

Ukraine Russia War Twitter Sentiment Analysis using Python

NSUBUGA EMMANUEL REAGAN

Many countries are supporting Ukraine by introducing economic sanctions on Russia. There are a lot of tweets about the Ukraine and Russia war where people tend to update about the ground truths, what they feel about it, and who they are supporting. In this project, I analyze the sentiments of people over the Ukraine and Russian War.

The dataset that I am using for the task of Twitter sentiment analysis on the Ukraine and Russia War is downloaded from Kaggle. This dataset was initially collected from Twitter and is updated regularly.

I'll start by importing the necessary Python libraries and the dataset to get started with this task:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import nltk
import re
from nltk.corpus import stopwords
import string

data = pd.read_csv("filename.csv")
print(data.head())
```

	id	conversation_id	created_at
0	1630366235354451969	1630152070530576385	2023-02-28 00:36:15 UTC
1	1630366226424778753	1630366226424778753	2023-02-28 00:36:13 UTC
2	1630366225930027011	1630366225930027011	2023-02-28 00:36:13 UTC
3	1630366223056662530	1630351686974992385	2023-02-28 00:36:12 UTC
4	1630366221483884545	1629903982255644672	2023-02-28 00:36:12 UTC

	date	time	timezone	user_id	username
0	2023-02-28	00:36:15	0	1493761817406894086	tomasliptai
1	2023-02-28	00:36:13	0	1526694166662721536	paperfloure
2	2023-02-28	00:36:13	0	1053018392939167746	katetbar1
3	2023-02-28	00:36:12	0	602371247	jlhrdhmom
4	2023-02-28	00:36:12	0	1053594763214184448	phemikali

	name	place	geo	source	user_rt_id	user_rt	retweet_id
0	Tomas Liptai	NaN	NaN	NaN	NaN	NaN	NaN
1	Smell the roses	NaN	NaN	NaN	NaN	NaN	NaN
2	@etak	NaN	NaN	NaN	NaN	NaN	NaN
3	JLHrdh	NaN	NaN	NaN	NaN	NaN	NaN
4	rolarkcybersecurity	NaN	NaN	NaN	NaN	NaN	NaN

	reply_to	retweet_date	translate
0	['screen_name': 'nazijaeger__', 'name': 'nazi...]	NaN	NaN
1	NaN	NaN	NaN
2	NaN	NaN	NaN
3	['screen_name': 'Mainelifer', 'name': 'Mainel...]	NaN	NaN
4	['screen_name': 'Pottingpinks', 'name': 'GRS'...]	NaN	NaN

	trans_src	trans_dest
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN

[5 rows x 36 columns]

Let's have a quick look at all the column names of the dataset:

```
print(data.columns)

Index(['id', 'conversation_id', 'created_at', 'date', 'time', 'timezone',
       'user_id', 'username', 'name', 'place', 'tweet', 'language', 'mentions',
       'urls', 'photos', 'replies_count', 'retweets_count', 'likes_count',
       'hashtags', 'cashtags', 'link', 'retweet', 'quote_url', 'video',
       'thumbnail', 'near', 'geo', 'source', 'user_rt_id', 'user_rt',
       'retweet_id', 'reply_to', 'retweet_date', 'translate', 'trans_src',
       'trans_dest'],
      dtype='object')
```

We only need three columns for this task (username, tweet, and language); I will only select these columns and move forward:

```
data = data[["username", "tweet", "language"]]
```

Let's have a look at whether any of these columns contains any null values or not:

```
data.isnull().sum()

username    0
tweet       0
language    0
dtype: int64
```

So none of the columns has null values, let's have a quick look at how many tweets are posted in which language:

```
data["language"].value_counts()

en      8858
pt       440
it       194
qme      105
und        60
in        47
ru        44
ja        42
es        36
ca        20
qht        20
th         19
fr         18
de         14
ko          9
vi          8
nl          8
ro          7
fi          7
ar          6
zxx         6
uk          6
cs          6
zh          5
pl          5
qam         4
tl          4
da          3
eu          2
no          2
hi          2
```

```

tr      2
hu      1
cy      1
lv      1
el      1
bn      1
Name: language, dtype: int64

```

So most of the tweets are in English. Let's prepare this data for the task of sentiment analysis. Here I will remove all the links, punctuation, symbols and other language errors from the tweets:

```

nltk.download('stopwords')
stemmer = nltk.SnowballStemmer("english")
stopword=set(stopwords.words('english'))

def clean(text):
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text
data["tweet"] = data["tweet"].apply(clean)

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

```

Now let's have a look at the wordcloud of the tweets, which will show the most frequently used words in the tweets by people sharing their feelings and updates about the Ukraine and Russia war:

