bem como calcula a frequência que cada um aparece no texto. Assim, os 50 lemas mais frequentes são guardados num arquivo novo em formato csv. Tem-se:

	Lemma	Frequency
1	que	1625
2	de	1620
3	e	1219
4	não	692
5	da	655
6	se	506
7	com	426
8	em	380
9	o	374
10	um	335
11	eu	321
12	carlos	303
13	para	299
14	é	299
15	uma	253
16	ao	236
17	na	235
18	por	218

20	à	214
21	do	200
22	ele	180
23	ma	179
24	como	179
25	meu	176
26	mais	173
27	sua	166
28	era	163
29	lhe	150
30	seu	145
31	ano	139
32	quando	135
33	ela	131
34	minha	117
35	amigo	108
36	cassilda	106
37	onde	104
38	ou	103

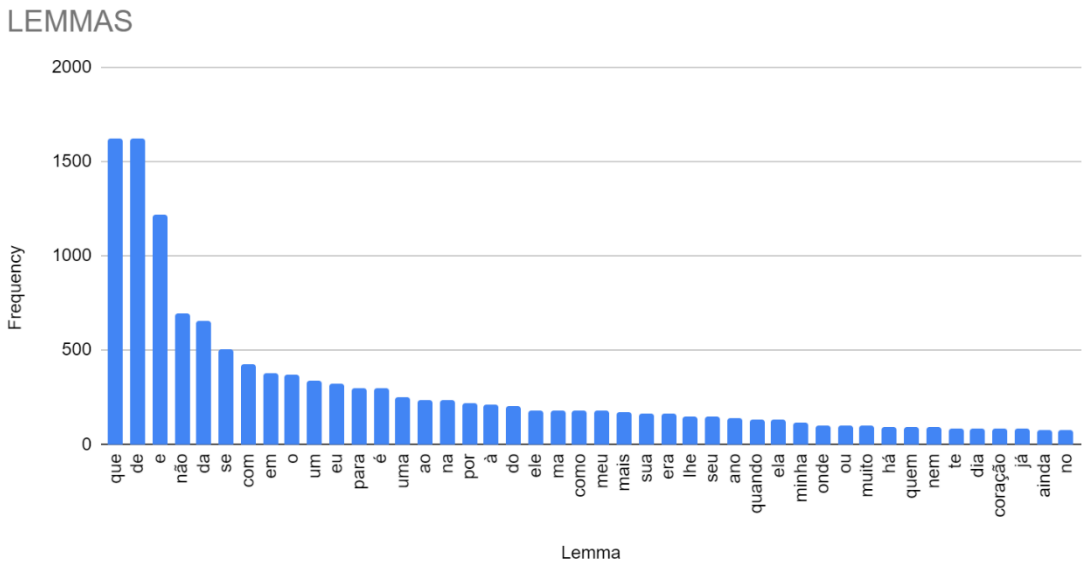
39	muito	99
40	mulher	95
41	há	95
42	senhora	95
43	disse	92
44	quem	91
45	nem	91
46	te	85
47	dia	85
48	coração	84
49	já	83
50	ainda	80
51	no	79
52		

Depois, procedi à limpeza manual do mesmo. Resultou:

	Lemma	Frequency
2	que	1625
3	de	1620
4	e	1219
5	não	692
6	da	655
7	se	506
8	com	426
9	em	380
10	o	374
11	um	335
12	eu	321
13	para	299
14	é	299
15	uma	253
16	ao	236
17	na	235
18	por	218
19	à	214

20	do	200
21	ele	180
22	ma	179
23	como	179
24	meu	176
25	mais	173
26	sua	166
27	era	163
28	lhe	150
29	seu	145
30	ano	139
31	quando	135
32	ela	131
33	minha	117
34	onde	104
35	ou	103
36	muito	99
37	há	95
38	quem	91
39	nem	91
40	te	85
41	dia	85
42	coração	84
43	já	83
44	ainda	80
45	no	79
46		
47		
48		
49		
50		
51		

Como proposta de visualização, criei o seguinte gráfico:



Conclui-se que o lema “que” é o mais frequente.

1.4 Lugares

Por fim, para esta obra, foi pertinente analisar os nomes de lugares que surgem na obra, bem como calcular a sua frequência. Assim, foi criado o seguinte programa:

```python
import spacy
import os
import csv

with open('a_mulher_fatal1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

locations = []

for ent in doc.ents:
    if ent.label_ == "LOC":
        locations.append(ent.text)

from collections import Counter

locations_counts = Counter(locations)

top30_locations = locations_counts.most_common(30)

print(top30_locations)

with open('LOCALIDADESa_mulher_fatal.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Localidade', 'Contagem'])  # Adiciona o cabeçalho das colunas

    for location, count in top30_locations:
        writer.writerow([location, count])

```

Depois de carregar o documento de texto com a obra, em formato de *one sentence per line*, e o armazenar numa variável, a biblioteca *SpaCy* é carregada com um modelo de processamento de texto em português. O texto é analisado usando o modelo carregado, e o código procura todas as entidades que representam localidades. Caso uma entidade seja identificada como uma localidade, ela é adicionada a uma lista. Em seguida, é feita uma contagem das localidades presentes na lista usando a classe `*Counter*` da biblioteca *Python*. Esta classe cria um dicionário onde as chaves são as localidades e os valores são as contagens de ocorrência. O programa seleciona as 30 localidades mais comuns e salva-com as respetivas contagens num arquivo CSV, onde cada linha contém o nome da localidade e sua contagem de ocorrências.

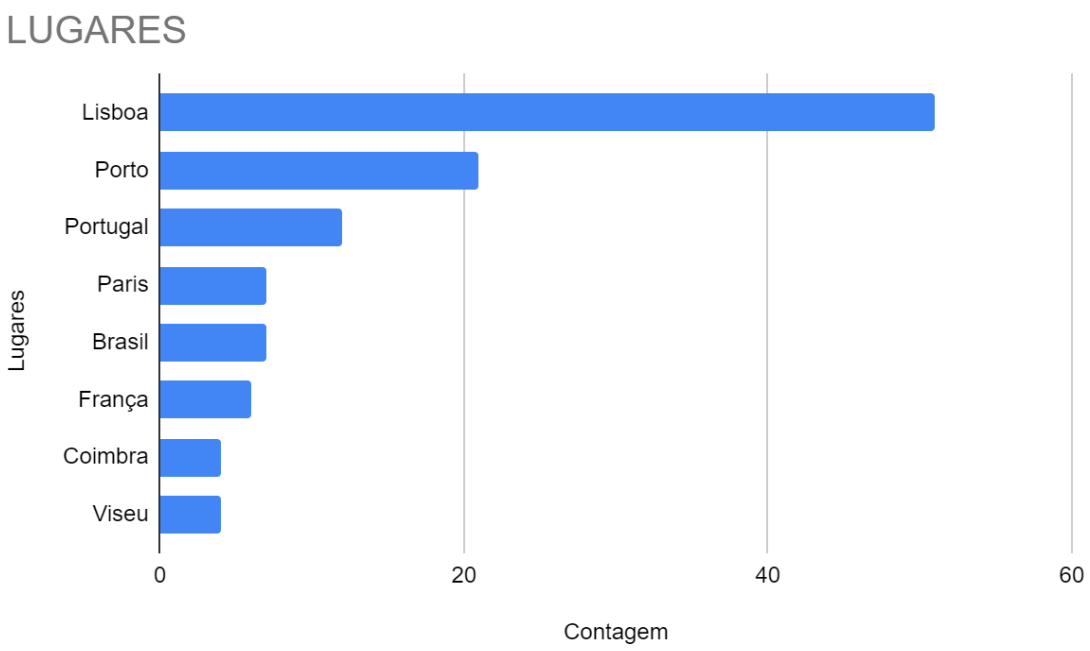
Assim:

	Localidade	Contagem
1	Lisboa	51
2	Virgínia	21
3	Porto	18
4	Laura	11
5	Esteia	11
6	Sei	9
7	Portugal	8
8	Tenho	8
9	Olha	7
10	Paris	7
11	Brasil	7
12	França	6
13	Perpétua	6
14	Vamos	6
15	Vou	5
16	Júlia	5
17	Queres	5
18	Cassilda	5
19	Céu	4
20	em Portugal	4
21	Estás	4
22	Coimbra	4
23	Providência	4
24	Carlos	4
25	Minha	4
26	Viseu	4
27	Cassilda Arcourt	4
28	Filomena	4
29	Colégio da Form	3
30	Queira	3
31		

Mais uma vez, foi feita uma limpeza manual dos dados. Tem-se como resultado:

	Localidade	Contagem
1	Lisboa	51
2	Porto	21
3	Portugal	12
4	Paris	7
5	Brasil	7
6	França	6
7	Coimbra	4
8	Viseu	4
9		

Como proposta de visualização, foi criado o gráfico seguinte:



Conclui-se que Lisboa é o lugar mais mencionado.

2. Amor de Salvação

Para esta obra, foi analisado e extraídos os verbos, as localidades, as palavras-chave e feita uma análise de sentimento por cada capítulo do livro. O documento de texto que contem a obra, foi utilizado sempre em formato de *one sentence per line.*

2.1 Verbos

Para a obra Amor de Salvação, foi relevante extrair um top dos 20 verbos que surgem com mais frequência na obra. Então, foi criado o seguinte programa:

```python
import os
import csv
from collections import Counter

with open('amor_de_salvacao1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

dicionario = dict()

for token in doc:
    if token.pos_ == "VERB":
        if token.lemma_ in dicionario.keys():
            dicionario[token.lemma_] += 1
        else:
            dicionario[token.lemma_] = 1

verb_counts = Counter(dicionario)
top20_verbs = verb_counts.most_common(20)

print(top20_verbs)

with open('VERBOSamor_de_salvacao.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Verbo', 'Contagem'])  # Adiciona o cabeçalho das colunas

    for verb, count in top20_verbs:
        writer.writerow([verb, count])
```

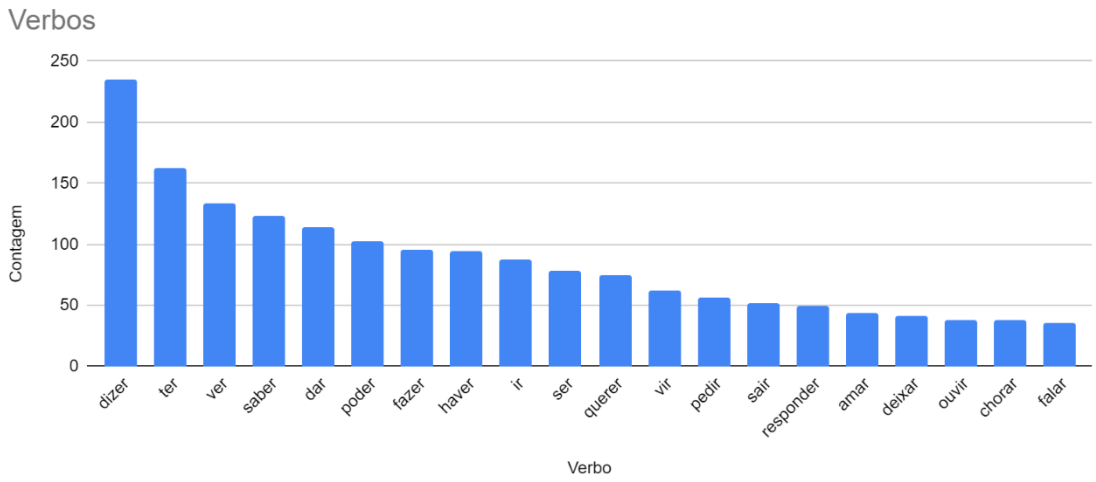
O código realiza uma análise de verbos usando a biblioteca *Spacy*, com o modelo de linguagem em português. Ele conta a frequência de ocorrência de cada verbo no texto e imprime os 20 verbos mais frequentes. Além disso, os resultados são salvos em um arquivo CSV com duas colunas: "Verbo" e "Contagem". Em seguida, um dicionário vazio é criado para armazenar os verbos e as suas contagens.

