Imports

```
#imports
%reload_ext google.colab.data_table
from google.colab import data_table
from google.colab import files
from apiclient.discovery import build


import pandas as pd
import numpy as np

import gspread
from oauth2client.service_account import ServiceAccountCredentials
!pip install gspread

import time

#instalando biblioteca
!pip install youtube_transcript_api
from youtube_transcript_api import YouTubeTranscriptApi
import datetime

#Importando bibliotecas world cloud
import re
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from os import path
import matplotlib.pyplot as plt

#Parte de Texto (HuggingFace)
!pip install transformers
!pip install huggingface
!pip install -U spacy
!python -m spacy download en_core_web_sm
!pip install spacy
!python -m spacy download pt_core_news_sm

from transformers import AutoTokenizer, AutoModel
from transformers import AutoTokenizer, AutoModelForSequenceClassification
from transformers import AutoTokenizer, AutoModel,BertTokenizer, BertForSequenceClassification,pipeline
import spacy
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: gspread in /usr/local/lib/python3.7/dist-packages (3.4.2)
Requirement already satisfied: google-auth in /usr/local/lib/python3.7/dist-packages (from gspread) (1.35.0)
Requirement already satisfied: requests>=2.2.1 in /usr/local/lib/python3.7/dist-packages (from gspread) (2.23.0)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests>=2.2.1->gspr
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests>=2.2.1->gspread)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests>=2.2.1->gsp
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth->gspread) (4.
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-packages (from google-auth->gspread) (1.15.
Requirement already satisfied: setuptools>=40.3.0 in /usr/local/lib/python3.7/dist-packages (from google-auth->gspread
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-auth->gspr
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from google-auth->gsp
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (from pyasn1-modules>=0.
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: youtube_transcript_api in /usr/local/lib/python3.7/dist-packages (0.4.4)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from youtube_transcript_api) (2.23.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->youtube_tr
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->youtube_tra
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->youtube_transcri
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: transformers in /usr/local/lib/python3.7/dist-packages (4.21.3)
Requirement already satisfied: huggingface-hub<1.0,>=0.1.0 in /usr/local/lib/python3.7/dist-packages (from transformer
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (2022.6
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.7/dist-packages (from transformers) (6.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages (from transformers) (21.3)
Requirement already satisfied: tokenizers!=0.11.3,<0.13,>=0.11.1 in /usr/local/lib/python3.7/dist-packages (from trans
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (1.21.6)
```

```
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-packages (from transformers) (3.8.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from transformers) (2.23.0)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packages (from transformers) (4.12.
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-packages (from transformers) (4.64.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.7/dist-packages (from huggingface-
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging>=20.
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->transform
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->transformers) (2
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->transformer
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->transforme
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: huggingface in /usr/local/lib/python3.7/dist-packages (0.0.1)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: spacy in /usr/local/lib/python3.7/dist-packages (3.4.1)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.7/dist-packages (from spacy) (2.0.8)
Requirement already satisfied: typing-extensions<4.2.0,>=3.7.4 in /usr/local/lib/python3.7/dist-packages (from spacy)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from spacy) (57.4.0)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.10.0,>=1.7.4 in /usr/local/lib/python3.7/dist-packages (from s
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages (from spacy) (2.11.3)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.9 in /usr/local/lib/python3.7/dist-packages (from spacy) (3.0.
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.7/dist-packages (from spacy) (2.23.0)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from spacy) (1.0
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.7/dist-packages (from spacy) (3.3.0)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.7/dist-packages (from spacy) (1.0.8
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.7/dist-packages (from spacy) (2.4.4)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.7/dist-packages (from spacy) (4.64.0)
Requirement already satisfied: wasabi<1.1.0,>=0.9.1 in /usr/local/lib/python3.7/dist-packages (from spacy) (0.10.1)
```

```
pip install dlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: dlib in /usr/local/lib/python3.7/dist-packages (19.24.0)
```

```
!pip3 install face_recognition
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: face_recognition in /usr/local/lib/python3.7/dist-packages (1.3.0)
Requirement already satisfied: Click>=6.0 in /usr/local/lib/python3.7/dist-packages (from face_recognition) (7.1.2)
Requirement already satisfied: Pillow in /usr/local/lib/python3.7/dist-packages (from face_recognition) (7.1.2)
Requirement already satisfied: dlib>=19.7 in /usr/local/lib/python3.7/dist-packages (from face_recognition) (19.24.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from face_recognition) (1.21.6)
Requirement already satisfied: face-recognition-models>=0.3.0 in /usr/local/lib/python3.7/dist-packages (from face_recog
```

```
!pip install -U fer
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: fer in /usr/local/lib/python3.7/dist-packages (22.4.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from fer) (2.23.0)
Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.7/dist-packages (from fer) (4.6.0.66)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from fer) (4.64.0)
Requirement already satisfied: keras>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from fer) (2.8.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (from fer) (3.2.2)
Requirement already satisfied: mtcnn>=0.1.1 in /usr/local/lib/python3.7/dist-packages (from fer) (0.1.1)
Requirement already satisfied: opencv-python>=4.1.0 in /usr/local/lib/python3.7/dist-packages (from mtcnn>=0.1.1->fer) (
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python>=4.1.0->mtcnn
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->fer) (1.4.4
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib->fer) (0.11.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->fer) (2.
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->matp
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1->matplotlib
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from r
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->fer) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->fer) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->fer) (2022.6
```

```
!pip install --upgrade youtube_dl
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: youtube_dl in /usr/local/lib/python3.7/dist-packages (2021.12.17)
```

```
from fer.fer import FER
import cv2
from fer import Video
from fer import FER
from matplotlib import pyplot as plt
import numpy as np
import glob
from os import listdir
from os.path import isfile, join
```

```
import numpy
import os
import face_recognition
from google.colab.patches import cv2_imshow
import pandas as pd
```

