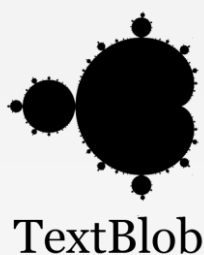


Analise de Sentimento em Frases



Com NLTK, TextBlob e
Transformers



Analise de Sentimento

Analise de Sentimento

Análise de sentimento é um termo que se refere ao uso de **processamento de linguagem natural**, análise de texto e linguística computacional para determinar a atitude de um falante ou escritor em relação a um tópico específico.

Basicamente, ajuda a determinar **se um texto expressa sentimentos** positivos, negativos ou neutros. A análise de sentimento é uma excelente maneira de descobrir como as pessoas, principalmente os consumidores, se sentem sobre um determinado tópico, produto ou ideia.



Artigo (Tutorial)

Nesse tutorial vou **explorar técnicas e uso de bibliotecas** para gerar a análise de sentimento em uma base de dados do Twitter.

A ideia é **explorar técnicas** e **conhecimento sobre as bibliotecas**.



Atenção !

As **bibliotecas que vou utilizar** possuem algoritmos para identificar a expressão de sentimentos em suas funções, **assim não vou utilizar Machine Learning** no processo de aprendizado e também vou utilizar uma **base de dados do Twitter em Inglês**.

Caso queira fazer um processo de análise de sentimento com um modelo, recomendo o link abaixo.

<https://www.youtube.com/watch?v=ywbzwTc51y4>

Vamos utilizar uma base de dados da Kaggle

<https://www.kaggle.com/crowdflower/twitter-airline-sentiment>

Vamos importar as bibliotecas necessárias

```
[14] # Lib para modelagem de Dados
import pandas as pd
# Lib para uso de vetores
import numpy as np
# Lib para mineração de textos
import nltk
# Função para extrair palavras sem relevâncias
from nltk.corpus import stopwords
# Lib para trabalhar com textos
import string
# Lib para expressões regulares
import re
# Lib para trabalhar com abreviações no inglês
# Exemplo: you're --> you are
import contractions
```

Vamos ler os dados e explorar algumas informações

```
[4] # Lendo a Base de Dados
Base_Dados = pd.read_csv('Tweets.csv')

# Verificando as primeiras linhas
Base_Dados.head()
```

↳

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence
0	570306133677760513	neutral	1.0000	NaN	NaN
1	570301130888122368	positive	0.3486	NaN	0.0000
2	570301083672813571	neutral	0.6837	NaN	NaN
3	570301031407624196	negative	1.0000	Bad Flight	0.7033
4	570300817074462722	negative	1.0000	Can't Tell	1.0000

```
[5] # Verificando as dimensões da base de dados
Base_Dados.shape
```

(14640, 15)

```
[6] # Verificando as colunas da Base de Dados
Base_Dados.columns
```

```
Index(['tweet_id', 'airline_sentiment', 'airline_sentiment_confidence',
      'negativereason', 'negativereason_confidence', 'airline',
      'airline_sentiment_gold', 'name', 'negativereason_gold',
      'retweet_count', 'text', 'tweet_coord', 'tweet_created',
      'tweet_location', 'user_timezone'],
      dtype='object')
```

Retirando colunas sem necessidades

```
[7] # Retirando colunas desnecessarias para nosso case
Base_Filtrada = Base_Dados.drop(['tweet_id','retweet_count',
                                  'tweet_coord', 'tweet_created',
                                  'tweet_location','name',
                                  'user_timezone'],axis = 1)

# Verificando a nova dimensão
Base_Filtrada.head()
```

	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	airline_sentiment_gold
0	neutral	1.0000	NaN	NaN	Virgin America	NaN
1	positive	0.3486	NaN	0.0000	Virgin America	NaN
2	neutral	0.6837	NaN	NaN	Virgin America	NaN
3	negative	1.0000	Bad Flight	0.7033	Virgin America	NaN
4	negative	1.0000	Can't Tell	1.0000	Virgin America	NaN

Retirando os valores ‘Nan’ → Só enche o saco esses ‘NaN’ -_-’

```
[10] # Retinando os NaN da base de dados
Base_Filtrada['negativereason'] = Base_Filtrada['negativereason'].fillna('')
Base_Filtrada['negativereason_confidence'] = Base_Filtrada['negativereason_confidence'].fillna('')
Base_Filtrada['airline_sentiment_gold'] = Base_Filtrada['airline_sentiment_gold'].fillna('')
Base_Filtrada['negativereason_gold'] = Base_Filtrada['negativereason_gold'].fillna('')

# Verificando os primeiros registros
Base_Filtrada.head()
```

