**LAPORAN TUGAS CRAWLING DAN PREPROCESSING DATA TEXT**

**MATA KULIAH TEXT MINING & NATURAL LANGUAGE PROCESSING**

Logo

Description automatically generated

**Tim Penyusun:**

1. <5231811022> <Lathif Ramadhan>
2. <5231811029> <Andini Angel Meivita>
3. <5231811033> <Rama Panji Nararendra>
4. <5231811036> <Giffari Riyanda Pradithya>

**PROGRAM STUDI SAINS DATA PROGRAM SARJANA**

**FAKULTAS SAINS & TEKNOLOGI**

**UNIVERSITAS TEKNOLOGI YOGYAKARTA**

**2025**

1. **Hastag yang digunakan:** #ramadhan
2. **Tuliskan langkah-langkah crawling data text:**
3. **Tuliskan/screenshotkan codingnya**

**Jawab:**

**Link file kodenya:**

<https://colab.research.google.com/drive/1wxDKwRGiu_pd9CDx6kNHN6I8RfUdxLrS#scrollTo=o2wMKSTWhoNW>

# instal Node.js dan npm (Node Package Manager) di sistem

!curl -sL https://deb.nodesource.com/setup\_18.x | sudo -E bash -

!sudo apt-get install -y nodejs

# nama file dari data yang berhasil dikumpulkan

data = "ramadan\_1.csv"

# kata kunci dari data yang ingin dicari

search\_keyword = "Ramadan"