Lista de ID´s vídeos

```
ids_videos = [
 'prkZ-s8jP5g',
 '2WesQczDivs',
 '8fPswf4DWQs',
 'oVIJD_tuRPY',
 'qwdWfoZ9LhA',
 '4QBcnZ7ty9A&t=85s',
 't0WiAn_TIkI',
 'Ei76ULTV7Rk',
 'QlV3fOz2zto',
 'RyakNVXt_M0&t=18s',
 'cakyAaARAaw',
 'Tiktok',
 'UiA3SeSXHyg',
 'T91JRKQzDxw',
 'p1O58fm5pGI',
 'wtr0EGO-_7w',
 'Re5yvOVbC5I',
 '-1O2jmZ7B9s',
 'FPaXLhgIMzE',
 '1n9injcF-7k',
 'Q7u3aueKfaA',
 'cnmankDlTbc',
 'y_aWXEgxNnA',
 'KElAO-bIk0w',
 'hFlTNRLmRbM']
```

Funções e Métodos *

- Só executar

```
def put_data(dataList, worksheet, cellInterval):
  # add valores de viewCount no sheets
  cell_list = worksheet.range(cellInterval)

  for i in range(len(cell_list)):
    cell_list[i].value = dataList[i]

  # Update in batch
  return worksheet.update_cells(cell_list)


def get_values_from(worksheet, cellInteval):
  # Acessando os valores de ids diretamente da planilha
  cell_list = worksheet.range(cellInteval)
  data_fromSheets = []

  for i in range(len(cell_list)):
    data_fromSheets.append(cell_list[i].value)

  return data_fromSheets

def buscar_canais_por_nome(nome, youtube):
  request = youtube.search().list(q=nome, part='snippet', type='channel', maxResults=10)
  response = request.execute()
  return response

def criar_nova_planilha(nome, numLinhas, numColumn, shClient):
  #cria uma pagina para cada uma nova página na planilha
  shClient.add_worksheet(nome, numLinhas, numColumn)
```

```
def cria_varias_planilhas(listNomes, numLinhas, numColumn, shClient):
  #cria uma tabela para cada canal
  for i in range(len(listNomes)):
    shClient.add_worksheet('{}'.format(listNomes[i]), 100, 10)
  print('criou')

def buscador_video_2(idVideo):

    arquivo = open("{}.txt".format(idVideo), "a")

    df_all_datas_ch = pd.DataFrame()
    dados_do_video = youtube.search().list(q=[idVideo], part='snippet', type='video').execute()

    #extração de dados desejados
    lista_videosID = [dados_do_video['items'][0]['id']['videoId']]
    publishedAt = [dados_do_video['items'][0]['snippet']['publishedAt']]
    titles_videos = [dados_do_video['items'][0]['snippet']['title']]
    channelsName = [dados_do_video['items'][0]['snippet']['channelTitle']]
    descriptionList = [dados_do_video['items'][0]['snippet']['description']]

    return dados_do_video

def videoID_(i):
  video = ids_videos[i-1]


  return video,i



#busca pelo video, com base no ID passado
#extrai caracteristicas desejadas (title, desc, numviews...)
#salva num csv

def buscador_video_dataframe(idVideo):

    arquivo = open("{}.txt".format(idVideo), "a")

    df_all_datas_ch = pd.DataFrame()
    dados_do_video = youtube.search().list(q=[idVideo],part='snippet', type='video').execute()

    #extração de dados desejados
    lista_videosID = [dados_do_video['items'][0]['id']['videoId']]
    publishedAt = [dados_do_video['items'][0]['snippet']['publishedAt']]
    titles_videos = [dados_do_video['items'][0]['snippet']['title']]
    channelsName = [dados_do_video['items'][0]['snippet']['channelTitle']]
    descriptionList = [dados_do_video['items'][0]['snippet']['description']]

    #Call the videos.list method to retrieve statistics details for each video.
    video_statistic = [youtube.videos().list(id=idVideo, part='statistics').execute()]

    try:
       # Select a likesCount_list
        likesCount_list = [video_statistic[0]['items'][0]['statistics']['likeCount']]
    except:
        likesCount_list = ['indisponivel']

    try:
        # Select a views count
        viewsCount_list = [video_statistic[0]['items'][0]['statistics']['viewCount']]
    except:
        viewsCount_list = ['indisponivel']

    try:
        # Select a commentCount_list
        commentCount_list = [video_statistic[0]['items'][0]['statistics']['commentCount']]
    except:
        commentCount_list = ['indisponivel']

    try:
        # Select a dislikeCount_list
        dislikeCount_list = [video_statistic[0]['items'][0]['statistics']['dislikeCount']]

    except:
        dislikeCount_list = ['indisponivel']
```

```
    df_search = {
        'id': lista_videosID,
        'title': titles_videos,
        'channel': channelsName,
        'date_p': publishedAt,
        'description': descriptionList,
        'views': likesCount_list,
        'likes': viewsCount_list,
        'dislikes': dislikeCount_list,
        'comments': commentCount_list
      }

    columns = [
        'id',
        'title',
        'channel',
        'date_p',
        'description',
        'views',
        'likes',
        'dislikes',
        'comments'
      ]

    df_search = pd.DataFrame(df_search, columns=columns)
    print(df_search.shape)
    df_all_datas_ch = df_all_datas_ch.append(df_search)
    return df_all_datas_ch

#formata o valor de segundos para hh:mm:ss
def format_time(segundo):
    return str(datetime.timedelta(seconds=float(segundo)))
```