	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	airline_sentiment_gold
0	neutral	1.0000			Virgin America	
1	positive	0.3486		0	Virgin America	
2	neutral	0.6837			Virgin America	
3	negative	1.0000	Bad Flight	0.7033	Virgin America	
4	negative	1.0000	Can't Tell	1	Virgin America	

Utilizando uma técnica para tratamento dos textos

```
[12] # Esse mesmo tratamento pode ser aplicado no Portugues
      # Exceto a erapa do contractions

      # Função para fazer o tratamento no Texto
      # Serão aplicados diversos tratamentos
      def Limpeza_Texto(Texto):

          if Texto:
              # Retirando as abreviações do Ingles
              # exemplo: you're --> you are | i'm -> I am
              Texto = contractions.fix(Texto)

              # Retirando os pontos '...' do Texto
              Texto = ' '.join(Texto.split('.'))

              # Removendo Acentuações, Caracteres Especiais
              # E transformando tudo em minúsculo
              Texto = re.sub( r'\s+', ' ',
                             re.sub('[^A-Za-z0-9]', ' ',
                                     Texto.strip().lower())).strip()

              # Transoformando os valores numeros em espaço
              Texto = re.sub(r'\W+', ' ', Texto.strip().lower()).strip()

              # Quebrando a frase em uma lista com as palavras
              Texto = [Palavra for Palavra in Texto.split()]

              # Retornando a Lista
              return Texto

      # Aplicando a função
      # No Lambda, a lista será unificada em uma unica celula
      Base_Filtrada['text'] = Base_Filtrada['text'].apply( lambda Linha: ' '.join(Limpeza_Texto(Linha) ) )
```

Veja o Antes x Depois

```
[13] # Analisando o resultado da Função

      # Comparando o texto depois do Tratamento
      print('Como era antes do tratamento ...')
      for Antes in Base_Dados['text'].head():
          print( Antes )

      print('\n', 'Como ficou depois ...')
      for Depois in Base_Filtrada['text'].head():
          print( Depois )

      Como era antes do tratamento ...
      @VirginAmerica What @dhepburn said.
      @VirginAmerica plus you've added commercials to the experience... tacky.
      @VirginAmerica I didn't today... Must mean I need to take another trip!
      @VirginAmerica it's really aggressive to blast obnoxious "entertainment" in your guests' faces & they have little recourse
      @VirginAmerica and it's a really big bad thing about it

      Como ficou depois ...
      virginamerica what dhepburn said
      virginamerica plus you have added commercials to the experience tacky
      virginamerica i did not today must mean i need to take another trip
      virginamerica it is really aggressive to blast obnoxious entertainment in your guests faces amp they have little recourse
      virginamerica and it is a really big bad thing about it
```

Instalando alguns complementos do NLTK

```
[▶] # Iremos precisar instalar alguns pacotes extra da NLTK
      # Esses pacotes irão ajudar na mineração dos textos

      nltk.download(['names', # lista de nomes comuns em inglês
                    'stopwords', # Palabras sem relevância
                    ])
```

```
↳ [nltk_data] Downloading package names to /root/nltk_data...
   [nltk_data] Unzipping corpora/names.zip.
   [nltk_data] Downloading package stopwords to /root/nltk_data...
   [nltk_data] Unzipping corpora/stopwords.zip.
```

Vamos explorar a função de Analise de Sentimento da NLTK

```
[26] # Função implementar e facilitar tarefas de análise de sentimento usando
# recursos e classificadores NLTK
from nltk.sentiment import SentimentIntensityAnalyzer

# Atribuindo a Função
Função_SIA = SentimentIntensityAnalyzer()

# Essa função retorna um score com categorias
# Cada categoria tera um valor
# quanto mais alto o valor, mais indica o possivel sentimento da frase

# 1º Exemplo
# Vamos escrever "Eu odeio chocolate" em inglês e ver o retorno da função
print('Odeio Chocolate: ', Função_SIA.polarity_scores('i hate chocolate'), '\n' )

# 2º Exemplo
# Vamos escrever "Eu amo chocolate" em inglês e ver o retorno da função
print('Amo Chocolate: ', Função_SIA.polarity_scores('i love chocolate')) )
```

Odeio Chocolate: {'neg': 0.787, 'neu': 0.213, 'pos': 0.0, 'compound': -0.5719}

Amo Chocolate: {'neg': 0.0, 'neu': 0.192, 'pos': 0.808, 'compound': 0.6369}

Vamos aplicar na nossa base de dados essa função

```
[32] # Gerando o Score quanto ao Sentimento da Frase com a Função_SIA

# Convertendo em uma lista as frases
Textos = Base_Filtrada['text'].tolist()

# Lista para receber valores
Scores_Negativos = []
Scores_Neutros = []
Scores_Positivos = []
Combinação_Scores = []
Finale = []

# Loop nos Textos
for Frases in Textos:


    # Função para gerar o Score de Emoção
    Score_Emoção = Função_SIA.polarity_scores(Frases)

    # Adicionando o score negativo nas listas
    Scores_Negativos.append(Score_Emoção['neg'])

    # Adicionando o score positivo nas listas
    Scores_Positivos.append(Score_Emoção['pos'])

    # Adicionando o score Neutros nas listas
    Scores_Neutros.append(Score_Emoção['neu'])

    # Adicionando a composição nas listas
    Combinação_Scores.append(Score_Emoção['compound'])
```