O *loop* percorre cada *token* no documento e verifica se esse *token* é um verbo. Se for, o lema do verbo é adicionado ao dicionário, aumentando sua contagem de ocorrência. Se o verbo já estiver presente no dicionário, sua contagem é atualizada. Caso contrário, o verbo é adicionado ao dicionário com uma contagem inicial de 1.

A biblioteca `*Counter*` é usada para contar a frequência dos verbos no dicionário. Os 20 verbos mais comuns são selecionados usando o método `*most_common*(20)` e armazenados na lista `*top20_verbs*`. A lista extraída é a seguinte:

	Verbo	Contagem
2	dizer	235
3	ter	162
4	ver	134
5	saber	123
6	dar	114
7	poder	103
8	fazer	96
9	haver	95
10	ir	88
11	ser	78
12	querer	75
13	vir	62
14	pedir	56
15	sair	52
16	responder	50
17	amar	44
18	deixar	42
19	ouvir	38
20	chorar	38
21	falar	36

Neste caso, não efetuei qualquer tipo de limpeza uma vez que os dados se revelaram bastante precisos. Então, como proposta de visualização creio que um gráfico de barras acabou por ser a melhor opção:



Como era de esperar, os verbos mais comuns são os verbos copulativos. Porém, é de realçar o surgimento dos verbos amar e chorar no top 20 dos verbos que mais surgem na obra.

2.2 Análise de Sentimento

Para fazer a analise de sentimento de cada capítulo, criei o seguinte programa *Python,* que utiliza a biblioteca *NLTK (Natural Language Toolkit*) para realizar uma análise de sentimentos num dado texto. Ele utiliza o módulo `*SentimentIntensityAnalyzer*` do *NLTK*, que é responsável por atribuir pontuações de sentimento a diferentes partes do texto.

```python
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer

nltk.download('vader_lexicon')

sia = SentimentIntensityAnalyzer()

caminho_arquivo = "amor_de_salvacao1spl.txt"

with open(caminho_arquivo, 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

capitulos = texto.split('# Capítulo')

for i, capitulo in enumerate(capitulos[1:]):
    sentiment = sia.polarity_scores(capitulo)
    print(f"Sentimento do Capítulo {i+1}: {sentiment}")
```

Primeiro, o código importa as bibliotecas necessárias e faz o download do léxico *Vader do NLTK*, que é usado para análise de sentimentos. Em seguida, é criada uma instância do `*SentimentIntensityAnalyzer*`.

Depois, o código lê o arquivo *txt* da obra *A mulher fatal* no formato de *one sentence per line* e armazena-o numa variável. O texto é dividido em capítulos usando um marcador específico (no caso, foi colocada um # em cada capítulo no documento de texto). Através de um *loop,* cada capítulo é analisado usando o `*SentimentIntensityAnalyzer*`, que retorna uma pontuação de sentimento para aquele capítulo.

Em resumo, o código realiza a análise de sentimentos no texto, dividindo-o em capítulos e calculando a pontuação de sentimento para cada um deles. Os resultados no terminal são:

```
Sentimento do Capítulo 1: {'neg': 0.016, 'neu': 0.972, 'pos': 0.012, 'compound': -0.5897}
Sentimento do Capítulo 2: {'neg': 0.016, 'neu': 0.97, 'pos': 0.013, 'compound': -0.4035}
Sentimento do Capítulo 3: {'neg': 0.02, 'neu': 0.98, 'pos': 0.0, 'compound': -0.9157}
Sentimento do Capítulo 4: {'neg': 0.02, 'neu': 0.957, 'pos': 0.023, 'compound': 0.8377}
Sentimento do Capítulo 5: {'neg': 0.018, 'neu': 0.972, 'pos': 0.01, 'compound': -0.86}
Sentimento do Capítulo 6: {'neg': 0.017, 'neu': 0.972, 'pos': 0.011, 'compound': -0.7067}
Sentimento do Capítulo 7: {'neg': 0.023, 'neu': 0.961, 'pos': 0.016, 'compound': -0.8035}
Sentimento do Capítulo 8: {'neg': 0.006, 'neu': 0.987, 'pos': 0.007, 'compound': 0.7374}
Sentimento do Capítulo 20: {'neg': 0.022, 'neu': 0.971, 'pos': 0.006, 'compound': -0.972}
Sentimento do Capítulo 21: {'neg': 0.02, 'neu': 0.972, 'pos': 0.008, 'compound': -0.78}
Sentimento do Capítulo 22: {'neg': 0.022, 'neu': 0.972, 'pos': 0.006, 'compound': -0.954}
Sentimento do Capítulo 23: {'neg': 0.012, 'neu': 0.971, 'pos': 0.017, 'compound': 0.9655}
Sentimento do Capítulo 10: {'neg': 0.014, 'neu': 0.973, 'pos': 0.012, 'compound': 0.814}
Sentimento do Capítulo 11: {'neg': 0.015, 'neu': 0.972, 'pos': 0.013, 'compound': 0.8492}
Sentimento do Capítulo 12: {'neg': 0.038, 'neu': 0.957, 'pos': 0.005, 'compound': -0.9686}
Sentimento do Capítulo 13: {'neg': 0.012, 'neu': 0.984, 'pos': 0.004, 'compound': -0.9453}
Sentimento do Capítulo 14: {'neg': 0.014, 'neu': 0.982, 'pos': 0.004, 'compound': -0.9547}
Sentimento do Capítulo 15: {'neg': 0.022, 'neu': 0.971, 'pos': 0.006, 'compound': -0.9839}
Sentimento do Capítulo 16: {'neg': 0.027, 'neu': 0.965, 'pos': 0.009, 'compound': -0.9796}
Sentimento do Capítulo 17: {'neg': 0.017, 'neu': 0.98, 'pos': 0.004, 'compound': -0.9706}
Sentimento do Capítulo 18: {'neg': 0.025, 'neu': 0.975, 'pos': 0.0, 'compound': -0.982}
Sentimento do Capítulo 19: {'neg': 0.011, 'neu': 0.972, 'pos': 0.017, 'compound': 0.9533}
Sentimento do Capítulo 20: {'neg': 0.022, 'neu': 0.971, 'pos': 0.006, 'compound': -0.972}
Sentimento do Capítulo 21: {'neg': 0.02, 'neu': 0.972, 'pos': 0.008, 'compound': -0.78}
Sentimento do Capítulo 22: {'neg': 0.022, 'neu': 0.972, 'pos': 0.006, 'compound': -0.954}
Sentimento do Capítulo 23: {'neg': 0.012, 'neu': 0.971, 'pos': 0.017, 'compound': 0.9655}
PS C:\Users\ines2\OneDrive\Ambiente de Trabalho\avd2023\trabalho final avd>
```

Para construir uma proposta de visualização para a análise de sentimento do livro Amor de Salvação, utilizou-se o mesmo método que se tinha utilizado no livro anterior. Então, o programa foi o seguinte:

```python
import matplotlib.pyplot as plt

pontuacoes = [
    {'neg': 0.016, 'neu': 0.972, 'pos': 0.012, 'compound': -0.5897},
    {'neg': 0.016, 'neu': 0.97, 'pos': 0.013, 'compound': -0.4035},
    {'neg': 0.02, 'neu': 0.98, 'pos': 0.0, 'compound': -0.9157},
    {'neg': 0.02, 'neu': 0.957, 'pos': 0.023, 'compound': 0.8377},
    {'neg': 0.018, 'neu': 0.972, 'pos': 0.01, 'compound': -0.86},
    {'neg': 0.017, 'neu': 0.972, 'pos': 0.011, 'compound': -0.7067},
    {'neg': 0.023, 'neu': 0.961, 'pos': 0.016, 'compound': -0.8035},
    {'neg': 0.006, 'neu': 0.987, 'pos': 0.007, 'compound': 0.7374},
    {'neg': 0.015, 'neu': 0.976, 'pos': 0.009, 'compound': -0.6935},
    {'neg': 0.014, 'neu': 0.973, 'pos': 0.012, 'compound': 0.814},
    {'neg': 0.015, 'neu': 0.972, 'pos': 0.013, 'compound': 0.8492},
    {'neg': 0.038, 'neu': 0.957, 'pos': 0.005, 'compound': -0.9686},
    {'neg': 0.012, 'neu': 0.984, 'pos': 0.004, 'compound': -0.9453},
    {'neg': 0.014, 'neu': 0.982, 'pos': 0.004, 'compound': -0.9547},
    {'neg': 0.022, 'neu': 0.971, 'pos': 0.006, 'compound': -0.9839},
    {'neg': 0.027, 'neu': 0.965, 'pos': 0.009, 'compound': -0.9796},
    {'neg': 0.017, 'neu': 0.98, 'pos': 0.004, 'compound': -0.9706},
    {'neg': 0.025, 'neu': 0.975, 'pos': 0.0, 'compound': -0.982},
    {'neg': 0.011, 'neu': 0.972, 'pos': 0.017, 'compound': 0.9533},
    {'neg': 0.022, 'neu': 0.971, 'pos': 0.006, 'compound': -0.972},
    {'neg': 0.02, 'neu': 0.972, 'pos': 0.008, 'compound': -0.78},
    {'neg': 0.022, 'neu': 0.972, 'pos': 0.006, 'compound': -0.954},
    {'neg': 0.012, 'neu': 0.971, 'pos': 0.017, 'compound': 0.9655}
]

valores_compound = [sentimento['compound'] for sentimento in pontuacoes]

rotulos = [f"Capítulo {i+1}" for i in range(len(pontuacoes))]
```

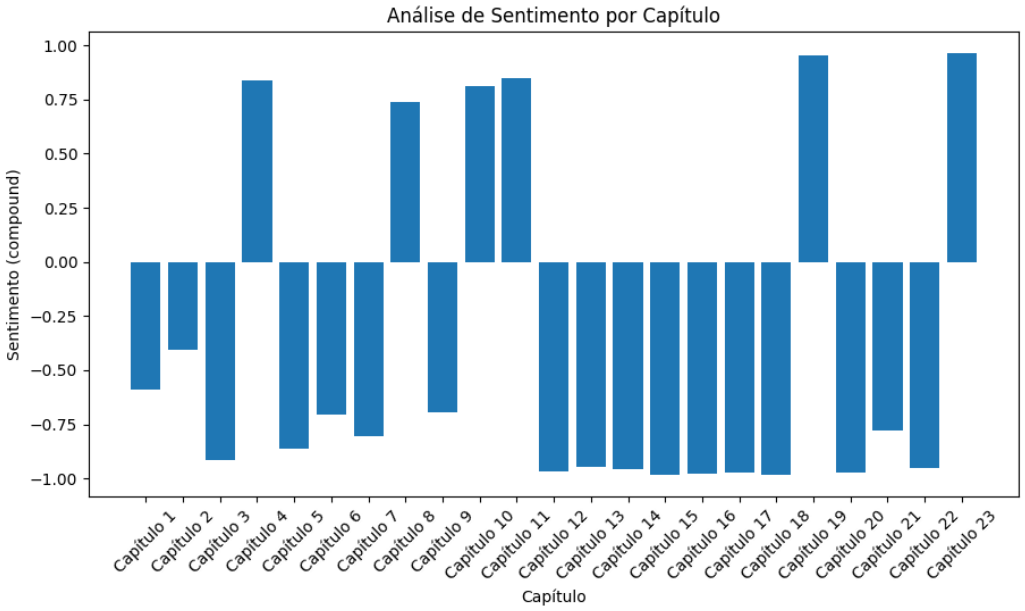
```python
    {'neg': 0.02, 'neu': 0.972, 'pos': 0.008, 'compound': -0.78},
    {'neg': 0.022, 'neu': 0.972, 'pos': 0.006, 'compound': -0.954},
    {'neg': 0.012, 'neu': 0.971, 'pos': 0.017, 'compound': 0.9655}
]

valores_compound = [sentimento['compound'] for sentimento in pontuacoes]

rotulos = [f"Capítulo {i+1}" for i in range(len(pontuacoes))]

plt.figure(figsize=(10, 6))
plt.bar(rotulos, valores_compound)
plt.xlabel('Capítulo')
plt.ylabel('Sentimento (compound)')
plt.title('Análise de Sentimento por Capítulo')
plt.xticks(rotation=45)
plt.show()
```

O código em questão utiliza a biblioteca Matplotlib em conjunto com os resultados da análise de sentimento por capítulos para criar um gráfico de barras. O objetivo é visualizar a variação do sentimento ao longo dos capítulos de uma obra. Inicialmente, os resultados da análise de sentimento por capítulo são armazenados numa lista chamada `*pontuacoes*`. Cada elemento dessa lista é um dicionário que contém as pontuações de negatividade (*'neg'*), neutralidade (*'neu'*), positividade (*'pos'*) e a pontuação geral do sentimento (*'compound'*) para um capítulo específico.