Módulo 1:

- Extração de dados
- Organização de dados
- Permanencia de dados

Conectando com API

- Só executar
  - youtube-v3
  - google SpreadSheets

Youtube v3

```
API_KEY = "AIzaSyBZXYtvA72pyk8dCCdid_fPmQm7pxbj4Q4" #chave de autenticação do projeto, para acesso do

#Construct a Resource object for interacting with an API. The serviceName and version are the names from the
youtube = build('youtube', 'v3', developerKey=API_KEY)

type(youtube)
```

```
    googleapiclient.discovery.Resource
```

Video Recovery - Busca de Videos

- busca de um único video
- extração de dados desejados

Novas Buscas

Processo:

- 1. busca de dados, pelo id do vídeo

- 2. get captions

```
i = int(input("Insira o número do vídeo: "))
id,indice = videoID_(i)
videoid = id
videoData = buscador_video_dataframe(videoid)
videoData
```

```
Insira o número do vídeo: 18
(1, 9)
```

1 entry Filter

index	id	title	channel	date_p	description	views	likes	dislikes	comments
0	-1O2jmZ7B9s	TEMOS QUE VALORIZAR A BIODIVERSIDADE DA AMAZÔNIA	Lula	2021-07-12T21:02:29Z	Nós temos plena consciência de que hoje não podemos discutir nenhum modelo de desenvolvimento sem levar em conta a ...	1075	6239	indisponivel	98

downloand dos dados em csv

```
from google.colab import files
videoData.to_csv('video' + str(indice) + '_metadados.csv',index=False)
files.download('video' + str(indice) + '_metadados.csv')
```

Get Captions:

```
try:
    #acessa o caption do video e add na lista de allCaptions
    caption = YouTubeTranscriptApi.get_transcript(videoid,languages=["pt"])
except:
    #caso não tenha caption

    print("Caption desativado")
df_caption = pd.DataFrame(caption)
df_caption.insert(2, 'tempo_inicial (h:m:s)', df_caption.start.apply(format_time)) # formatando a coluna do t
df_caption = df_caption.drop(['start'],axis=1) # removendo a coluna start

pd.DataFrame(caption)
```

1 to 25 of 37 entries Filter

index	text	start	duration
0	o teu o teu na minha bagagem política e	0.0	5.91
1	do meu legado político um prazer e	3.72	4.32
2	orgulho de Poder Dizer para vocês assim	5.91	4.41
3	entrevista quilo governo do PT nós	8.04	3.889
4	diminuímos oitenta por cento	10.32	3.989
5	desmatamento da Amazônia e que nós	11.929	5.131
6	Assumimos um compromisso no encontro de	14.309	5.31
7	copenhague-2009 que que a gente iria	17.06	4.21
8	cuidar da questão ambiental e o Brasil	19.619	3.601
9	virou referência no mundo até o encontro	21.27	4.74
10	de pares nós temos consciência de que	23.22	5.25
11	hoje você não pode discutir nenhum	26.01	4.859
12	modelo de desenvolvimento se você não	28.47	4.56
13	levar em conta a questão ambiental e o	30.869	6.241
14	Brasil tem o privilégio de 360 milhões	33.03	7.049
15	de hectares de floresta tropical e uma	37.11	6.3
16	Amazônia Legal extraordinária e que você	40.079	5.341
17	não pode imaginar que você para ganhar	43.41	3.93
18	dinheiro paciente mover você tem que	45.42	5.4
19	desmatar o que você precisa é utilizar a	47.34	5.789
20	biodiversidade da Amazônia você pode	50.82	4.59
21	fazer parceria com quase todos os países	53.129	5.16
22	do mundo que a gente utilizar a empresa	55.41	4.6
23	da biodiversidade da Amazônia	58.289	4.151
24	a desenvolver a Amazônia para ajudar	60.01	5.369

Show 25 per page

1 2

Módulo 2:

- Manibulação de dados
- Visualização dos dados
- Interpretação dos dados

filtrando captions

```
df_caption
#start(s): momento em que o fragmento text{...} começa a ser falado
```

1 to 25 of 37 entries Filter

index	text	tempo_inicial (h:m:s)	duration
0	o teu o teu na minha bagagem política e	0:00:00	5.91
1	do meu legado político um prazer e	0:00:03.720000	4.32
2	orgulho de Poder Dizer para vocês assim	0:00:05.910000	4.41
3	entrevista quilo governo do PT nós	0:00:08.040000	3.889
4	diminuímos oitenta por cento	0:00:10.320000	3.989
5	desmatamento da Amazônia e que nós	0:00:11.929000	5.131
6	Assumimos um compromisso no encontro de	0:00:14.309000	5.31
7	copenhague-2009 que que a gente iria	0:00:17.060000	4.21
8	cuidar da questão ambiental e o Brasil	0:00:19.619000	3.601
9	virou referência no mundo até o encontro	0:00:21.270000	4.74
10	de pares nós temos consciência de que	0:00:23.220000	5.25
11	hoje você não pode discutir nenhum	0:00:26.010000	4.859
12	modelo de desenvolvimento se você não	0:00:28.470000	4.56
13	levar em conta a questão ambiental e o	0:00:30.869000	6.241
14	Brasil tem o privilégio de 360 milhões	0:00:33.030000	7.049
15	de hectares de floresta tropical e uma	0:00:37.110000	6.3
16	Amazônia Legal extraordinária e que você	0:00:40.079000	5.341
17	não pode imaginar que você para ganhar	0:00:43.410000	3.93
18	dinheiro paciente mover você tem que	0:00:45.420000	5.4
19	desmatar o que você precisa é utilizar a	0:00:47.340000	5.789
20	biodiversidade da Amazônia você pode	0:00:50.820000	4.59
21	fazer parceria com quase todos os países	0:00:53.129000	5.16
22	do mundo que a gente utilizar a empresa	0:00:55.410000	4.6
23	da biodiversidade da Amazônia	0:00:58.289000	4.151
24	a desenvolver a Amazônia para ajudar	0:01:00.010000	5.369

