MINERAÇÃO DE TEXTO (Parte 3) - ANÁLISE DE SENTIMENTOS COM CLASSIFICAÇÃO BINÁRIA

Livro: The Amateur Emigrant **de** Robert Louis Stevenson

Instalando o NLTK (NATURAL LANGUAGE TOOLKIT)

In [1]:
```
pip install nltk
```

```
Requirement already satisfied: nltk in c:\users\mmateus\anaconda3\lib\site-packages
(3.6.5)
Requirement already satisfied: click in c:\users\mmateus\anaconda3\lib\site-packages
(from nltk) (8.0.3)
Requirement already satisfied: joblib in c:\users\mmateus\anaconda3\lib\site-package
s (from nltk) (1.1.0)
Requirement already satisfied: regex>=2021.8.3 in c:\users\mmateus\anaconda3\lib\sit
e-packages (from nltk) (2021.8.3)
Requirement already satisfied: tqdm in c:\users\mmateus\anaconda3\lib\site-packages
(from nltk) (4.62.3)
Requirement already satisfied: colorama in c:\users\mmateus\anaconda3\lib\site-packa
ges (from click->nltk) (0.4.4)
Note: you may need to restart the kernel to use updated packages.
```

Importando as bibliotecas necessárias:

In [2]:
```
import nltk
import matplotlib.pyplot as plt
import pandas as pd
import re

# Importando o Corpora e funções do NLTK...
from nltk.corpus import stopwords
from nltk.corpus import opinion_lexicon
from nltk.tokenize import word_tokenize

nltk.download('stopwords', quiet=True)
nltk.download('opinion_lexicon', quiet=True)
nltk.download('punkt', quiet=True)

# ... e do Matplotlib
plt.style.use('ggplot')
```

Importando o texto de Stevenson:

In [3]:
```
df = pd.read_csv('data/TheAmateurEmigrant.txt',sep='\t')\
    .dropna()

df.head(10)
```

Out[3]:

	text
0	THE AMATEUR EMIGRANT
1	THE SECOND CABIN

	text
2	I first encountered my fellow-passengers on th...
3	Thence we descended the Clyde in no familiar s...
4	on each other as on possible enemies. A few S...
5	already grown acquainted on the North Sea, wer...
6	their long pipes; but among English speakers d...
7	reigned supreme. The sun was soon overclouded...
8	grew sharp as we continued to descend the wide...
9	falling temperature the gloom among the passen...

Preparação do texto:

In [4]:
```
# Processo de limpeza, organização e tokenização do texto (dados não estruturados)
def clean_text(text):
    text = text.lower()
    text = text.replace("'", '')
    text = re.sub(r'[^\w]', ' ', text)
    text = re.sub(r'\s+', ' ', text)
    text = text.strip()
    return text

df['text'] = df['text'].map(clean_text)
df['text'] = df['text'].map(word_tokenize)

df.head()
```

Out[4]:

	text
0	[the, amateur, emigrant]
1	[the, second, cabin]
2	[i, first, encountered, my, fellow, passengers...
3	[thence, we, descended, the, clyde, in, no, fa...
4	[on, each, other, as, on, possible, enemies, a...

In [5]:
```
df = df.text.explode().to_frame('token')
df.head(10)
```

Out[5]:

	token
0	the
0	amateur
0	emigrant
1	the
1	second
1	cabin
2	i

	token
2	first
2	encountered
2	my

In [6]:
```
df.token.value_counts().head(10)
```

Out[6]:
```
the    1548
and    1079
of      860
a       808
to      687
was     498
in      481
i       419
he      406
had     275
Name: token, dtype: int64
```

In [7]:
```
# Removendo Stop Words
stopwords = set(stopwords.words('english'))
```

In [8]:
```
df = df[~df.token.isin(stopwords)]
```

In [9]:
```
df.token.value_counts().head(10)
```

Out[9]:
```
one         129
man          94
like         70
would        67
said         56
could        55
upon         55
two          51
steerage     50
good         46
Name: token, dtype: int64
```

Machine Learning: Classificando os sentimentos positivos e negativos com o Opinion Lexicon

In [10]:
```
sentiment_lexicon = {
    **{w: 'positive' for w in opinion_lexicon.positive()},
    **{w: 'negative' for w in opinion_lexicon.negative()}
}

df['sentiment'] = df['token'].map(sentiment_lexicon)
df = df[~df.sentiment.isna()] # ommit words out of opinion lexicon

df.head(10)
```