```

text = " ".join(i for i in data.tweet)
stopwords = set(STOPWORDS)
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()

```



Now I will add three more columns in this dataset as Positive, Negative, and Neutral by calculating the sentiment scores of the tweets:



```

nltk.download('vader_lexicon')
sentiments = SentimentIntensityAnalyzer()
data["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in data["tweet"]]
data["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in data["tweet"]]
data["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in data["tweet"]]
data = data[["tweet", "Positive", "Negative", "Neutral"]]
print(data.head())

```

```

[nltk_data] Downloading package vader_lexicon to /root/nltk_data...

```

	tweet	Positive	Negative	Neutral
0	nazijaeg derwen russia place satan rule well	0.259	0.000	0.741
1	russia haarp could destroy usa one fell swoop ...	0.000	0.280	0.720
2	putin give steven seagal order friendship	0.367	0.000	0.633
3	mainelif baddcomani it alway project russia	0.000	0.000	1.000
4	pottingpink mfarussia modrussia milhistrf muze...	0.068	0.078	0.854

Now let's have a look at the most frequent words used by people with positive sentiments:

```

positive = ' '.join([i for i in data['tweet'][data['Positive'] > data["Negative"]]])
stopwords = set(STOPWORDS)
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(positive)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()

```

theory **119** **120** **121** **122** **123** **124** **125** **126** **127** **128** **129** **130** **131** **132** **133** **134** **135** **136** **137** **138** **139** **140** **141** **142** **143** **144** **145** **146** **147** **148** **149** **150** **151** **152** **153** **154** **155** **156** **157** **158** **159** **160** **161** **162** **163** **164** **165** **166** **167** **168** **169** **170** **171** **172** **173** **174** **175** **176** **177** **178** **179** **180** **181** **182** **183** **184** **185** **186** **187** **188** **189** **190** **191** **192** **193** **194** **195** **196** **197** **198** **199** **200** **201** **202** **203** **204** **205** **206** **207** **208** **209** **210** **211** **212** **213** **214** **215** **216** **217** **218** **219** **220** **221** **222** **223** **224** **225** **226** **227** **228** **229** **230** **231** **232** **233** **234** **235** **236** **237** **238** **239** **240** **241** **242** **243** **244** **245** **246** **247** **248** **249** **250** **251** **252** **253** **254** **255** **256** **257** **258** **259** **260** **261** **262** **263** **264** **265** **266** **267** **268** **269** **270** **271** **272** **273** **274** **275** **276** **277** **278** **279** **280** **281** **282** **283** **284** **285** **286** **287** **288** **289** **290** **291** **292** **293** **294** **295** **296** **297** **298** **299** **300** **301** **302** **303** **304** **305** **306** **307** **308** **309** **310** **311** **312** **313** **314** **315** **316** **317** **318** **319** **320** **321** **322** **323** **324** **325** **326** **327** **328** **329** **330** **331** **332** **333** **334** **335** **336** **337** **338** **339** **340** **341** **342** **343** **344** **345** **346** **347** **348** **349** **350** **351** **352** **353** **354** **355** **356** **357** **358** **359** **360** **361** **362** **363** **364** **365** **366** **367** **368** **369** **370** **371** **372** **373** **374** **375** **376** **377** **378** **379** **380** **381** **382** **383** **384** **385** **386** **387** **388** **389** **390** **391** **392** **393** **394** **395** **396** **397** **398** **399** **400** **401** **402** **403** **404** **405** **406** **407** **408** **409** **410** **411** **412** **413** **414** **415** **416** **417** **418** **419** **420** **421** **422** **423** **424** **425** **426** **427** **428** **429** **430** **431** **432** **433** **434** **435** **436** **437** **438** **439** **440** **441** **442** **443** **444** **445** **446** **447** **448** **449** **450** **451** **452** **453** **454** **455** **456** **457** **458** **459** **460** **461** **462** **463** **464** **465** **466** **467** **468** **469** **470** **471** **472** **473** **474** **475** **476** **477** **478** **479** **480** **481** **482** **483** **484** **485** **486** **487** **488** **489** **490** **491** **492** **493** **494** **495** **496** **497** **498** **499** **500** **501** **502** **503** **504** **505** **506** **507** **508** **509** **510** **511** **512** **513** **514** **515** **516** **517** **518** **519** **520** **521** **522** **523** **524** **525** **526** **527** **528** **529** **530** **531** **532** **533** **534** **535** **536** **537** **538** **539** **540** **541** **542** **543** **544** **545** **546** **547** **548** **549** **550** **551** **552** **553** **554** **555** **556** **557** **558** **559** **560** **561** **562** **563** **564** **565** **566** **567** **568** **569** **570** **571** **572** **573</**