Show 25 per page 1 2

WorldCloud

```
#Importando bibliotecas
import re
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from os import path
import matplotlib.pyplot as plt


# Quando precisar dos dados COM os T´s executar SOMENTE ESTA!
data = ""
j=0
for i in df_caption.text.values:
    data = data + " [Ti_" + str(j) + "] " + i #+ " [Tf_" + str(j) +"]"
    j += 1

#Ti_x (tempo inicial x)


# Quando precisar dos dados SEM os T´s executar SOMENTE ESTA!
data = ""
j=0
for i in df_caption.text.values:
    data = data +  i #+ " [Tf_" + str(j) +"]"
    j += 1

#Ti_x (tempo inicial x)
```

Vizualização que facilita a leitura

stopword

remove palavras lixo

```
#Definindo a lista de stopwords
!gdown 'https://drive.google.com/uc?id=1cTHaENpsFyJz91lUY_HZkP3maR_Rj3f3'
pathfileStop = '/content/stopwords_portuguese.txt'
with open(pathfileStop, 'a') as f:
  f.write('\ntext'+"'"+'\n')
  f.write('duration'+"'"+'\n')
  f.write('start'+"'"+'\n')
  f.write('text'+"'"+'\n')
  f.write('do'+"'"+'\n')
  f.write('por'+"'"+'\n')
  f.write('tá'+"'"+'\n')
  f.write('das'+"'"+'\n')
  f.write('ele'+"'"+'\n')
  f.write('mas'+"'"+'\n')
  f.write('ou'+"'"+'\n')
  f.write('foi'+"'"+'\n')
  f.write('da'+"'"+'\n')
  f.write('nós'+"'"+'\n')
  f.write('é'+"'"+'\n')
  f.write('e'+"'"+'\n')
  f.write('não'+"'"+'\n')
  f.write('tem'+"'"+'\n')
  f.write('já'+"'"+'\n')
  f.write('também'+"'"+'\n')
  f.write("'"+'\n')
  f.write('então'+'\n')
  f.write('pessoa'+'\n')
  f.write('aí'+'\n')
  f.write('ainda'+'\n')
  f.write('que'+"'"+'\n')
  f.write('agora'+'\n')
  f.write('assim'+'\n')
  f.write('vai'+'\n')
  f.write('que'+'\n')
  f.write('aqui'+'\n')
  f.write('tá'+'\n')
  f.write('todo'+'\n')
  f.write('coisa'+'\n')
  f.write('né'+'\n')
  f.write('tudo'+'\n')
  f.write('lá'+'\n')
  f.write('outro'+'\n')
  f.write('hora'+'\n')
  f.write('tão'+'\n')
  f.write('a'+"'"+'\n')
  f.write('ea'+"'"+'\n')
  f.write('para'+"'"+'\n')
  f.write('o'+"'"+'\n')
  f.write('com'+"'"+'\n')
  f.write('né'+"'"+'\n')
  f.write('uma'+"'"+'\n')
  f.write('um'+"'"+'\n')
  f.write('na'+"'"+'\n')
  f.write('de'+"'"+'\n')
  f.write('como'+"'"+'\n')
  f.write('paulo no'+"'"+'\n')
  f.write('aqui'+"'"+'\n')
  f.write('essa'+"'"+'\n')
  f.write('gente'+"'"+'\n')
  f.write('mais'+"'"+'\n')
  f.write('se'+"'"+'\n')
  f.write('em'+"'"+'\n')
  f.write('aí'+"'"+'\n')
  f.write('muito'+"'"+'\n')
  f.write('você'+"'"+'\n')
  f.write('esse'+"'"+'\n')
  f.write('agora'+"'"+'\n')
```

```
    f.write('agora'+"'"+'\n')
    f.write('que'+"'"+'\n')
    f.write('então'+"'"+'\n')
    f.write('isso'+"'"+'\n')
    f.write('lá'+"'"+'\n')
    f.write('eu'+"'"+'\n')
    f.write('as'+"'"+'\n')
    f.write('dos'+"'"+'\n')
    f.write('brasil'+"'"+'\n')
    f.write('só'+"'"+'\n')
    f.write('os'+"'"+'\n')
    f.write('ser'+"'"+'\n')
    f.write('pessoas'+"'"+'\n')
    f.write('vão'+"'"+'\n')
    f.write('são'+"'"+'\n')
    f.write('ela'+"'"+'\n')
    f.write('porque'+"'"+'\n')
    f.write('ser'+"'"+'\n')
    f.write('vai'+"'"+'\n')
    f.write('ea'+'\n')
    f.write('até'+'\n')


    f.close

print('terminou')
```

```
Downloading...
From: https://drive.google.com/uc?id=1cTHaENpsFyJz91lUY_HZkP3maR_Rj3f3
To: /content/stopwords_portuguese.txt
100% 1.56k/1.56k [00:00<00:00, 2.80MB/s]
terminou
```

```
#Definindo a lista de stopwords
!gdown 'https://drive.google.com/uc?id=1cTHaENpsFyJz91lUY_HZkP3maR_Rj3f3'
pathfileStop = '/content/stopwords_portuguese.txt'
with open(pathfileStop, 'a') as f:
  f.write('\ntext'+"'"+'\n')
  f.write('duration'+"'"+'\n')
  f.write('start'+"'"+'\n')
  f.write('text'+"'"+'\n')
  f.write('do'+"'"+'\n')
  f.write('por'+"'"+'\n')
  f.write('tá'+"'"+'\n')
  f.write('das'+"'"+'\n')
  f.write('ele'+"'"+'\n')
  f.write('mas'+"'"+'\n')
  f.write('ou'+"'"+'\n')
  f.write('foi'+"'"+'\n')