O próximo passo consiste em extrair os valores do 'compound' de cada dicionário da lista `pontuacoes`. Esses valores são armazenados numa lista separada chamada

`valores_compound`. Esta lista representa a medida agregada do sentimento para cada capítulo. A lista `rotulos` é criada para armazenar os rótulos dos capítulos. Cada rótulo é definido como "Capítulo X", onde X é o número do capítulo correspondente. Com todos os dados necessários preparados, o código utiliza a biblioteca Matplotlib para criar o gráfico de barras. É criada uma figura com um tamanho específico utilizando a função `plt.figure(figsize=(10, 6))`. Em seguida, a função `plt.bar()` é utilizada para criar as barras do gráfico. Os rótulos dos capítulos são exibidos no eixo x e os valores do compound são representados no eixo y. Algumas funções do Matplotlib, como `plt.xlabel()`, `plt.ylabel()`, `plt.title()` e `plt.xticks(rotation=45)`, são utilizadas para adicionar rótulos aos eixos e ao gráfico em geral, além de fazer uma rotação de 45 graus nos rótulos dos capítulos para facilitar a leitura. Por fim, o gráfico é exibido na tela utilizando a função `plt.show()`.

Assim, o gráfico:



Para concluir, é maior a quantidade de capítulos com sentimento negativo do que positivo, embora o final seja positivo.

2.3 Palavras-Chave

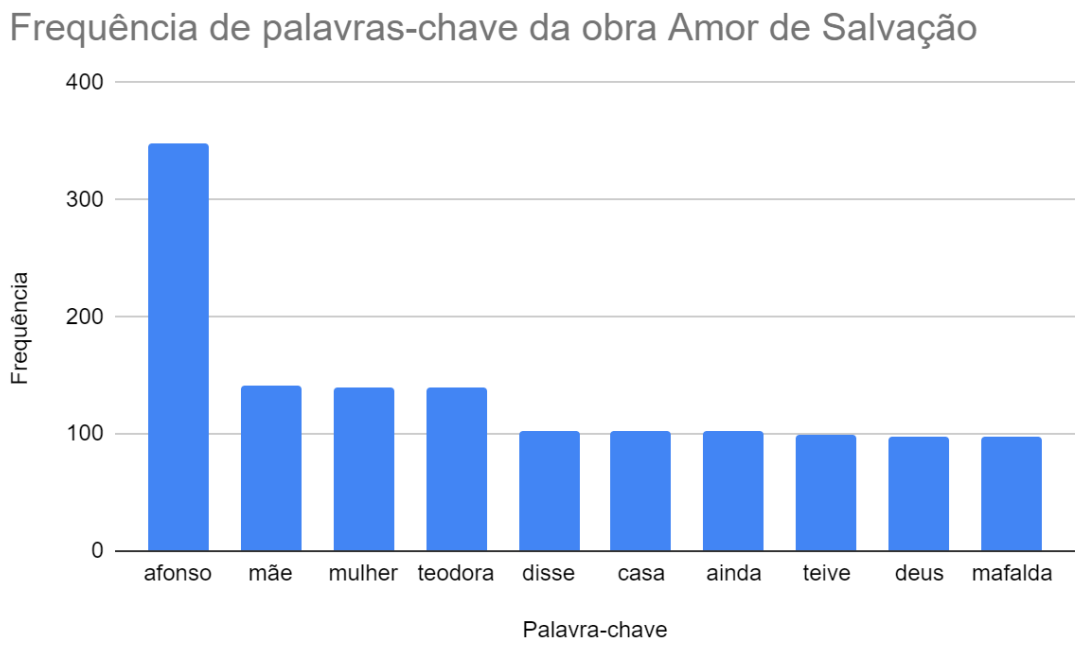
Para obra Amor de Salvação, foi relevante fazer uma extração de palavras-chave da obra. Assim, foi criado o seguinte programa em python:

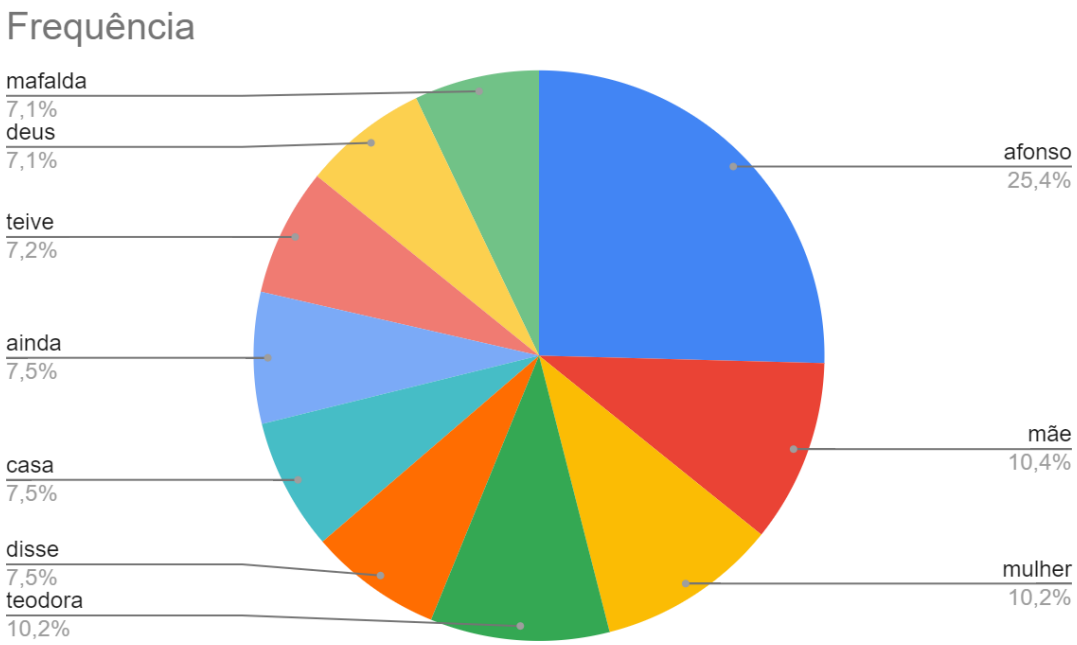
```python
import csv
from collections import Counter
import string
import nltk
from nltk.corpus import stopwords

nltk.download('stopwords')

stop_words = set(stopwords.words('portuguese'))

def extrair_palavras_chave(texto):
    texto = texto.translate(str.maketrans('', '', string.punctuation)).lower()
    palavras = texto.split()
    palavras = [palavra for palavra in palavras if palavra not in stop_words]
    contagem_palavras = Counter(palavras)
    return contagem_palavras

with open('amor_de_salvacao1spl.txt', 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

palavras_chave = extrair_palavras_chave(texto)
e
top_palavras_chave = palavras_chave.most_common(10)

nome_arquivo_csv = 'palavras_chave.csv'

with open(nome_arquivo_csv, 'w', newline='', encoding='utf-8') as arquivo_csv:
    writer = csv.writer(arquivo_csv)
    writer.writerow(['Palavra-chave', 'Frequência'])  # Cabeçalho das colunas

    for palavra, frequencia in top_palavras_chave:
        writer.writerow([palavra, frequencia])
```

Primeiro, são importadas as bibliotecas necessárias para o processamento de texto, contagem de palavras e manipulação de arquivos *csv*. Depois, faz-se uma baixa das *stop-words*, em português, através da NLTK. A função *extrair_palavras_chave* recebe o texto como entrada, onde dentro da mesma, será removida a pontuação de texto e convertidas as maiúsculas para minúsculas. Depois de o texto dividido e desprovido de *stop-words,* conta-se a frequência de ocorrência através da biblioteca Counter. Depois de extraídas as 10 palavras que mais surgem no texto, cria-se um documento csv com os resultados, sendo eles:

	Palavra-chave	Frequência
2	afonso	348
3	mãe	142
4	mulher	140
5	teodora	139
6	disse	103
7	casa	102
8	ainda	102
9	teive	99
10	deus	97
11	mafalda	97
12		
13		

Mais uma vez, não foi necessário recorrer a limpeza dos dados. Como proposta de visualização, foi criado um gráfico de barras e um gráfico circular.





O nome Afonso é, com destaque, a palavra mais utilizada.

2.4 Lugares

Por fim, para esta obra, foi pertinente analisar os nomes de lugares que surgem na obra, bem como calcular a sua frequência. Assim, foi utilizado o mesmo programa que em 1.4:

```python
import spacy
import os
import csv

with open('amor_de_salvacao1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

locations = []

for ent in doc.ents:
    if ent.label_ == "LOC":
        locations.append(ent.text)

from collections import Counter

locations_counts = Counter(locations)

top30_locations = locations_counts.most_common(30)

print(top30_locations)

with open('LUGARESamor_de_salvacao.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Localidade', 'Contagem'])
    for location, count in top30_locations:
        writer.writerow([location, count])
```

Depois de carregar o documento de texto com a obra Amor de Salvação, em formato de one sentence per line, e o armazenar numa variável, a biblioteca SpaCy é carregada com um modelo de processamento de texto em português. O texto é analisado usando o modelo carregado, e o código procura todas as entidades que representam localidades. Caso uma entidade seja identificada como uma localidade, ela é adicionada a uma lista. Em seguida, é feita uma contagem das localidades presentes na lista usando a classe `Counter` da biblioteca Python. Esta classe cria um dicionário onde as chaves são as localidades e os valores são as contagens de ocorrência. O programa seleciona as 30 localidades mais comuns e salva-com as respetivas contagens num arquivo CSV, onde cada linha contém o nome da localidade e sua contagem de ocorrências.