# limit baris pencarian

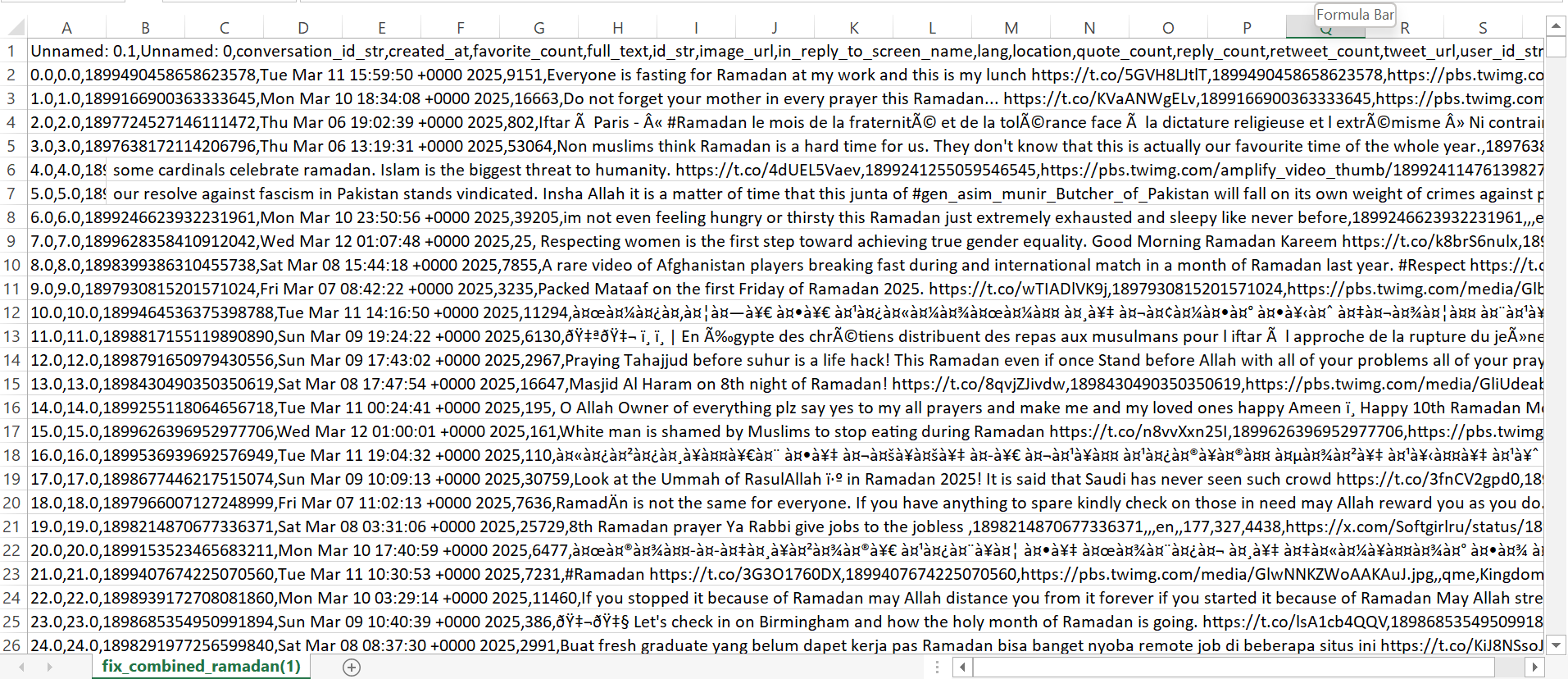
limit = 200

# jalankan proses crawling tweet dengan bantuan tweet-harvest menggunakan Twitter API token.

!npx --yes tweet-harvest@2.6.1 -o "ramadan\_1.csv" -s "{Ramadan}" -l {100} --token "----------------"

1. **Tuliskan/screenshotkan hasil crawlingnya**

**Jawab:**

****

1. **Berapa jumlah record yang didapatkan**

**Jawab: 6543 baris**

1. **Tuliskan langkah-langkah preprocessing data text:**

**Link Colab Coding Cleaning:**

<https://colab.research.google.com/drive/15PBqlsHWNYC5eV-kj9fXwbNN9nC9whRJ?usp=sharing>

**Link Colab Case Folding, Tokenizing, Filtering, Stemming, dan Simpan Data Bersih:**

<https://colab.research.google.com/drive/1pBv_cBklF6W0eB2FAZdkWwRCXI3_dIbM?usp=sharing#scrollTo=eW2NnAO87pD8>

1. **Tuliskan/screenshotkan coding cleaning**

**Jawab:**

import pandas as pd

import glob

import os

path = r'/content/' # use your path

from os import listdir

from os.path import isfile, join

onlyfiles = [f for f in listdir(path) if isfile(join(path, f))]

li = []

# print(all\_files)S

for filename in onlyfiles:

df = pd.read\_csv( r'/content/' + filename, index\_col=None, header=0)

li.append(df)

frame = pd.concat(li, axis=0, ignore\_index=True)

frame.to\_csv("fix\_combined\_ramadan\_real.csv", index=False)

df = pd.read\_csv('/content/fix\_combined\_ramadan\_real.csv')

df

df.drop\_duplicates()

df.to\_csv("/content/fix\_combined\_ramadan.csv", index=False)

Cleaning di preprocessing:

df.drop\_duplicates(subset = 'Tweet', keep = 'first', inplace = True)

# Memfilter datanya cuma bahasanya yang English

filt = (df\_twit['lang'] == 'en')

df\_twit = df\_twit.loc[filt, :]

df\_twit.reset\_index(inplace=True)

#menghilangkan mention/user

def remove\_pattern (tweet, pattern):

r = re.findall(pattern, tweet)

for i in r:

tweet = re.sub(i,'', tweet)

return tweet

df['remove\_user'] = np.vectorize(remove\_pattern)(df['teks'], "@[\w]\*")

df['remove\_user']

1. **Tuliskan/screenshotkan Case folding**

**Jawab:**

Case folding adalah proses mengubah semua teks menjadi huruf kecil agar lebih konsisten dalam analisis.

**→ Di kode ini, case folding terjadi secara otomatis saat tokenizing**

#tokenize tweets

tokenizer = TweetTokenizer(preserve\_case=True, strip\_handles=True, reduce\_len=True)

tweet\_tokens = tokenizer.tokenize(tweet)

# print(f"Word after tokenizer : {tweet\_tokens}")

tweets\_clean = []

for word in tweet\_tokens:

if (word not in stopwords\_english and word not in emoticons and word not in string.punctuation): # remove punctuation

# print(f"Word before stemming: {word}")

stem\_word = stemmer.stem(word) #stemming word

# print(f"Word after stem: {stem\_word}")

tweets\_clean.append(stem\_word)

return tweets\_clean

Parameter preserve\_case=False dalam TweetTokenizer akan secara otomatis mengubah teks menjadi huruf kecil.

Kedua kode ini walaupun fungsinya untuk tokenizing dan stemming tetapi secara tidak langsung mereka juga melakukan case folding, dimana untuk tokenizer preserve\_case nya di set ke False untuk di case folding.