  f.write('da'+"'"+'\n')
  f.write('nós'+"'"+'\n')
  f.write('é'+"'"+'\n')
  f.write('e'+"'"+'\n')
  f.write('não'+"'"+'\n')
  f.write('tem'+"'"+'\n')
  f.write('já'+"'"+'\n')
  f.write('também'+"'"+'\n')
  f.write("'"+'\n')
  f.write('então'+'\n')
  f.write('pessoa'+'\n')
  f.write('aí'+'\n')
  f.write('ainda'+'\n')
  f.write('que'+"'"+'\n')
  f.write('agora'+'\n')
  f.write('assim'+'\n')
  f.write('vai'+'\n')
  f.write('que'+'\n')
  f.write('aqui'+'\n')
  f.write('tá'+'\n')
  f.write('todo'+'\n')
  f.write('coisa'+'\n')
  f.write('né'+'\n')
  f.write('tudo'+'\n')
  f.write('lá'+'\n')
  f.write('outro'+'\n')
```

```
    f.write('hora'+'\n')
    f.write('tão'+'\n')
    f.write('a'+"'"+'\n')
    f.write('ea'+"'"+'\n')
    f.write('para'+"'"+'\n')
    f.write('o'+"'"+'\n')
    f.write('com'+"'"+'\n')
    f.write('né'+"'"+'\n')
    f.write('uma'+"'"+'\n')
    f.write('um'+"'"+'\n')
    f.write('na'+"'"+'\n')
    f.write('de'+"'"+'\n')
    f.write('como'+"'"+'\n')
    f.write('paulo no'+"'"+'\n')
    f.write('aqui'+"'"+'\n')
    f.write('essa'+"'"+'\n')
    f.write('gente'+"'"+'\n')
    f.write('mais'+"'"+'\n')
    f.write('se'+"'"+'\n')
    f.write('em'+"'"+'\n')
    f.write('aí'+"'"+'\n')
    f.write('muito'+"'"+'\n')
    f.write('você'+"'"+'\n')
    f.write('esse'+"'"+'\n')
    f.write('agora'+"'"+'\n')
    f.write('que'+"'"+'\n')
    f.write('então'+"'"+'\n')
    f.write('isso'+"'"+'\n')
    f.write('lá'+"'"+'\n')
    f.write('eu'+"'"+'\n')
    f.write('as'+"'"+'\n')
    f.write('dos'+"'"+'\n')
    f.write('brasil'+"'"+'\n')
    f.write('só'+"'"+'\n')
    f.write('os'+"'"+'\n')
    f.write('ser'+"'"+'\n')
    f.write('pessoas'+"'"+'\n')
    f.write('vão'+"'"+'\n')
    f.write('são'+"'"+'\n')
    f.write('ela'+"'"+'\n')
    f.write('porque'+"'"+'\n')
    f.write('ser'+"'"+'\n')
    f.write('vai'+"'"+'\n')
    f.write('ea'+'\n')
    f.write('até'+'\n')
    f.write('Música'+'\n')


    f.close

print('terminou')
```

```
Downloading...
From: https://drive.google.com/uc?id=1cTHaENpsFyJz91lUY_HZkP3maR_Rj3f3
To: /content/stopwords_portuguese.txt
100% 1.56k/1.56k [00:00<00:00, 2.59MB/s]
terminou
```

```
stopwords= set(STOPWORDS)

#Adicionando a lista stopwords em português
new_words = []
#with open("/stopwords_portuguese.txt", 'r') as f:
with open(pathfileStop, 'r') as f:
    [new_words.append(word) for line in f for word in line.split()]

new_stopwords = stopwords.union(new_words)


plt.figure(figsize=(20,10))
wc = WordCloud(min_font_size=20,
               max_font_size=300,
               background_color='white',
               mode="RGB",
               stopwords= new_stopwords,
```

```
                width=2000,
                height=1000,
                normalize_plurals= True).generate(data)


#plt.title("Wordcloud", fontsize=40, color="red")
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
plt.show()
```

Módulo 3:

Análise Morfossintática

```
# pip install -U spacy
# python -m spacy download en_core_web_sm
import spacy
from spacy.lang.pt.examples import sentences

# Carregando elementos de NLP em portugês
nlp = spacy.load("pt_core_news_sm")

# Processand
text = ("Eu vou lanchar")
doc = nlp(text)

# Analyze syntax
print("Noun phrases:", [chunk.text for chunk in doc.noun_chunks])
print("Verbs:", [token.lemma_ for token in doc if token.pos_ == "VERB"])

# Find named entities, phrases and concepts
for entity in doc.ents:
    print(entity.text, entity.label_)
```

```
    Noun phrases: ['Eu']
    Verbs: ['lanchar']
```

```
from google.colab import files
try:
    #acessa o caption do video e add na lista de allCaptions
    caption = YouTubeTranscriptApi.get_transcript(videoid,languages=["pt"])
except:
    #caso não tenha caption
    print("Caption desativado")
df_caption = pd.DataFrame(caption)
```