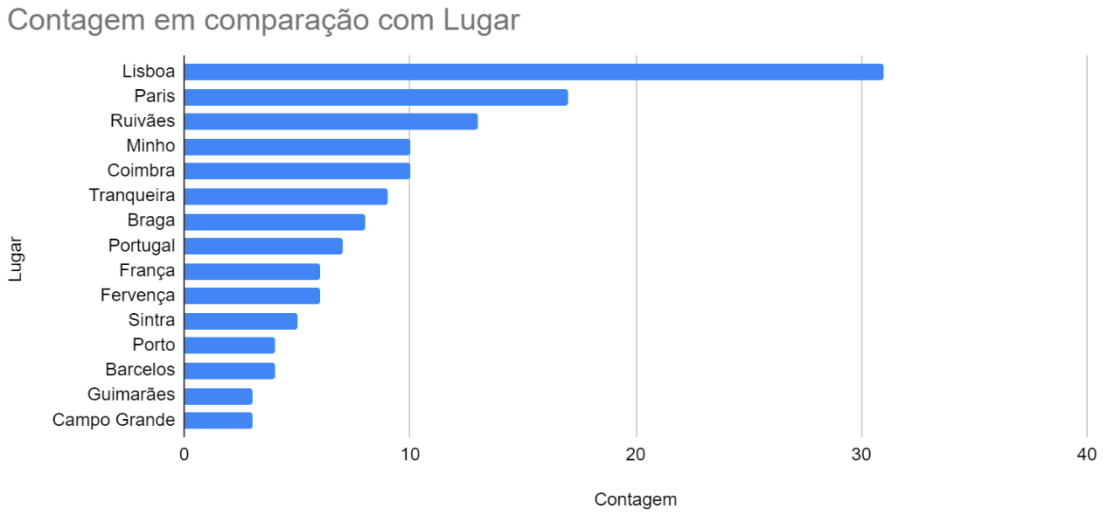
O documento csv, então:

	Localidade	Contagem
2	Palmira	33
3	Lisboa	31
4	Paris	17
5	Céu	16
6	Ruivães	13
7	Eleutério	13
8	Minho	10
9	Coimbra	10
10	Tranqueira	9
11	Braga	8
12	Portugal	7
13	Terra	7
14	França	6
15	Fervença	6
16	Eleutério Romão	6
17	Vamos	6
18	Estás	5
19	Minha	5
20	Sintra	5
21	Entrei	4
22	Porto	4
23	Queres	4
24	Barcelos	4
25	Guimarães	3
26	Campo Grande	3
27	Afonso de Teive	3
28	Criador	3
29	Ursulinas	3
30	Dizia	3
31	Romão	3

Procedeu-se a uma limpeza manual dos dados:

	Lugar	Contagem
1	Lugar	Contagem
2	Lisboa	31
3	Paris	17
4	Ruivães	13
5	Minho	10
6	Coimbra	10
7	Tranqueira	9
8	Braga	8
9	Portugal	7
10	França	6
11	Fervença	6
12	Sintra	5
13	Porto	4
14	Barcelos	4
15	Guimarães	3
16	Campo Grande	3
17		

Criou-se a seguinte proposta de visualização:



Lisboa é então o lugar mais mencionado na obra.

3. O Romance de Um Homem Rico

Para esta obra, foi analisado e extraídos os lemas, as entidades, os adjetivos e feita uma análise de sentimento por cada capítulo do livro. O documento de texto que contem a obra, foi utilizado sempre em formato de *one sentence per line.*

3.1 Entidades

Para extrair as entidades da obra O Romance Dum Homem Rico, utilizei o seguinte programa Python, utilizando a biblioteca spaCy:

```python
import spacy
import os
import csv

with open('o_romance_dum_homem_rico1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

characters = []

for ent in doc.ents:
    if ent.label_ == "PER":
        characters.append(ent.text)

from collections import Counter

characters_counts = Counter(characters)

top30_characters = characters_counts.most_common(30)

print(top30_characters)

with open('Entidades_oromancedumhomemrico.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Personagem', 'Contagem'])

    for character, count in top30_characters:
        writer.writerow([character, count])
```

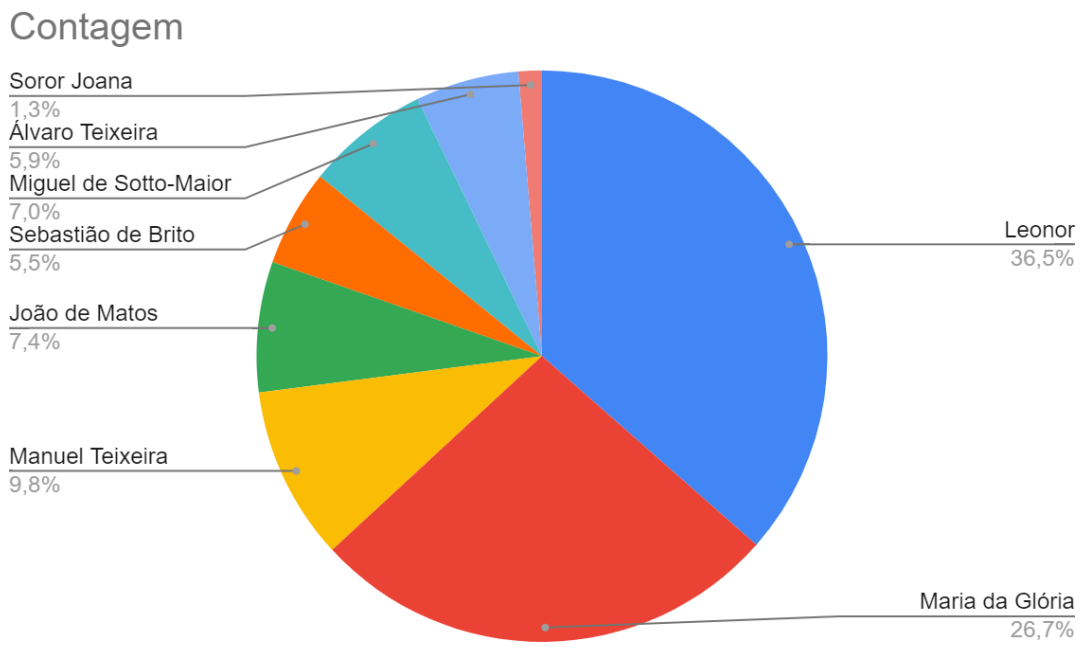
Este programa extrairá um Top 30 personagens, juntamente com o número de vezes que cada uma surge na obra. O programa cria um documento csv com os resultados:

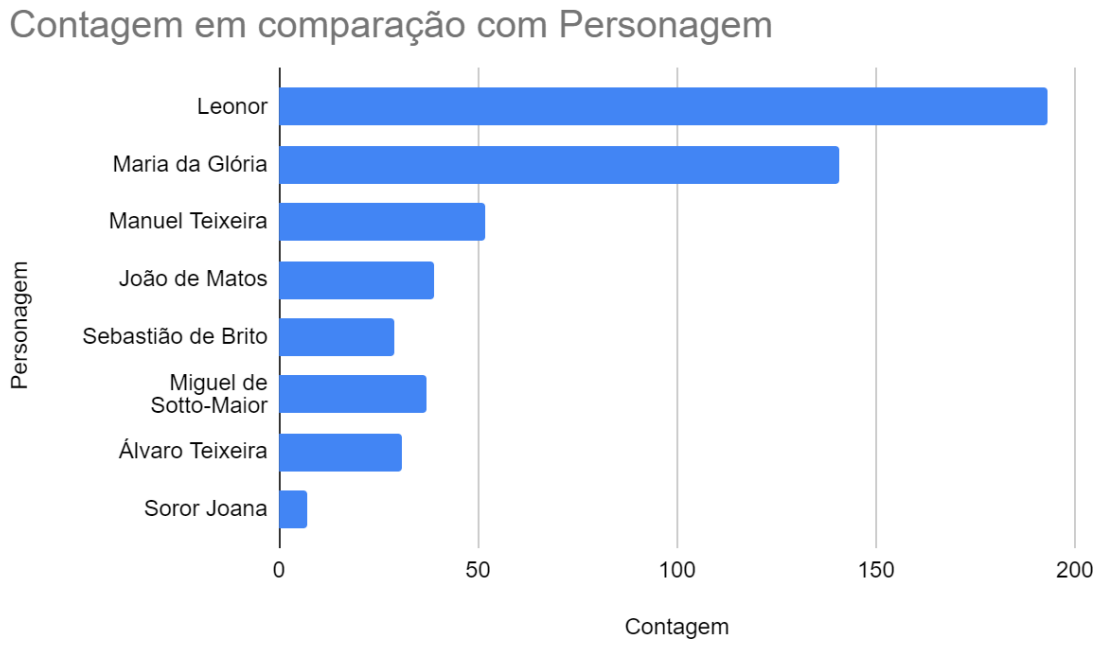
Personagem	Contagem
Leonor	189
Maria da Glória	96
—	49
Deus	49
Manuel Teixeira	48
João de Matos	39
Maria	38
Sebastião de Br	29
Miguel de Sotto-	26
Senhor	13
Vossa Senhoria	11
Álvaro	11
Vossa Excelênci	10
Fui	8
Álvaro Teixeira	7
— Eu	7
Soror Joana	7
Maria da Glória.	7
Ferro	6
Sotto-Maior	6
Padre Álvaro	5
Tenho	5
Tinham	5
Miguel	5
Álvaro Teixeira c	4
padre Álvaro	4
Abriu	4
Manuel Teixeira	4
Leonor	4
Espero	3

Depois de uma análise manual dos dados, obteve-se:

	Personagem	Contagem
1		
2	Leonor	193
3	Maria da Glória	141
4	Manuel Teixeira	52
5	João de Matos	39
6	Sebastião de Bri	29
7	Miguel de Sotto-	37
8	Álvaro Teixeira	31
9	Soror Joana	7
10		
11		

Como proposta de visualização, neste caso, foram criados dois gráficos. Um circular e outro de barras.





Leonor é, portanto, a personagem mais mencionada no livro.

3.2 Adjetivos

Em segundo lugar, para esta obra, foi analisado o top 20 adjetivos mais utilizados, através do seguinte código python:

```python
import spacy
import csv
from collections import Counter

nlp = spacy.load('pt_core_news_sm')

with open('o_romance_dum_homem_rico1spl.txt', 'r', encoding='utf-8') as file:
    text = file.read()

doc = nlp(text)

adjectives = Counter()

for token in doc:
    if token.pos_ == 'ADJ':
        adjectives[token.lemma_] += 1

top20_adjectives = adjectives.most_common(20)

with open('adjetivos_oromancedumhomemrico.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Adjetivo', 'Frequência'])
    writer.writerows(top20_adjectives)

```

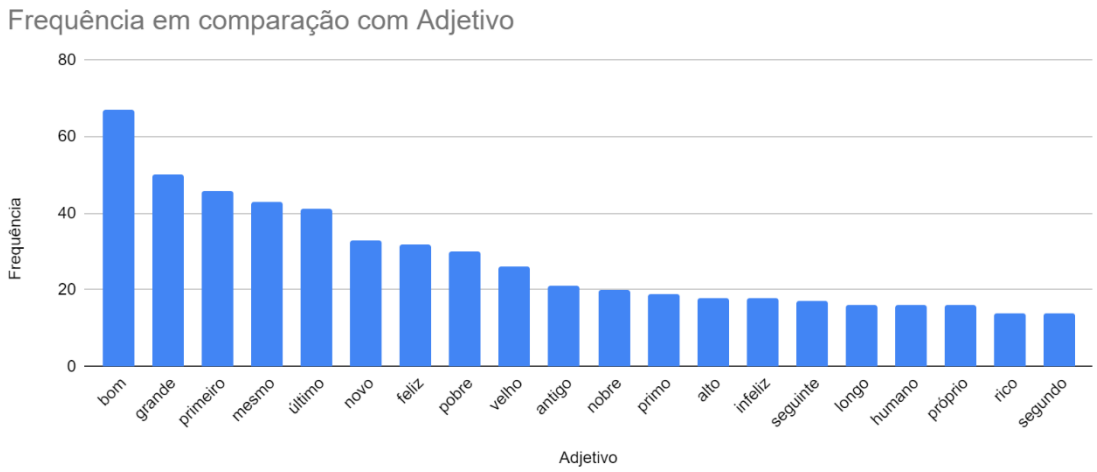
O programa utiliza a biblioteca *Spacy* para extrair os 20 adjetivos mais frequentes da obra em formato TXT. Primeiro, o texto é lido a partir de um arquivo TXT e processado

pelo *Spacy*. Em seguida, os adjetivos são identificados e contados. Os 20 adjetivos mais comuns são armazenados num arquivo *CSV*.