Continuação da imagem anterior

```
# Condição para flegar a marcação da emoção

# Caso a Combinação maior que Zero
# A Flag será Positivo
if Score_Emoção['compound'] > 0:
    Finale.append('positive')

# Caso a Combinação menor que Zero
# A Flag será Positivo
elif Score_Emoção['compound'] < 0:
    Finale.append('negative')

# Caso contrário negativa
else:
    Finale.append('neutral')

# Atribuindo as listas na nossa base de dados
Base_Filtrada['negative_score'] = Scores_Negativos
Base_Filtrada['positive_score'] = Scores_Positivos
Base_Filtrada['neutral_score'] = Scores_Neutros
Base_Filtrada['compound_score'] = Combinação_Scores
Base_Filtrada['Finale'] = Finale
```

Veja como fica os dados

```
[33] # Verificando o resultado
Base_Filtrada[['text', 'negative_score', 'positive_score',
               'neutral_score', 'compound_score', 'Finale']].head(10)
```

	text	negative_score	positive_score	neutral_score	compound_score	Finale
0	virginamerica what dhepburn said	0.000	0.000	1.000	0.0000	neutral
1	virginamerica plus you have added commercials ...	0.000	0.000	1.000	0.0000	neutral
2	virginamerica i did not today must mean i need...	0.000	0.000	1.000	0.0000	neutral
3	virginamerica it is really aggressive to blast...	0.216	0.123	0.661	-0.2716	negative
4	virginamerica and it is a really big bad thing...	0.296	0.000	0.704	-0.5829	negative
5	virginamerica seriously would pay 30 a flight ...	0.238	0.069	0.693	-0.5945	negative
6	virginamerica yes nearly every time i fly vx t...	0.000	0.172	0.828	0.4019	positive
7	virginamerica really missed a prime opportunit...	0.142	0.175	0.683	0.1458	positive
8	virginamerica well i did not but now i do d	0.000	0.205	0.795	0.1406	positive
9	virginamerica it was amazing and arrived an ho...	0.000	0.340	0.660	0.7717	positive

Vamos avaliar o resultado

```
[35] # Vamos avaliar o quanto a Lib NLTK tem de acurácia no seu modelo

# Função para metricas de classificação
from sklearn.metrics import classification_report

# Avaliando a performance da Lib NTL
print('Avaliando a performance da NLTK', '\n')
print( classification_report(
    Base_Filtrada['airline_sentiment'],
    Base_Filtrada['Finale'] ) )
```

Avaliando a performance da NLTK

	precision	recall	f1-score	support
negative	0.90	0.45	0.60	9178
neutral	0.43	0.32	0.37	3099
positive	0.27	0.90	0.42	2363
accuracy			0.49	14640
macro avg	0.53	0.56	0.46	14640
weighted avg	0.70	0.49	0.52	14640

A Acurácia não foi das melhores.
Vale lembrar que estamos usando um algoritmo da própria biblioteca



Vamos utilizar outras Biblioteca

```
[36] # Vamos testar agora com a textblob
      #TextBlob é uma lib para processamento de dados textuais
```

```
pip install textblob
```

```
Requirement already satisfied: textblob in /usr/local/lib/python3.7/dist-packages (0.15)
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.7/dist-packages (from textblob)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nltk>=3.1->textblob)
```

```
[40] # Função para analisar as frases
      from textblob import TextBlob
```

```
[52] # Vamos definir uma frase
      # Frase transcrita: 'Eu odeio chocolate'
      Frase = TextBlob('i hate chocolate')
```

```
[53] # Identificando as palavras
      Frase.words
```

```
WordList(['i', 'hate', 'chocolate'])
```

```
[54] # Identificando a frase
      Frase.sentences
```

```
[Sentence("i hate chocolate")]
```

```
[56] # Utilizando o score de avaliação de sentimento
      # Tem a mesma função da 'Função_SIA' da NLTK
```

```
# O valor varia de -1 a 1
# Mais proximo de 1 - Positivo
# Mais proxima de -1 Negativo
# 0 neutro
```

```
print( Frase.polarity )
```

```
-0.8
```