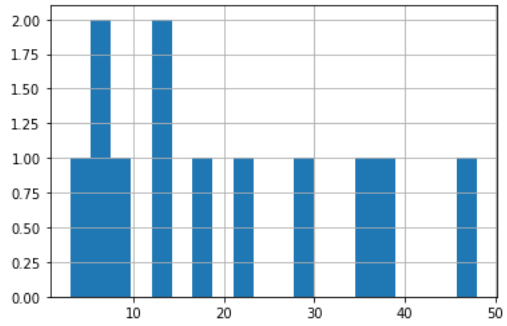
```
df_caption.insert(2, 'tempo_inicial (h:m:s)', df_caption.start.apply(format_time)) # formatando a coluna do t
df_caption = df_caption.drop(['start'],axis=1) # removendo a coluna start

#==============================Preparação do modelo Classificador FinBertPTBR=========================

tokenizer_ptbr = BertTokenizer.from_pretrained("turing-usp/FinBertPTBR")
model_ptbr = BertForSequenceClassification.from_pretrained("turing-usp/FinBertPTBR")
classifier_ptbr = pipeline('text-classification', model = model_ptbr, tokenizer= tokenizer_ptbr)

#==============================Aplicação do FinBertPTBR nos fragmentos de texto dos frames==================

Texto = ""
for i in range(len(df_caption.text.values)):
  Texto = Texto + df_caption.text.values[i] + " "



df_ptbr = [Texto]
tab_ptbr = pd.DataFrame(df_ptbr)
resultados_ptbr = []
for i in range(len(tab_ptbr.values)):
  resultados_ptbr.append(classifier_ptbr(tab_ptbr.values[i][0]))

# Tabela com análise morfossintática das frases ditas em cada 'quadro de vídeo'
frases = []
for i in df_caption.text.values:  # Fragmentos de texto em portugês
  frases.append(i)
results = []
lista_ = []
for v in range(len(frases)):
# Carregando elementos de NLP em portugês
  nlp = spacy.load("pt_core_news_sm")
  text = (frases[v])
  doc = nlp(text)
  lista_ = []
  for token in doc:
      lista_.append([token.text, token.pos_])


  lista= np.matrix(lista_).transpose()
  lista[0,0] + " = " +  lista[1,0]
  resultado = []
  for i in range(lista.shape[1]):
    resultado.append(lista[0,i] + " = " + lista[1,i])

  frase_ =""
  for j in range(len(resultado)):
    frase_ = frase_ + resultado[j] + " , "

  results.append([[frases[v]],[frase_]])


#captions_video18 = pd.DataFrame(results)
#captions_video18.to_excel('Análise_Linguística-Video18.xlsx')

#Tabela geral com análise do sentimento, análise morfossintática
tab_ptbr = []
for i in range(len(frases)):
  df_ptbr = frases[i]
  tab_ptbr.append(df_ptbr)

resultados_ptbr = []

for i in range(len(tab_ptbr)):
  resultados_ptbr.append(classifier_ptbr(tab_ptbr[i]))

columns = ('Fragmentos de Caption', 'Dados para Análise Morfossintática', 'Dados para Análise de Polaridade/E
for i in range(len(resultados_ptbr)):
  results[i].append(resultados_ptbr[i])

data = pd.DataFrame(results,columns =columns)
df_textclassific = data.join(df_caption.iloc[:,1:])
```

```
df_textclassific.to_csv('Análise Morfossintática e Polaridade.csv')
files.download("/content/Análise Morfossintática e Polaridade.csv")
```

```
classes_morfossintaticas = []
classes_morfossintaticas_ = []

for i in range(len(results)):
  for j in range(len(results[i][1][0].split(','))):

    if len(results[i][1][0].split(',')[j].split('=')) == 2:
      classes_morfossintaticas.append(results[i][1][0].split(',')[j].split('=')[1])

    else:

      classes_morfossintaticas_.append(results[i][1][0].split(',')[j].split('=')[0])
```

```
pd.DataFrame(classes_morfossintaticas).value_counts().hist(bins=20)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe663ee1510>
```