Os resultados obtidos foram:

	Adjetivo	Frequência
2	bom	67
3	grande	50
4	primeiro	46
5	mesmo	43
6	último	41
7	novo	33
8	feliz	32
9	pobre	30
10	velho	26
11	antigo	21
12	nobre	20
13	primo	19
14	alto	18
15	infeliz	18
16	seguinte	17
17	longo	16
18	humano	16
19	próprio	16
20	rico	14
21	segundo	14

Uma vez que não foi necessária qualquer tipo de limpeza aos dados, cria-se a seguinte proposta de visualização:



Conclui-se que *bom* é o adjetivo mais comum na obra.

3.3 Lemas

Para extrair um top 50 lemas da obra O Romance Dum Homem Rico, foi criado o seguinte programa:

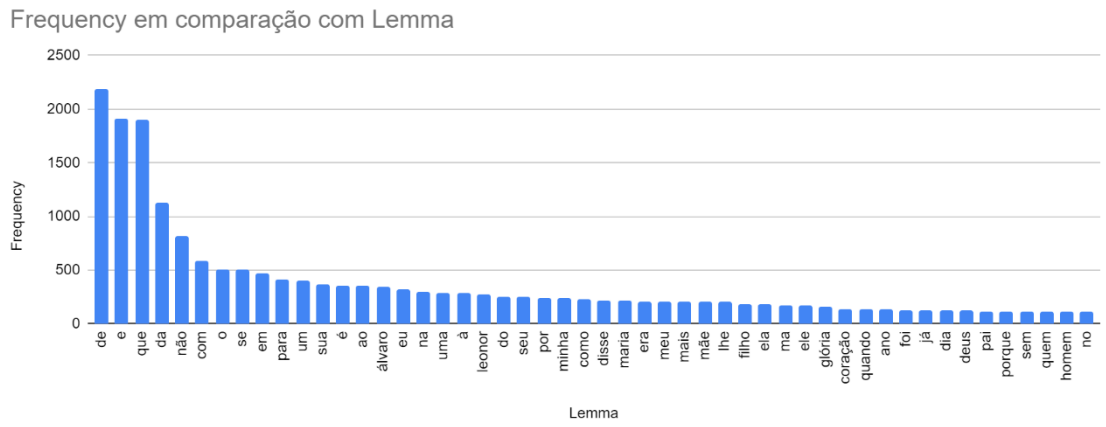
```python
import nltk
from nltk import FreqDist
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import csv

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

def extract_lemmas(text):
    lemmatizer = WordNetLemmatizer()
    stop_words = set(stopwords.words('english'))

    tokens = nltk.word_tokenize(text)
    lemmas = [lemmatizer.lemmatize(token.lower()) for token in tokens if token.isalnum() and token.lower() not in stop_words]

    return lemmas

def save_lemmas_to_csv(lemmas, output_file):
    fdist = FreqDist(lemmas)
    top_lemmas = fdist.most_common(50)

    with open(output_file, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(['Lemma', 'Frequency'])
        for lemma, freq in top_lemmas:
            writer.writerow([lemma, freq])

input_file = 'o_romance_dum_homem_rico1spl.txt'

output_file = 'TOP50oromancedumhomemrico.csv'

with open(input_file, 'r') as file:
    text = file.read()

lemmas = extract_lemmas(text)

save_lemmas_to_csv(lemmas, output_file)
```

O programa utiliza o NLTK para extrair os lemas da obra, carregada em formato de texto, bem como calcula a frequência que cada um aparece no texto. Assim, os 50 lemas mais frequentes são guardados num arquivo novo em formato csv. Tem-se:

O programa utiliza o NLTK para extrair os lemas da obra, carregada em formato de texto, bem como calcula a frequência que cada um aparece no texto. Assim, os 50 lemas mais frequentes são guardados num arquivo novo em formato csv. Tem-se:

	Lemma	Frequency
1	de	2183
2	e	1907
3	que	1895
4	da	1124
5	não	817
6	com	586
7	o	503
8	se	499
9	em	474
10	para	408
11	um	397
12	sua	362
13	é	355
14	ao	352
15	álvaro	340
16	eu	323
17	na	292
18	uma	287
19	à	283
20	leonor	278
21	do	252
22	seu	249
23	por	243
24	minha	241
25	como	224
26	disse	216
27	maria	213
28	era	210
29	meu	210
30	mais	205
31	mãe	201
32	lhe	200
33	filho	186
34	ela	181
35	ma	175
36	ele	172
37	glória	158
38	coração	141
39	quando	140
40	ano	139
41	foi	126
42	já	125
43	dia	125
44	deus	124
45	pai	113
46	porque	113
47	sem	112
48	quem	108
49	homem	107
50	no	107
51		

Uma vez que todos os resultados são lemas, não se realizou qualquer tipo de limpeza. Como proposta de visualização:



Como esperado, as preposições e conjunções são os lemas mais presentes no texto.

3.4 Análise de Sentimento

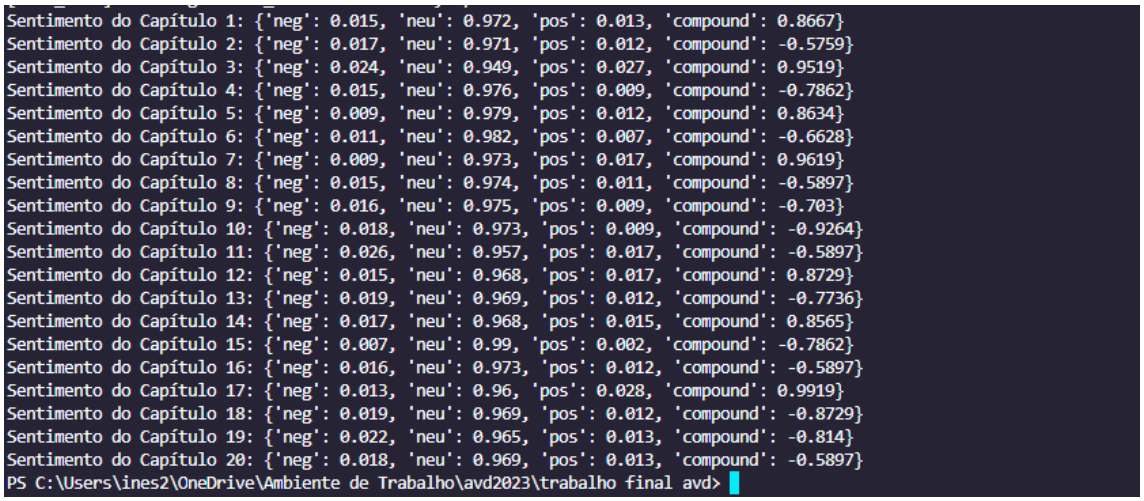
Para fazer a analise de sentimento de cada capítulo desta obra, foi utilizado o seguinte programa *Python*, que utiliza a biblioteca *NLTK (Natural Language Toolkit*) para realizar uma análise de sentimentos num dado texto. Ele utiliza o módulo `*SentimentIntensityAnalyzer*` do *NLTK*, que é responsável por atribuir pontuações de sentimento a diferentes partes do texto.

```python
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer

nltk.download('vader_lexicon')

sia = SentimentIntensityAnalyzer()

caminho_arquivo = "o_romance_dum_homem_rico1spl.txt"

with open(caminho_arquivo, 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

capitulos = texto.split('# Capítulo')

for i, capitulo in enumerate(capitulos[1:]):
    sentiment = sia.polarity_scores(capitulo)
    print(f"Sentimento do Capítulo {i+1}: {sentiment}")

```

Primeiro, o código importa as bibliotecas necessárias e faz o download do léxico Vader do NLTK, que é usado para análise de sentimentos. Em seguida, é criada uma instância do `SentimentIntensityAnalyzer`.

Depois, o código lê o arquivo txt da obra A mulher fatal no formato de one sentence per line e armazena-o numa variável. O texto é dividido em capítulos usando um marcador específico (no caso, foi colocada um # em cada capítulo no documento de texto). Através de um loop, cada capítulo é analisado usando o `SentimentIntensityAnalyzer`, que retorna uma pontuação de sentimento para aquele capítulo.

Em resumo, o código realiza a análise de sentimentos no texto, dividindo-o em capítulos e calculando a pontuação de sentimento para cada um deles. Os resultados no terminal são:



Para uma análise mais profunda, tem-se:

- Capítulo 1: Sentimento positivo moderado (compound: 0.8667)
- Capítulo 2: Sentimento negativo moderado (compound: -0.5759)
- Capítulo 3: Sentimento positivo forte (compound: 0.9519)
- Capítulo 4: Sentimento negativo moderado (compound: -0.7862)
- Capítulo 5: Sentimento positivo moderado (compound: 0.8634)
- Capítulo 6: Sentimento negativo moderado (compound: -0.6628)
- Capítulo 7: Sentimento positivo forte (compound: 0.9619)
- Capítulo 8: Sentimento negativo moderado (compound: -0.5897)
- Capítulo 9: Sentimento negativo moderado (compound: -0.703)
- Capítulo 10: Sentimento negativo forte (compound: -0.9264)
- Capítulo 11: Sentimento negativo moderado (compound: -0.5897)
- Capítulo 12: Sentimento positivo moderado (compound: 0.8729)
- Capítulo 13: Sentimento negativo moderado (compound: -0.7736)
- Capítulo 14: Sentimento positivo moderado (compound: 0.8565)
- Capítulo 15: Sentimento negativo moderado (compound: -0.7862)
- Capítulo 16: Sentimento negativo moderado (compound: -0.5897)
- Capítulo 17: Sentimento positivo forte (compound: 0.9919)
- Capítulo 18: Sentimento negativo moderado (compound: -0.8729)
- Capítulo 19: Sentimento negativo moderado (compound: -0.814)
- Capítulo 20: Sentimento negativo moderado (compound: -0.5897)

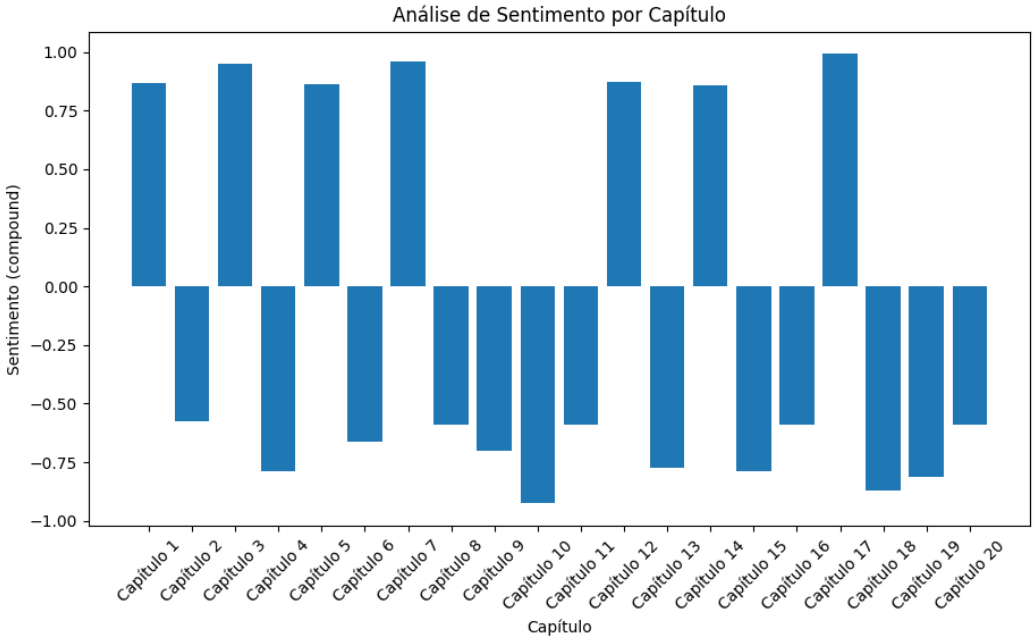
Essa análise é baseada no valor do atributo *'compound'* de cada capítulo. O valor composto (*compound*) varia de -1 a 1, representando um sentimento negativo forte a um sentimento positivo forte, respetivamente. Valores próximos de 0 indicam sentimentos neutros.