Vamos utilizar uma técnica para gerar a análise dos sentimentos

```
[59] # Gerando o Score quanto ao Sentimento da Frase com a TextBlob

# Convertendo em uma lista as frases
Textos_02 = Base_Filtrada['text'].tolist()

# Listas para salvar os valores
Score_Frase = []
Finale = []

# Loop nos Textos
for Frase in Textos_02:

    # Atribuindo a função do TextBlob
    Atribuindo_TextBlob = TextBlob(Frase)

    # Gerando o Score de sentimento
    Gerando_Score = Atribuindo_TextBlob.polarity

    # Adicionando o score nas listas
    Score_Frase.append( Gerando_Score )

    # Condição sobre o score

    # Caso maior que 0 - Positivo
    if Gerando_Score > 0:
        Finale.append('positive')

    # Caso menor que 0 - Negativo
    elif Gerando_Score < 0:
        Finale.append('negative')

    # Caso contrario Neutro
    else:
        Finale.append('neutral')

# Atribuindo as listas na base de dados
Base_Filtrada['Score_Frase'] = Score_Frase
Base_Filtrada['Finale_Blob'] = Finale
```

```
[60] # Verificando os registros
Base_Filtrada[['text', 'Score_Frase', 'Finale_Blob']].head()
```

	text	Score_Frase	Finale_Blob
0	virginamerica what dhepburn said	0.00000	neutral
1	virginamerica plus you have added commercials ...	0.00000	neutral
2	virginamerica i did not today must mean i need...	-0.31250	negative
3	virginamerica it is really aggressive to blast...	0.00625	positive
4	virginamerica and it is a really big bad thing...	-0.35000	negative

Vamos avaliar o modelo

```
[61] # Vamos avaliar o quanto a Lib TextBlob tem de acurácia no seu modelo

# Função para metricas de classificação
from sklearn.metrics import classification_report

# Avaliando a performance da Lib NTL
print('Avaliando a performance da TextBlob', '\n')
print( classification_report(
    Base_Filtrada['airline_sentiment'],
    Base_Filtrada['Finale_Blob'] ) )
```

Avaliando a performance da TextBlob

	precision	recall	f1-score	support
negative	0.88	0.35	0.50	9178
neutral	0.32	0.57	0.41	3099
positive	0.32	0.75	0.45	2363
accuracy			0.46	14640
macro avg	0.51	0.56	0.45	14640
weighted avg	0.67	0.46	0.48	14640

A Acurácia não foi das melhores novamente hahaha



Vamos utilizar outras Biblioteca

```
[65] pip install transformers

Collecting transformers
  Downloading transformers-4.9.2-py3-none-any.whl (2.6 MB)
    |████████████████████████████████████████| 2.6 MB 12.2 MB/s
```

```
[67] # Vamos testar agora com a pipeline
#Piple é uma lib para processamento de dados textuais

# Função para mineração de emoção
from transformers import pipeline

# Definindo a função
Função_Classificação = pipeline('sentiment-analysis')

# Aplicando em uma frase para identificar o sentimento
Função_Classificação('i hate chocolate')[0]['label']

'NEGATIVE'
```

```
[71] # Aplicando a classificação nos textos
Base_Filtrada['Classificação_Transformer'] = Base_Filtrada['text'].apply(
    lambda Frase: Função_Classificação(Frase)[0]['label'].lower() )

# Vamos retirar os sentimentos neutros
Base_Filtrada_Cortada = Base_Filtrada[
    Base_Filtrada['airline_sentiment'] != 'neutral']
```

Vamos avaliar o modelo

```
[87] # Vamos avaliar o quanto a Lib TextBlob tem de acurácia no seu modelo

# Função para metricas de classificação
from sklearn.metrics import classification_report

# Avaliando a performance da Lib NTL
print('Avaliando a performance da Transformes', '\n')
print( classification_report(
    Base_Filtrada_Cortada['airline_sentiment'], Base_Filtrada_Cortada['Classificação_Transformer'] ) )
```

Avaliando a performance da Transformes

	precision	recall	f1-score	support
negative	0.86	0.97	0.91	33
positive	0.96	0.81	0.88	27
accuracy			0.90	60
macro avg	0.91	0.89	0.90	60
weighted avg	0.91	0.90	0.90	60

A Acurácia foi boa, porém nessa biblioteca apenas é considerado “Positivo” ou “Negativo”, logico que fica mais fácil assertar a categoria quanto se tem menos opções de classificações.

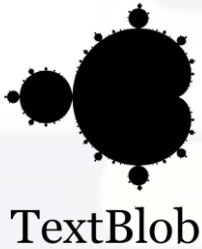


Final

Esse guia foi elaborada para demonstrar como analisar sentimentos usando alguns frameworks.

Link do Colab

https://colab.research.google.com/drive/1FVxU-zKS_K1gbWWXDlap0chjkgpnGIBL?usp=sharing



Odemir Depieri Jr

Data Intelligence Analyst Sr
Tech Lead
Specialization AI