Módulo 4

Detecção de Polaridade

```
#================================Preparação do modelo Classificador FinBertPTBR==========================

tokenizer_ptbr = BertTokenizer.from_pretrained("turing-usp/FinBertPTBR")
model_ptbr = BertForSequenceClassification.from_pretrained("turing-usp/FinBertPTBR")
classifier_ptbr = pipeline('text-classification', model = model_ptbr, tokenizer= tokenizer_ptbr)

#================================Aplicação do FinBertPTBR nos fragmentos de texto dos frames==================
```

```
#tab_ptbr = pd.DataFrame(df_caption.text.values)
#resultados_ptbr = []
#for i in range(len(tab_ptbr.values)):
#  resultados_ptbr.append(classifier_ptbr(tab_ptbr.values[i][0]))
# Tabela com análise morfossintática das frases ditas em cada 'quadro de vídeo'
#Tabela geral com análise do sentimento, análise morfossintática
tab_ptbr = []
for i in range(len(frases)):
  df_ptbr = frases[i]
  tab_ptbr.append(df_ptbr)

resultados_ptbr = []

for i in range(len(tab_ptbr)):
  resultados_ptbr.append(classifier_ptbr(tab_ptbr[i]))
```

```
resultados_ptbr
```

```
[[{'label': 'NEUTRAL', 'score': 0.5295822620391846}],
 [{'label': 'POSITIVE', 'score': 0.42632123827934265}],
 [{'label': 'POSITIVE', 'score': 0.5236883759498596}],
 [{'label': 'NEUTRAL', 'score': 0.4094569981098175}],
 [{'label': 'POSITIVE', 'score': 0.4318948984146118}],
 [{'label': 'NEUTRAL', 'score': 0.4995131492614746}],
 [{'label': 'POSITIVE', 'score': 0.493160218000412}],
 [{'label': 'POSITIVE', 'score': 0.6828931570053101}],
 [{'label': 'NEUTRAL', 'score': 0.5964054465293884}],
 [{'label': 'NEUTRAL', 'score': 0.4179705083370209}],
 [{'label': 'NEUTRAL', 'score': 0.4536206126213074}],
 [{'label': 'NEUTRAL', 'score': 0.5471360683441162}],
 [{'label': 'NEUTRAL', 'score': 0.5551355481147766}],
 [{'label': 'POSITIVE', 'score': 0.5228729844093323}],
 [{'label': 'NEGATIVE', 'score': 0.45540493726730347}],
 [{'label': 'NEUTRAL', 'score': 0.576172173023239}],
 [{'label': 'POSITIVE', 'score': 0.4542777240276336 7}],
 [{'label': 'NEUTRAL', 'score': 0.5975528359413147}],
 [{'label': 'NEUTRAL', 'score': 0.6036877036094666}],
 [{'label': 'POSITIVE', 'score': 0.43885576725006104}],
 [{'label': 'NEUTRAL', 'score': 0.4863400459289551}],
 [{'label': 'NEUTRAL', 'score': 0.5024268627166748}],
 [{'label': 'NEGATIVE', 'score': 0.6140471696853638}],
 [{'label': 'POSITIVE', 'score': 0.42402184009552}],
 [{'label': 'NEUTRAL', 'score': 0.46254903078079224}],
 [{'label': 'POSITIVE', 'score': 0.6972026228904724}],
 [{'label': 'POSITIVE', 'score': 0.5736519694328308}],
 [{'label': 'NEUTRAL', 'score': 0.37460678815841675}],
 [{'label': 'NEUTRAL', 'score': 0.3731783628463745}],
 [{'label': 'POSITIVE', 'score': 0.4143470227718353}],
 [{'label': 'POSITIVE', 'score': 0.4405781626701355}],
 [{'label': 'POSITIVE', 'score': 0.5586836934089661}],
 [{'label': 'NEUTRAL', 'score': 0.40447670221328735}],
 [{'label': 'POSITIVE', 'score': 0.4927245080471039}],
 [{'label': 'NEUTRAL', 'score': 0.5722704529762268}],
 [{'label': 'NEUTRAL', 'score': 0.417916864156723}],
 [{'label': 'NEGATIVE', 'score': 0.49161940813064575}],
 [{'label': 'POSITIVE', 'score': 0.37415409088134766}],
 [{'label': 'NEUTRAL', 'score': 0.6611707806587219}],
 [{'label': 'NEUTRAL', 'score': 0.5895301103591919}],
 [{'label': 'NEUTRAL', 'score': 0.6936940550804138}],
 [{'label': 'NEUTRAL', 'score': 0.43281039595603943}],
 [{'label': 'POSITIVE', 'score': 0.5677844882011414}],
 [{'label': 'POSITIVE', 'score': 0.6326255202293396}],
 [{'label': 'NEUTRAL', 'score': 0.5033527612686157}],
 [{'label': 'NEUTRAL', 'score': 0.4414883553981781}],
 [{'label': 'POSITIVE', 'score': 0.37470728158950806}],
 [{'label': 'NEUTRAL', 'score': 0.380868375301361 1}],
 [{'label': 'NEGATIVE', 'score': 0.6773945689201355}],
 [{'label': 'POSITIVE', 'score': 0.4270518720149994}],
 [{'label': 'NEGATIVE', 'score': 0.4595257639884949}],
 [{'label': 'NEGATIVE', 'score': 0.510378897190094}],
 [{'label': 'NEGATIVE', 'score': 0.5518338680267334}],
 [{'label': 'POSITIVE', 'score': 0.36411044001579285}],
 [{'label': 'NEGATIVE', 'score': 0.4391593635082245}],
 [{'label': 'POSITIVE', 'score': 0.3798079192638397}],
 [{'label': 'NEGATIVE', 'score': 0.4357345402240753}],
 [{'label': 'NEGATIVE', 'score': 0.43183937668800354}],
```