Pode observar-se que a maioria dos capítulos apresenta sentimentos moderados, sendo alguns positivos e outros negativos. Alguns capítulos específicos destacam-se com sentimentos positivos fortes (capítulo 3, capítulo 7 e capítulo 17), enquanto outros têm sentimentos negativos mais intensos (capítulo 10 e capítulo 18).

Em seguida, para criar uma proposta de visualização, foi feito um programa para criar um gráfico de barras:

```python
import matplotlib.pyplot as plt

pontuacoes = [
    {'neg': 0.015, 'neu': 0.972, 'pos': 0.013, 'compound': 0.8667},
    {'neg': 0.017, 'neu': 0.971, 'pos': 0.012, 'compound': -0.5759},
    {'neg': 0.024, 'neu': 0.949, 'pos': 0.027, 'compound': 0.9519},
    {'neg': 0.015, 'neu': 0.976, 'pos': 0.009, 'compound': -0.7862},
    {'neg': 0.009, 'neu': 0.979, 'pos': 0.012, 'compound': 0.8634},
    {'neg': 0.011, 'neu': 0.982, 'pos': 0.007, 'compound': -0.6628},
    {'neg': 0.009, 'neu': 0.973, 'pos': 0.017, 'compound': 0.9619},
    {'neg': 0.015, 'neu': 0.974, 'pos': 0.011, 'compound': -0.5897},
    {'neg': 0.016, 'neu': 0.975, 'pos': 0.009, 'compound': -0.703},
    {'neg': 0.018, 'neu': 0.973, 'pos': 0.009, 'compound': -0.9264},
    {'neg': 0.026, 'neu': 0.957, 'pos': 0.017, 'compound': -0.5897},
    {'neg': 0.015, 'neu': 0.968, 'pos': 0.017, 'compound': 0.8729},
    {'neg': 0.019, 'neu': 0.969, 'pos': 0.012, 'compound': -0.7736},
    {'neg': 0.017, 'neu': 0.968, 'pos': 0.015, 'compound': 0.8565},
    {'neg': 0.007, 'neu': 0.99, 'pos': 0.002, 'compound': -0.7862},
    {'neg': 0.016, 'neu': 0.973, 'pos': 0.012, 'compound': -0.5897},
    {'neg': 0.013, 'neu': 0.96, 'pos': 0.028, 'compound': 0.9919},
    {'neg': 0.019, 'neu': 0.969, 'pos': 0.012, 'compound': -0.8729},
    {'neg': 0.022, 'neu': 0.965, 'pos': 0.013, 'compound': -0.814},
    {'neg': 0.018, 'neu': 0.969, 'pos': 0.013, 'compound': -0.5897}
]

valores_compound = [sentimento['compound'] for sentimento in pontuacoes]

rotulos = [f"Capítulo {i+1}" for i in range(len(pontuacoes))]

plt.figure(figsize=(10, 6))
plt.bar(rotulos, valores_compound)
plt.xlabel('Capítulo')
plt.ylabel('Sentimento (compound)')
plt.title('Análise de Sentimento por Capítulo')
plt.xticks(rotation=45)
plt.show()
```

O resultado:



Conclui-se que a quantidade de capítulos negativos se sobrepõe à de capítulos positivos.

4. Onde está a Felicidade?

Para esta obra, foram analisados e extraídos os verbos do texto, as palavras-chave, os advérbios e foi feita uma análise de sentimento por cada capítulo do livro. O documento de texto que contém a obra, foi utilizado sempre em formato de *one sentence per line.*

4.1 Palavras-Chave

Para obra "Onde Está a Felicidade?", foi relevante fazer uma extração de palavras-chave da obra. Assim, foi criado o seguinte programa em *python*:

```python
import csv
from collections import Counter
import string
import nltk
from nltk.corpus import stopwords


nltk.download('stopwords')

stop_words = set(stopwords.words('portuguese'))

def extrair_palavras_chave(texto):
    texto = texto.translate(str.maketrans('', '', string.punctuation)).lower()
    palavras = texto.split()
    palavras = [palavra for palavra in palavras if palavra not in stop_words]
    contagem_palavras = Counter(palavras)
    return contagem_palavras

with open('onde_esta_a_felicidade1spl.txt', 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

palavras_chave = extrair_palavras_chave(texto)

top_palavras_chave = palavras_chave.most_common(10)

nome_arquivo_csv = 'felicidadepalavras_chave.csv'

with open(nome_arquivo_csv, 'w', newline='', encoding='utf-8') as arquivo_csv:
    writer = csv.writer(arquivo_csv)
    writer.writerow(['Palavra-chave', 'Frequência'])  # Cabeçalho das colunas

    for palavra, frequencia in top_palavras_chave:
        writer.writerow([palavra, frequencia])

print("As 10 principais palavras-chave foram extraídas e salvas com sucesso no arquivo", nome_arquivo_csv)

```

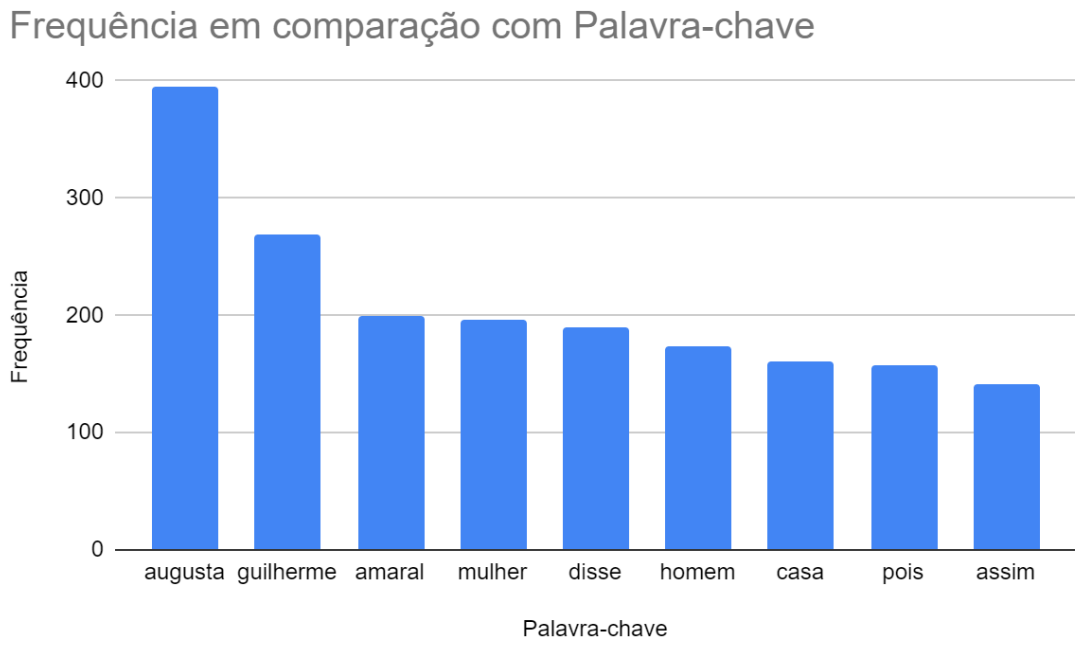
Primeiro, são importadas as bibliotecas necessárias para o processamento de texto, contagem de palavras e manipulação de arquivos csv. Depois, faz-se uma baixa das stop-words, em português, através da NLTK. A função extrair_palavras_chave recebe o texto como entrada, onde dentro da mesma, será removida a pontuação de texto e convertidas as maiúsculas para minúsculas. Depois de o texto dividido e desprovido de stop-words, conta-se a frequência de ocorrência através da biblioteca Counter. Depois de extraídas as 10 palavras que mais surgem no texto, cria-se um documento csv com os resultados, sendo eles:

	Palavra-chave	Frequência
2	—	2315
3	augusta	395
4	guilherme	269
5	amaral	200
6	mulher	197
7	disse	190
8	homem	174
9	casa	161
10	pois	158
11	assim	141
12		

Depois, foi feita então uma limpeza dos dados:

	Palavra-chave	Frequência
2	augusta	395
3	guilherme	269
4	amaral	200
5	mulher	197
6	disse	190
7	homem	174
8	casa	161
9	pois	158
10	assim	141
11		
12		

Para proposta de visualização, tem-se:



4.2 Advérbios

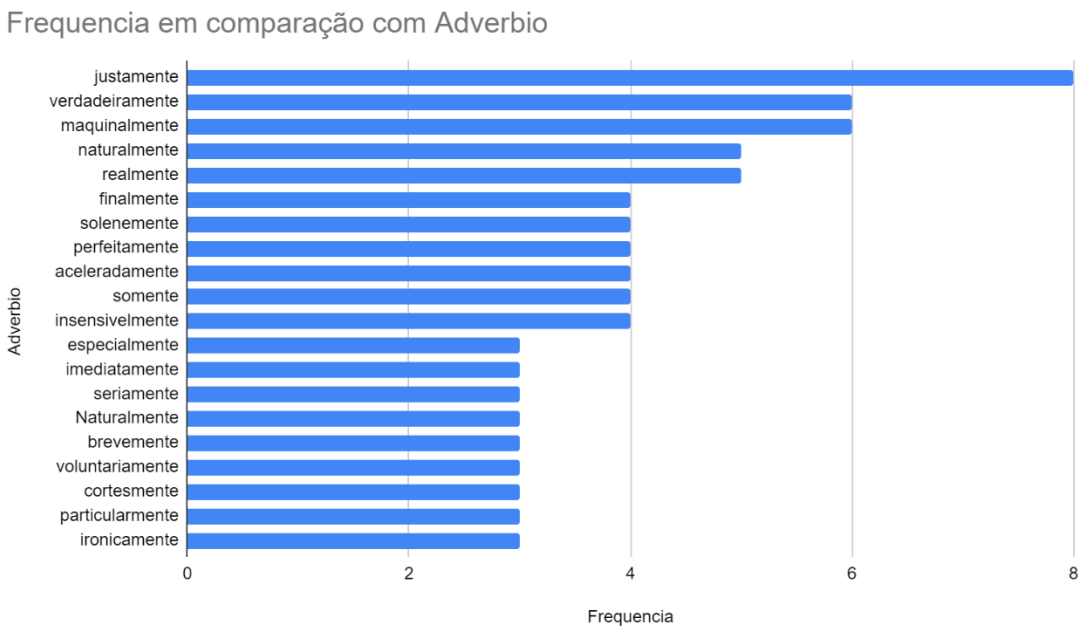
Seguidamente, a análise avançou para uma extração de um top 20 de advérbios que mais surgem na obra, bem como a frequência com que aparecem, através do seguinte programa:

```python
import csv
import re
from collections import Counter

def extrair_adverbios(texto):
    padrao = r'\b\w+mente\b'
    adverbios = re.findall(padrao, texto, re.IGNORECASE)
    return adverbios

def contar_frequencia(adverbios):
    contagem = Counter(adverbios)
    return contagem

def criar_csv(resultado, arquivo):
    with open(arquivo, 'w', newline='') as csv_file:
        writer = csv.writer(csv_file)
        writer.writerow(['Adverbio', 'Frequencia'])
        writer.writerows(resultado)
    print(f'O arquivo {arquivo} foi criado com sucesso!')

with open('onde_esta_a_felicidade1sp1.txt', 'r') as arquivo_txt:
    texto = arquivo_txt.read()

adverbios = extrair_adverbios(texto)

frequencia = contar_frequencia(adverbios)

top_20 = frequencia.most_common(20)

criar_csv(top_20, 'resultado.csv')

```

Este programa em Python permite extrair os 20 advérbios mais frequentes da obra em formato TXT e *one sentence per line*, gerando por fim um arquivo CSV com os resultados. Primeiro, o código lê o arquivo de texto e extrai os advérbios usando expressões regulares. Em seguida, conta a frequência de cada advérbio utilizando a biblioteca *Counter*. Depois disso, seleciona os 20 advérbios mais comuns e cria um arquivo CSV com duas colunas: "Adverbio" e "Frequência". Por fim, o código salva o arquivo CSV com os resultados:

	Adverbio	Frequencia
2	justamente	8
3	verdadeiramente	6
4	maquinalmente	6
5	naturalmente	5
6	realmente	5
7	finalmente	4
8	solenemente	4
9	perfeitamente	4
10	aceleradamente	4
11	somente	4
12	insensivelmente	4
13	especialmente	3
14	imediatamente	3
15	seriamente	3
16	Naturalmente	3
17	brevemente	3
18	voluntariamente	3
19	cortesmente	3
20	particularmente	3
21	ironicamente	3

Como proposta de visualização, uma vez que os dados não precisam de qualquer limpeza:



Percebe-se que, na obra, não é muito utilizado o uso de advérbios sendo que o mais utilizado apenas surge oito vezes.

