**TEXT PREPROCSSING STEPS**

1. **Text Cleaning**

**I.** **Converting to lowercase**

df = df.applymap(lambda x: x.lower() if isinstance(x, str) else x)

**II. Removing URLs**

import pandas as pd  
import re  
  
# Define a regex pattern to match URLs  
url\_pattern = re.compile(r'https?://\S+')  
  
# Define a function to remove URLs from text  
def remove\_urls(text):  
 return url\_pattern.sub('', text)  
  
# Apply the function to the 'text' column and create a new column 'clean\_text'  
df['Message'] = df['Message'].apply(remove\_urls)

**III. Removing remove non-word and non-whitespace characters**

df = df.replace(to\_replace=r'[^\w\s]', value='', regex=True)

**IV.** **Removing digits**

df = df.replace(to\_replace=r'\d', value='', regex=True)

## Tokenization

import nltk  
from nltk.tokenize import word\_tokenize  
  
df['Message'] = df['Message'].apply(word\_tokenize)

## Stopword Removal

## import nltk from nltk.corpus import stopwords stop\_words = set(stopwords.words('english')) df['Message'] = df['Message'].apply(lambda x: [word for word in x if word not in stop\_words])

## Stemming/Lemmatization

**I.** **Stemming**

import nltk  
from nltk.stem import PorterStemmer  
from nltk.tokenize import word\_tokenize  
import pandas as pd  
  
# Initialize the Porter Stemmer  
stemmer = PorterStemmer()  
  
# Define a function to perform stemming on the 'text' column  
def stem\_words(words):  
 return [stemmer.stem(word) for word in words]  
  
# Define a function to perform stemming on the 'text' column  
def stem\_words(words):  
 return [stemmer.stem(word) for word in words]  
  
# Apply the function to the 'text' column and create a new column 'stemmed\_text'  
df['stemmed\_messages'] = df['Message'].apply(stem\_words)

**II.** **Lemmatization**

import nltk  
nltk.download('averaged\_perceptron\_tagger')  
import nltk  
nltk.download('wordnet')  
  
import nltk  
from nltk.stem import WordNetLemmatizer  
from nltk.corpus import wordnet  
import pandas as pd  
  
# initialize lemmatizer  
lemmatizer = WordNetLemmatizer()  
  
# define function to lemmatize tokens  
def lemmatize\_tokens(tokens):  
 # convert POS tag to WordNet format  
 def get\_wordnet\_pos(word):  
 tag = nltk.pos\_tag([word])[0][1][0].upper()  
 tag\_dict = {"J": wordnet.ADJ,  
 "N": wordnet.NOUN,  
 "V": wordnet.VERB,  
 "R": wordnet.ADV}  
 return tag\_dict.get(tag, wordnet.NOUN)  
   
 # lemmatize tokens  
 lemmas = [lemmatizer.lemmatize(token, get\_wordnet\_pos(token)) for token in tokens]  
   
 # return lemmatized tokens as a list  
 return lemmas  
  
# apply lemmatization function to column of dataframe  
df['lemmatized\_messages'] = df['Message'].apply(lemmatize\_tokens)

## Reference

https://medium.com/@maleeshadesilva21/preprocessing-steps-for-natural-language-processing-nlp-a-beginners-guide-d6d9bf7689c9