import pandas as pd

import re

import nltk

from nltk.tokenize import TweetTokenizer

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

nltk.download('punkt')

# Initialize tools

tokenizer = TweetTokenizer()

analyzer = SentimentIntensityAnalyzer()

# 1. Preprocess Tweets

def preprocess\_tweet(tweet):

tweet = re.sub(r"http\S+|www\S+|https\S+", '', tweet) # Remove URLs

tweet = re.sub(r"\@\w+|\#", '', tweet) # Remove mentions and hashtags

tweet = re.sub(r"\s+", ' ', tweet).strip() # Remove extra whitespace

return tweet

# 2. Sentiment Analysis using VADER

def analyze\_sentiment(tweet):

score = analyzer.polarity\_scores(tweet)["compound"]

if score >= 0.05:

return "Positive"

elif score <= -0.05:

return "Negative"

else:

return "Neutral"

# 3. Emotion Analysis (placeholder function simulating pretrained RNN)

def analyze\_emotion(tweet):

# In real use, load a pretrained RNN model such as from HuggingFace

# Here we simulate with simple keyword matching for demo purposes

emotions = {

"joy": ["happy", "great", "excited", "love"],

"fear": ["scared", "afraid", "worried"],

"sadness": ["sad", "depressed", "unhappy"],

"anger": ["angry", "mad", "furious"],

"surprise": ["shocked", "surprised", "amazed"],

"disgust": ["disgusted", "gross", "nasty"]

}

tweet = tweet.lower()

for emotion, keywords in emotions.items():

if any(word in tweet for word in keywords):

return emotion.capitalize()

return "Neutral"

# 4. Apply pipeline to dataset

def analyze\_tweets(data\_path):

df = pd.read\_csv(data\_path)

df["cleaned\_tweet"] = df["tweet"].apply(preprocess\_tweet)

df["sentiment"] = df["cleaned\_tweet"].apply(analyze\_sentiment)

df["emotion"] = df["cleaned\_tweet"].apply(analyze\_emotion)

return df

# Example usage

# df\_results = analyze\_tweets("singapore\_covid\_tweets.csv")

# print(df\_results.head())