4.3 Verbos

Ainda para esta obra, foi relevante extrair um top dos 20 verbos que surgem com mais frequência na obra. Então, foi criado o seguinte programa:

```python
import spacy
import os
import csv
from collections import Counter

with open('onde_esta_a_felicidade1spl.txt', 'r', encoding='UTF-8') as file:
    text = file.read()

nlp = spacy.load('pt_core_news_sm')
doc = nlp(text)

dicionario = dict()

for token in doc:
    if token.pos_ == "VERB":
        if token.lemma_ in dicionario.keys():
            dicionario[token.lemma_] += 1
        else:
            dicionario[token.lemma_] = 1

verb_counts = Counter(dicionario)
top20_verbs = verb_counts.most_common(20)

print(top20_verbs)

with open('VERBOSondeestaafelicidade.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Verbo', 'Contagem'])

    for verb, count in top20_verbs:
        writer.writerow([verb, count])

```

O código realiza uma análise de verbos usando a biblioteca Spacy, com o modelo de linguagem em português. Ele conta a frequência de ocorrência de cada verbo no texto e imprime os 20 verbos mais frequentes. Além disso, os resultados são salvos em um arquivo CSV com duas colunas: "Verbo" e "Contagem". Em seguida, um dicionário vazio é criado para armazenar os verbos e as suas contagens.

O loop percorre cada token no documento e verifica se esse token é um verbo. Se for, o lema do verbo é adicionado ao dicionário, aumentando sua contagem de ocorrência. Se o verbo já estiver presente no dicionário, sua contagem é atualizada. Caso contrário, o verbo é adicionado ao dicionário com uma contagem inicial de 1.

A biblioteca `Counter` é usada para contar a frequência dos verbos no dicionário. Os 20 verbos mais comuns são selecionados usando o método `most_common(20)` e armazenados na lista `top20_verbs`. A lista extraída é a seguinte:

	Verbo	Contagem
2	dizer	458
3	ter	414
4	querer	271
5	fazer	252
6	poder	241
7	saber	241
8	dar	187
9	haver	153
10	vir	151
11	ver	146
12	ser	142
13	dever	113
14	falar	94
15	ir	85
16	entrar	85
17	parecer	81
18	deixar	80
19	conhecer	77
20	sair	74
21	passar	68

Neste caso, não efetuei qualquer tipo de limpeza uma vez que os dados se revelaram bastante precisos. Então, como proposta de visualização realizou-se uma wordcloud, através de um programa python:

```python
import matplotlib.pyplot as plt
from wordcloud import WordCloud

texto = "dizer ter querer fazer poder saber dar haver vir ver ser dever falar ir entrar parecer deixar conhecer sair passar."

wordcloud = WordCloud().generate(texto)

plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

O resultado foi o seguinte:

Ademais, foi feito também um gráfico de barras:

4.4 Análise de Sentimento

Por fim, para fazer a análise de sentimento de cada capítulo, criei o seguinte programa Python, que utiliza a biblioteca NLTK (Natural Language Toolkit) para realizar uma análise de sentimentos num dado texto. Ele utiliza o módulo `SentimentIntensityAnalyzer` do NLTK, que é responsável por atribuir pontuações de sentimento a diferentes partes do texto.

```python
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer

nltk.download('vader_lexicon')

sia = SentimentIntensityAnalyzer()

caminho_arquivo = "onde_esta_a_felicidade1spl.txt"

with open(caminho_arquivo, 'r', encoding='utf-8') as arquivo:
    texto = arquivo.read()

capitulos = texto.split('# Capítulo')

for i, capitulo in enumerate(capitulos[1:]):
    sentiment = sia.polarity_scores(capitulo)
    print(f"Sentimento do Capítulo {i+1}: {sentiment}")

```

Primeiro, o código importa as bibliotecas necessárias e faz o download do léxico Vader do NLTK, que é usado para análise de sentimentos. Em seguida, é criada uma instância do `SentimentIntensityAnalyzer`.

Depois, o código lê o arquivo txt da obra *Onde está a felicidade?* no formato de one sentence per line e armazena-o numa variável. O texto é dividido em capítulos usando um marcador específico (no caso, foi colocada um # em cada capítulo no documento de texto). Através de um loop, cada capítulo é analisado usando o `SentimentIntensityAnalyzer`, que retorna uma pontuação de sentimento para aquele capítulo.

Em resumo, o código realiza a análise de sentimentos no texto, dividindo-o em capítulos e calculando a pontuação de sentimento para cada um deles. Os resultados no terminal são:

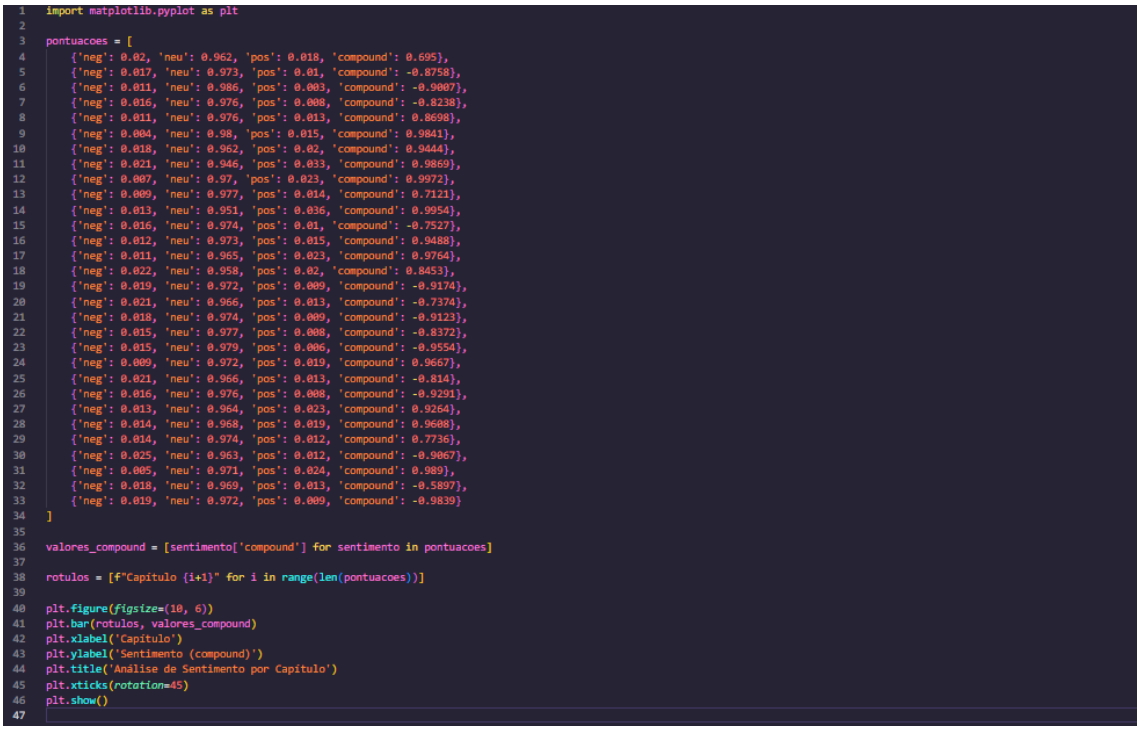
```
Sentimento do Capítulo 1: {'neg': 0.02, 'neu': 0.962, 'pos': 0.018, 'compound': 0.695}
Sentimento do Capítulo 2: {'neg': 0.017, 'neu': 0.973, 'pos': 0.01, 'compound': -0.8758}
Sentimento do Capítulo 3: {'neg': 0.011, 'neu': 0.986, 'pos': 0.003, 'compound': -0.9007}
Sentimento do Capítulo 4: {'neg': 0.016, 'neu': 0.976, 'pos': 0.008, 'compound': -0.8238}
Sentimento do Capítulo 5: {'neg': 0.011, 'neu': 0.976, 'pos': 0.013, 'compound': 0.8698}
Sentimento do Capítulo 6: {'neg': 0.004, 'neu': 0.98, 'pos': 0.015, 'compound': 0.9841}
Sentimento do Capítulo 7: {'neg': 0.018, 'neu': 0.962, 'pos': 0.02, 'compound': 0.9444}
Sentimento do Capítulo 8: {'neg': 0.021, 'neu': 0.946, 'pos': 0.033, 'compound': 0.9869}
Sentimento do Capítulo 9: {'neg': 0.007, 'neu': 0.97, 'pos': 0.023, 'compound': 0.9972}
Sentimento do Capítulo 10: {'neg': 0.009, 'neu': 0.977, 'pos': 0.014, 'compound': 0.7121}
Sentimento do Capítulo 11: {'neg': 0.013, 'neu': 0.951, 'pos': 0.036, 'compound': 0.9954}
Sentimento do Capítulo 12: {'neg': 0.016, 'neu': 0.974, 'pos': 0.01, 'compound': -0.7527}
Sentimento do Capítulo 13: {'neg': 0.012, 'neu': 0.973, 'pos': 0.015, 'compound': 0.9488}
Sentimento do Capítulo 14: {'neg': 0.011, 'neu': 0.965, 'pos': 0.023, 'compound': 0.9764}
Sentimento do Capítulo 15: {'neg': 0.022, 'neu': 0.958, 'pos': 0.02, 'compound': 0.8453}
Sentimento do Capítulo 16: {'neg': 0.019, 'neu': 0.972, 'pos': 0.009, 'compound': -0.9174}
Sentimento do Capítulo 17: {'neg': 0.021, 'neu': 0.966, 'pos': 0.013, 'compound': -0.7374}
Sentimento do Capítulo 18: {'neg': 0.018, 'neu': 0.974, 'pos': 0.009, 'compound': -0.9123}
Sentimento do Capítulo 19: {'neg': 0.015, 'neu': 0.977, 'pos': 0.008, 'compound': -0.8372}
Sentimento do Capítulo 20: {'neg': 0.015, 'neu': 0.979, 'pos': 0.006, 'compound': -0.9554}
Sentimento do Capítulo 21: {'neg': 0.009, 'neu': 0.972, 'pos': 0.019, 'compound': 0.9667}
Sentimento do Capítulo 22: {'neg': 0.021, 'neu': 0.966, 'pos': 0.013, 'compound': -0.814}
Sentimento do Capítulo 23: {'neg': 0.016, 'neu': 0.976, 'pos': 0.008, 'compound': -0.9291}
Sentimento do Capítulo 24: {'neg': 0.013, 'neu': 0.964, 'pos': 0.023, 'compound': 0.9264}
Sentimento do Capítulo 25: {'neg': 0.014, 'neu': 0.968, 'pos': 0.019, 'compound': 0.9608}
Sentimento do Capítulo 26: {'neg': 0.014, 'neu': 0.974, 'pos': 0.012, 'compound': 0.7736}
Sentimento do Capítulo 27: {'neg': 0.025, 'neu': 0.963, 'pos': 0.012, 'compound': -0.9067}
Sentimento do Capítulo 28: {'neg': 0.005, 'neu': 0.971, 'pos': 0.024, 'compound': 0.989}
Sentimento do Capítulo 29: {'neg': 0.018, 'neu': 0.969, 'pos': 0.013, 'compound': -0.5897}
Sentimento do Capítulo 30: {'neg': 0.019, 'neu': 0.972, 'pos': 0.009, 'compound': -0.9839}
```

Tipo de sentimento de cada capítulo com base no valor do compound:

- Capítulo 1: Sentimento positivo (compound: 0.695)
- Capítulo 2: Sentimento negativo (compound: -0.8758)
- Capítulo 3: Sentimento negativo (compound: -0.9007)
- Capítulo 4: Sentimento negativo (compound: -0.8238)
- Capítulo 5: Sentimento positivo (compound: 0.8698)
- Capítulo 6: Sentimento positivo (compound: 0.9841)
- Capítulo 7: Sentimento positivo (compound: 0.9444)
- Capítulo 8: Sentimento positivo (compound: 0.9869)
- Capítulo 9: Sentimento positivo (compound: 0.9972)

- Capítulo 10: Sentimento positivo (compound: 0.7121)
- Capítulo 11: Sentimento positivo (compound: 0.9954)
- Capítulo 12: Sentimento negativo (compound: -0.7527)
- Capítulo 13: Sentimento positivo (compound: 0.9488)
- Capítulo 14: Sentimento positivo (compound: 0.9764)
- Capítulo 15: Sentimento positivo (compound: 0.8453)
- Capítulo 16: Sentimento negativo (compound: -0.9174)
- Capítulo 17: Sentimento negativo (compound: -0.7374)
- Capítulo 18: Sentimento negativo (compound: -0.9123)
- Capítulo 19: Sentimento negativo (compound: -0.8372)
- Capítulo 20: Sentimento negativo (compound: -0.9554)
- Capítulo 21: Sentimento positivo (compound: 0.9667)
- Capítulo 22: Sentimento negativo (compound: -0.814)
- Capítulo 23: Sentimento negativo (compound: -0.9291)
- Capítulo 24: Sentimento positivo (compound: 0.9264)
- Capítulo 25: Sentimento positivo (compound: 0.9608)
- Capítulo 26: Sentimento positivo (compound: 0.7736)
- Capítulo 27: Sentimento negativo (compound: -0.9067)
- Capítulo 28: Sentimento positivo (compound: 0.989)
- Capítulo 29: Sentimento negativo (compound: -0.5897)
- Capítulo 30: Sentimento negativo (compound: -0.9839)

Para criar uma proposta de visualização, foi criado o seguinte programa:



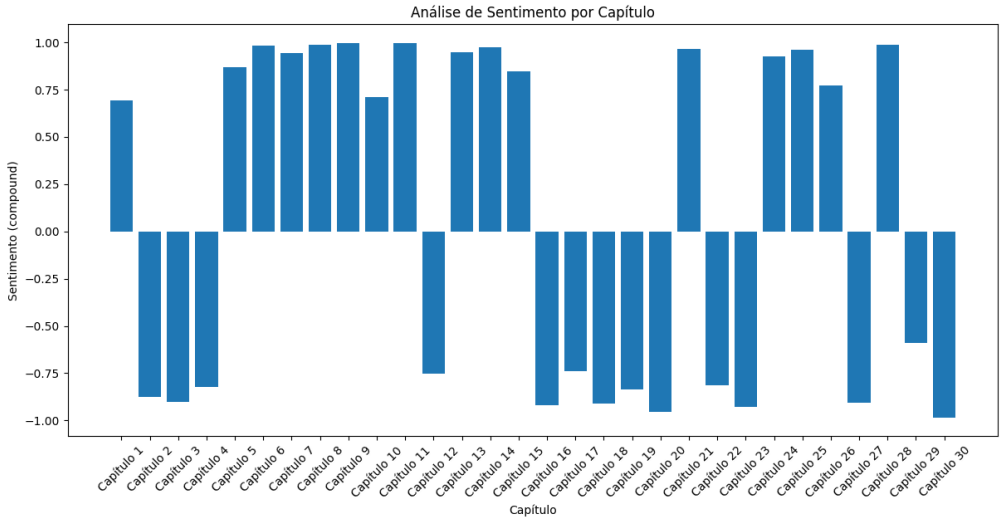
```python
import matplotlib.pyplot as plt