Detecção de Emoção

[ ] ↳ 7 células ocultas

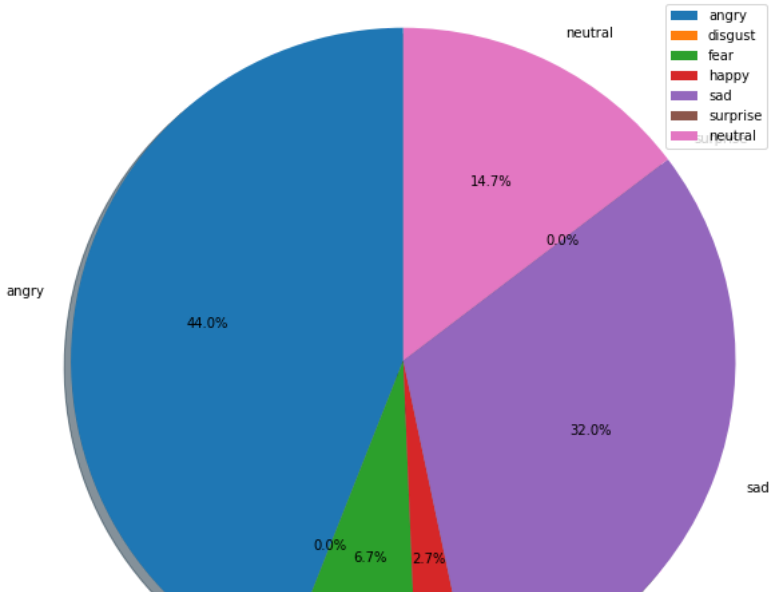
Geração do gráfico Pie

```
emocao_ = []
for i in range(len(emocao)):
  emocao_.append(emocao[i][0]['emotions'])


df_emotions = pd.DataFrame(emocao_)
df_emotions
```

1 to 25 of 76 entries Filter

index	angry	disgust	fear	happy	sad	surprise	neutral
0	0.7	0.01	0.04	0.0	0.16	0.0	0.09
1	0.5	0.1	0.01	0.0	0.38	0.0	0.01
2	0.34	0.29	0.03	0.0	0.32	0.0	0.02
3	0.01	0.0	0.06	0.13	0.46	0.0	0.34
4	0.33	0.01	0.06	0.06	0.34	0.0	0.21
5	0.42	0.12	0.03	0.01	0.21	0.0	0.22
6	0.73	0.03	0.03	0.0	0.19	0.0	0.03
7	0.6	0.0	0.02	0.0	0.17	0.0	0.21
8	0.61	0.01	0.06	0.0	0.31	0.0	0.02
9	0.89	0.01	0.08	0.0	0.02	0.0	0.0
10	0.48	0.01	0.13	0.0	0.38	0.0	0.0
11	0.85	0.01	0.01	0.0	0.06	0.0	0.06
12	0.6	0.0	0.03	0.01	0.26	0.0	0.09
13	0.82	0.0	0.05	0.01	0.1	0.0	0.02
14	0.36	0.0	0.03	0.07	0.37	0.0	0.17
15	0.45	0.0	0.13	0.06	0.2	0.09	0.09
16	0.78	0.01	0.01	0.0	0.14	0.0	0.07
17	0.14	0.0	0.08	0.05	0.34	0.11	0.28

```
angry = 0
disgust = 0
fear = 0
happy = 0
sad = 0
surprise = 0
neutral = 0

for i in range(len(emocao)):
  if df_emotions.angry[i]>df_emotions.disgust[i] and df_emotions.angry[i]>df_emotions.fear[i] and  df_emotion
    angry = angry + 1

  elif df_emotions.disgust[i]>df_emotions.angry[i] and df_emotions.disgust[i]>df_emotions.fear[i] and  df_emo
    disgust = disgust + 1

  elif df_emotions.fear[i]>df_emotions.angry[i] and df_emotions.fear[i]>df_emotions.disgust[i] and  df_emotio
    fear =fear + 1

  elif df_emotions.happy[i]>df_emotions.angry[i] and df_emotions.happy[i]>df_emotions.fear[i] and  df_emotion
    happy = happy + 1

  elif df_emotions.sad[i]>df_emotions.angry[i] and df_emotions.sad[i]>df_emotions.fear[i] and  df_emotions.sa
    sad = sad + 1

  elif df_emotions.neutral[i]>df_emotions.angry[i] and df_emotions.neutral[i]>df_emotions.fear[i] and  df_emo
    neutral = neutral + 1


  elif df_emotions.surprise[i]>df_emotions.angry[i] and df_emotions.surprise[i]>df_emotions.fear[i] and  df_e
    surprise = surprise + 1



import matplotlib.pyplot as plt

labels = 'angry', 'disgust', 'fear','happy','sad','surprise', 'neutral'

sizes = [angry, disgust,fear,happy,sad,surprise,neutral]

fig1, ax1 = plt.subplots(figsize=[10,10])

ax1.pie(sizes, labels = labels,autopct = "%.1f%%" ,shadow = True, startangle=90,center = (0,0))
plt.legend()
ax1.axis('equal')

plt.show()
```



Geração do DataFrame FER + polaridade

```
'''
0 - Angry
1 - Disgust
2 - fear
3 - happy
4- Sad
5 - Surprise
6 - neutral
'''
```

```
'''
lista = list(percent_emotions[0])
lista.sort(reverse=True)
lista
'''
```

```
    '\nlista = list(percent_emotions[0])\nlista.sort(reverse=True)\nlista\n'
```

```
ranking_emocoes = []
for i in range(len(emocao_)):
  ranking_emocoes.append({k: v for k, v in sorted(emocao_[i].items(), key=lambda item: item[1], reverse=True)
```

```
len(ranking_emocoes)
emocoes_ordenadas = []
for i in range(len(ranking_emocoes)):
  emocoes_ordenadas.append(ranking_emocoes[i])
```

```
from google.colab import files
FER = pd.DataFrame(np.transpose(emocoes_ordenadas))
Hugging_face = pd.DataFrame(resultados_ptbr)

FER.to_csv("FER-model.csv")
Hugging_face.to_csv("Hugging_face.csv")

files.download("FER-model.csv")
files.download("Hugging_face.csv")
```

```
len(resultados_ptbr)
```

```
    37
```

Face Recognition

[ ] ↳ *13 células ocultas*