Out[10]:

	token	sentiment
3	askance	negative

	token	sentiment
4	enemies	negative
5	friendly	positive
6	suspicion	negative
7	supreme	positive
8	sharp	positive
9	falling	negative
9	gloom	negative
11	scarce	negative
12	cold	negative

In [11]:
```
df.token.value_counts().head(10)
```

Out[11]:
```
like      70
good      46
well      32
work      27
better    23
sick      21
great     17
enough    16
hard      12
poor      11
Name: token, dtype: int64
```

Sumarizando as palavras por sentimentos

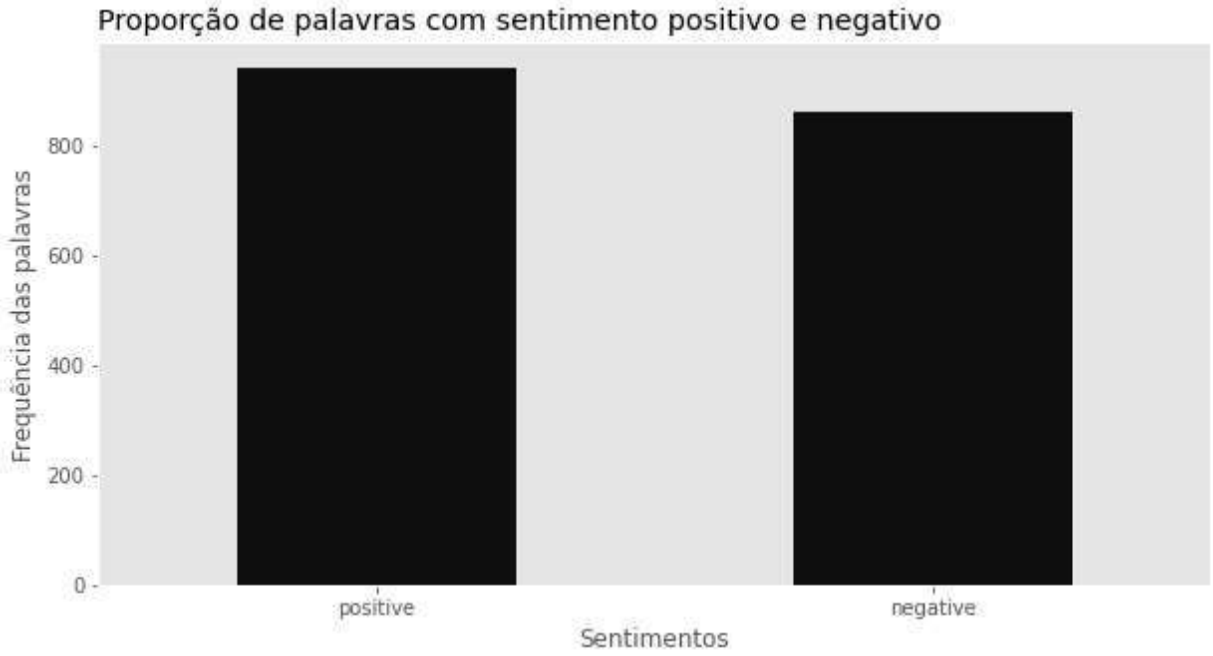
In [12]:
```
summary_df = df.sentiment.value_counts().to_frame('n')
summary_df['prop'] = summary_df['n'] / summary_df.n.sum()

summary_df.round(3)
```

Out[12]:

	n	prop
positive	942	0.522
negative	863	0.478

Gráfico com a Classificação Binária dos sentimentos

In [13]:
```
summary_df.n.plot.bar(legend=False, figsize=(10, 5), grid=False, color='darkblue')
plt.xlabel('Sentimentos')
plt.ylabel('Frequência das palavras')
plt.title('Proporção de palavras com sentimento positivo e negativo', loc='left')
plt.xticks(rotation=0);
```



A narrativa de viagem de Stevenson mostra um certo equilíbrio entre palavras com 52,2% dos sentimentos sendo positivos contra 47,8% de negativos; isso se reflete, em termos de quantidade, no gráfico acima. Por se tratar de uma história real, ambientada num navio cuja tripulação era composta, em sua maioria, por homens desempregados (bem pelas esposas e filhos de muitos deles), que estavam em busca de melhores condições de vida na América, esse resultado pode causar surpresa em quem esperava que o tom da história tendesse mais para o lado negativo. No entanto, o escritor soube dosar a sua narrativa com fatos de como esses viajantes se divertiam, a camaradagem que existia entre eles e a esperança que depositavam no Novo Mundo.