pontuacoes = [
    {'neg': 0.02, 'neu': 0.962, 'pos': 0.018, 'compound': 0.695},
    {'neg': 0.017, 'neu': 0.973, 'pos': 0.01, 'compound': -0.8758},
    {'neg': 0.011, 'neu': 0.986, 'pos': 0.003, 'compound': -0.9007},
    {'neg': 0.016, 'neu': 0.976, 'pos': 0.008, 'compound': -0.8238},
    {'neg': 0.011, 'neu': 0.976, 'pos': 0.013, 'compound': 0.8698},
    {'neg': 0.004, 'neu': 0.98, 'pos': 0.015, 'compound': 0.9841},
    {'neg': 0.018, 'neu': 0.962, 'pos': 0.02, 'compound': 0.9444},
    {'neg': 0.021, 'neu': 0.946, 'pos': 0.033, 'compound': 0.9869},
    {'neg': 0.007, 'neu': 0.97, 'pos': 0.023, 'compound': 0.9972},
    {'neg': 0.009, 'neu': 0.977, 'pos': 0.014, 'compound': 0.7121},
    {'neg': 0.013, 'neu': 0.951, 'pos': 0.036, 'compound': 0.9954},
    {'neg': 0.016, 'neu': 0.974, 'pos': 0.01, 'compound': -0.7527},
    {'neg': 0.012, 'neu': 0.973, 'pos': 0.015, 'compound': 0.9488},
    {'neg': 0.011, 'neu': 0.965, 'pos': 0.023, 'compound': 0.9764},
    {'neg': 0.022, 'neu': 0.958, 'pos': 0.02, 'compound': 0.8453},
    {'neg': 0.019, 'neu': 0.972, 'pos': 0.009, 'compound': -0.9174},
    {'neg': 0.021, 'neu': 0.966, 'pos': 0.013, 'compound': -0.7374},
    {'neg': 0.018, 'neu': 0.974, 'pos': 0.009, 'compound': -0.9123},
    {'neg': 0.015, 'neu': 0.977, 'pos': 0.008, 'compound': -0.8372},
    {'neg': 0.015, 'neu': 0.979, 'pos': 0.006, 'compound': -0.9554},
    {'neg': 0.009, 'neu': 0.972, 'pos': 0.019, 'compound': 0.9667},
    {'neg': 0.021, 'neu': 0.966, 'pos': 0.013, 'compound': -0.814},
    {'neg': 0.016, 'neu': 0.976, 'pos': 0.008, 'compound': -0.9291},
    {'neg': 0.013, 'neu': 0.964, 'pos': 0.023, 'compound': 0.9264},
    {'neg': 0.014, 'neu': 0.968, 'pos': 0.019, 'compound': 0.9608},
    {'neg': 0.014, 'neu': 0.974, 'pos': 0.012, 'compound': 0.7736},
    {'neg': 0.025, 'neu': 0.963, 'pos': 0.012, 'compound': -0.9067},
    {'neg': 0.005, 'neu': 0.971, 'pos': 0.024, 'compound': 0.989},
    {'neg': 0.018, 'neu': 0.969, 'pos': 0.013, 'compound': -0.5897},
    {'neg': 0.019, 'neu': 0.972, 'pos': 0.009, 'compound': -0.9839}
]

valores_compound = [sentimento['compound'] for sentimento in pontuacoes]

rotulos = [f"Capítulo {i+1}" for i in range(len(pontuacoes))]

plt.figure(figsize=(10, 6))
plt.bar(rotulos, valores_compound)
plt.xlabel('Capítulo')
plt.ylabel('Sentimento (compound)')
plt.title('Análise de Sentimento por Capítulo')
plt.xticks(rotation=45)
plt.show()
```

O código em questão utiliza a biblioteca Matplotlib em conjunto com os resultados da análise de sentimento por capítulos para criar um gráfico de barras. O objetivo é visualizar a variação do sentimento ao longo dos capítulos de uma obra. Inicialmente, os resultados da análise de sentimento por capítulo são armazenados numa lista chamada `*pontuacoes*`. Cada elemento dessa lista é um dicionário que contém as pontuações de negatividade (*'neg'*), neutralidade (*'neu'*), positividade (*'pos'*) e a pontuação geral do sentimento (*'compound'*) para um capítulo específico.

O próximo passo consiste em extrair os valores do 'compound' de cada dicionário da lista `pontuacoes`. Esses valores são armazenados numa lista separada chamada `valores_compound`. Esta lista representa a medida agregada do sentimento para cada capítulo. A lista `rotulos` é criada para armazenar os rótulos dos capítulos. Cada rótulo é definido como "Capítulo X", onde X é o número do capítulo correspondente. Com todos os dados necessários preparados, o código utiliza a biblioteca Matplotlib para criar o gráfico de barras. É criada uma figura com um tamanho específico utilizando a função `plt.figure(figsize=(10, 6))`. Em seguida, a função `plt.bar()` é utilizada para criar as barras do gráfico. Os rótulos dos capítulos são exibidos no eixo x e os valores do compound são representados no eixo y. Algumas funções do Matplotlib, como `plt.xlabel()`, `plt.ylabel()`, `plt.title()` e `plt.xticks(rotation=45)`, são utilizadas para adicionar rótulos aos eixos e ao gráfico em geral, além de fazer uma rotação de 45 graus

nos rótulos dos capítulos para facilitar a leitura. Por fim, o gráfico é exibido na tela utilizando a função `plt.show()`.

Assim, o gráfico:



Conclusão

Ao analisar e avaliar as obras de Camilo Castelo Branco, observa-se que o sentimento predominante das suas obras é geralmente negativo. Através de uma profunda exploração da condição humana, o autor retrata temas como o sofrimento, a tragédia, a melancolia e os conflitos internos das personagens. As suas narrativas mergulham em dilemas morais, paixões não correspondidas, traições e destinos trágicos, criando um ambiente carregado de emoções negativas. Ademais, a análise e visualização de dados traz uma dimensão adicional à literatura, promovendo uma compreensão mais profunda e uma conexão mais significativa entre os escritores, os investigadores e, claro, os leitores. Estas ferramentas auxiliam no desenvolvimento e evolução da literatura, tornando-a mais dinâmica, relevante e impactante no contexto contemporâneo.

Bibliografia

OpenAI. (2022). ChatGPT – OpenAI

https://chat.openai.com/

NLTK 2023, NLTK Project

https://www.nltk.org/

Hutto, C.J., & Gilbert, E. (2014). vaderSentiment: VADER Sentiment Analysis. Versão 3.3.2 [Software]

https://pypi.org/project/vaderSentiment/

Github Inês Leão

https://github.com/inesctleao/avd2023/tree/main