Untuk bagian kode yang melakukan case folding di tokenizing adalah

tokenizer = TweetTokenizer(preserve\_case=True, strip\_handles=True, reduce\_len=True)

tweet\_tokens = tokenizer.tokenize(tweet)

Dan untuk bagian kode yang melakukan case folding di stemming adalah

stem\_word = stemmer.stem(word)

1. **Tuliskan/screenshotkan Tokenizing**

Tokenizing adalah proses memecah teks menjadi kata-kata atau token.

**Jawab:**

→ **Bagian kode yang melakukan tokenizing:**

tokenizer = TweetTokenizer(preserve\_case=False, strip\_handles=True, reduce\_len=True)

tweet\_tokens = tokenizer.tokenize(tweet)

* tokenizer.tokenize(tweet) akan memecah teks menjadi daftar kata-kata individu (token).
* preserve\_case=False akan membuat kata-kata menjadi huruf kecil (bagian dari case folding juga).

1. **Tuliskan/screenshotkan Filtering**

Filtering adalah proses membersihkan teks dari karakter atau kata yang tidak diperlukan, seperti angka, URL, tanda baca, stopwords, dan emotikon.

**Jawab:**

→ **Bagian kode yang melakukan filtering:**

def tweet\_clean(tweet):

#remove angka

tweet = re.sub('[0-9]+', '', tweet)

# print(f"ss")

# remove stock market tickers Like $GE

tweet = re.sub(r'\$\w\*', '', tweet)

# remove old style retweet text "RT"

tweet = re.sub(r'RT: [\s]+','', tweet)

#remove hyperlinks

tweet = re.sub(r'https?:\/\/.\*[\r\n]\*', '', tweet)

#remove coma

tweet = re.sub(r',','', tweet)

# remove hashtags

# only removing the hash # sign from the word

tweet = re.sub(r'#','', tweet)

#Happy Emoticons

emoticons\_happy = set([

':-)', ':)', ';)', ':0)', ':]', '3', ':c)', ':>', '=]', ':^)', ':-D', ':D' '8-D', '8D', '-3', '-3', ':-))', ":'-)" 'x-D', 'xD', 'X-D', 'XD' '>:P', 'x-p', 'xp', 'XP', ':-p', 'p', 'p', 'b', 'b', '>:)', '<3' ])

#Sad Emoticons

emoticons\_sad = set([

'L', ':-/', '>:/', 'S', '>:', '>:[','',':-(',':[', ':-||', ':-[', ':-<', '=\\', '=/', '>:(', ':(', '>.<', 'c', ':{', '>:\\', ';(' ])

#all emoticons (happy + sad)

emoticons = emoticons\_happy.union(emoticons\_sad)

#tokenize tweets

tokenizer = TweetTokenizer(preserve\_case=True, strip\_handles=True, reduce\_len=True)

tweet\_tokens = tokenizer.tokenize(tweet)

# print(f"Word after tokenizer : {tweet\_tokens}")

tweets\_clean = []

for word in tweet\_tokens:

if (word not in stopwords\_english and word not in emoticons and word not in string.punctuation): # remove punctuation

# print(f"Word before stemming: {word}")

stem\_word = stemmer.stem(word) #stemming word

# print(f"Word after stem: {stem\_word}")

tweets\_clean.append(stem\_word)

return tweets\_clean

df['tweet\_clean'] = df['remove\_user'].apply(lambda x: tweet\_clean(x))

#remove punct

def remove\_punct(text):

text = " ".join([char for char in text if char not in string.punctuation])

return text

df['Tweet'] = df['tweet\_clean'].apply(lambda x: remove\_punct(x))

1. **Tuliskan/screenshotkan Stemming**

Stemming adalah proses mengubah kata menjadi bentuk dasarnya.

**Jawab:**

**→ Bagian kode yang melakukan stemming menggunakan Sastrawi:**

**stem\_word = stemmer.stem(word) #stemming word**

1. **Tuliskan/screenshotkan Simpan data text bersih**

**Jawab:**

df.sort\_values('Tweet', inplace = True)

df.drop(df.columns[[0,1]], axis = 1, inplace = True)

df.drop\_duplicates(subset = 'Tweet', keep = 'first', inplace = True)

df.to\_csv('ramadan\_clean\_tweet.csv', encoding='utf8', index=False)

df.head(10)

1. **Lampirkan file RAW hasil Crawling dan file bersih hasil preprocessing**