

# Praca domowa 3

## Temat Analiza opinii o energii nuklearnej

Grzegorz Grygorowicz

Już od jakiegoś czasu są rozmowy na temat energii nuklearnej, zbudza mieszane uczucia, nie który sądzą, że jest nie dobra dla otoczenia a nie którzy wręcz przeciwnie. Jakie emocje wywołała decyzja Niemiec, czy podejście do tego tematu Francji. Celem tej pracy jest analiza tweetów używających hashtagu "#nuclearenergy"

### Pobranie tweetów za pomocą snscreape.

Zakres dat jest od grudnia 2022 do kwietnia 2023 (Niemcy zamykają elektrownie atomowe). Pobranie ok 75 tweetów na dzień, o ile tle było danego dnia.

Tweety z danego miesiąca są zapisywane w pliku csv o nazwie nuclearenergy-Tweets-[miesiąc]-[rok]

Łącznie jest 11325 tweetów.

```
In [ ]: import snscreape.modules.twitter as sntwitter
import pandas as pd
from datetime import date
from datetime import timedelta
import os

def GrabTweets(month,year,num_tweets_per_day,query):
    if month in [1, 3, 5, 7, 8, 10, 12]:
        days_in_month = 31
        total_tweets = days_in_month * num_tweets_per_day

    elif month in [4, 6, 9, 11]:
        days_in_month = 30
        total_tweets = days_in_month * num_tweets_per_day

    elif month == 2:
        days_in_month = 28
        total_tweets = days_in_month * num_tweets_per_day

    tweets_list = []

    until = date(year, month, 2).isoformat()
    since = date(year, month, 1).isoformat()

    counter = 0

    while counter != total_tweets:
        for i,tweet in enumerate(sntwitter.TwitterHashtagScraper('{}'.format(query) + ' since:{} ur'.format(since), until=until).get_items()):
            if i >= num_tweets_per_day:
                until = (date.fromisoformat(until) + timedelta(days=1)).isoformat()
                since = (date.fromisoformat(since) + timedelta(days=1)).isoformat()
                break
```

```

        break
    tweets_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username])
    counter += 1

DF = pd.DataFrame(tweets_list, columns=['Date', 'Tweet ID', 'Text', 'Username'])
os.chdir('D:/studia/IO/aiart_opinion/cv')

DF.to_csv('{}-Tweets-{}-{}.csv'.format(query,month,year), encoding='utf-8',
          print("Done!")

```

In [ ]:

```

query = "nuclearenergy"
year1 = 2022
year2 = 2023
num_tweets_per_day = 75

```

```

GrabTweets(12,year1,num_tweets_per_day,query)
GrabTweets(1,year2,num_tweets_per_day,query)
GrabTweets(2,year2,num_tweets_per_day,query)
GrabTweets(3,year2,num_tweets_per_day,query)
GrabTweets(4,year2,num_tweets_per_day,query)

```

C:\Users\Shiryon\AppData\Local\Temp\ipykernel\_6628\3268144449.py:34: DeprecationWarning: content is deprecated, use rawContent instead  
 tweets\_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username, tweet.replyCount, tweet.retweetCount, tweet.likeCount, tweet.quoteCount])  
 Unavailable user in card on tweet 1601343181789937664  
 User 14204245 not found in user refs in card on tweet 1601343181789937664  
 Unavailable user in card on tweet 1601332291313324033  
 User 14204245 not found in user refs in card on tweet 1601332291313324033  
 Unavailable user in card on tweet 1601654509864091648  
 User 14204245 not found in user refs in card on tweet 1601654509864091648  
 Done!

C:\Users\Shiryon\AppData\Local\Temp\ipykernel\_6628\3268144449.py:34: DeprecationWarning: content is deprecated, use rawContent instead  
 tweets\_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username, tweet.replyCount, tweet.retweetCount, tweet.likeCount, tweet.quoteCount])  
 Twitter responded with an error: TimeoutError: Timeout: Unspecified  
 Empty user ref object in card on tweet 1614144768292061184  
 Empty user ref object in card on tweet 1614144768292061184  
 User 4503599629184530 not found in user refs in card on tweet 1614144768292061184  
 User 4503599629184530 not found in user refs in card on tweet 1614144768292061184  
 Unavailable user in card on tweet 1620565715945422848  
 Unavailable user in card on tweet 1620565715945422848  
 User 1203690121373069312 not found in user refs in card on tweet 1620565715945422848  
 User 1203690121373069312 not found in user refs in card on tweet 1620565715945422848  
 Done!

```
C:\Users\Shiryon\AppData\Local\Temp\ipykernel_6628\3268144449.py:34: DeprecationWarning: content is deprecated, use rawContent instead
    tweets_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username, t
weet.replyCount, tweet.retweetCount, tweet.likeCount, tweet.quoteCount])
Unavailable user in card on tweet 1622370809602867201
Unavailable user in card on tweet 1622370809602867201
User 1203690121373069312 not found in user refs in card on tweet 1622370809602867
201
User 1203690121373069312 not found in user refs in card on tweet 1622370809602867
201
Unavailable user in card on tweet 1622370802694852613
Unavailable user in card on tweet 1622370802694852613
User 1203690121373069312 not found in user refs in card on tweet 1622370802694852
613
User 1203690121373069312 not found in user refs in card on tweet 1622370802694852
613
Unavailable user in card on tweet 1625235924186460180
User 14204245 not found in user refs in card on tweet 1625235924186460180
Could not translate t.co card URL on tweet 1625536317948928000
Could not translate t.co card URL on tweet 1627031729230819329
Done!
```

```
C:\Users\Shiryon\AppData\Local\Temp\ipykernel_6628\3268144449.py:34: DeprecationWarning: content is deprecated, use rawContent instead
    tweets_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username, t
weet.replyCount, tweet.retweetCount, tweet.likeCount, tweet.quoteCount])
Empty user ref object in card on tweet 1636873254995656707
Empty user ref object in card on tweet 1636873254995656707
User 4503599629184530 not found in user refs in card on tweet 1636873254995656707
User 4503599629184530 not found in user refs in card on tweet 1636873254995656707
Unavailable user in card on tweet 1636851192788844545
User 27073265 not found in user refs in card on tweet 1636851192788844545
Could not translate t.co card URL on tweet 1636826617719980033
Unavailable user in card on tweet 1638285652398579713
User 27073265 not found in user refs in card on tweet 1638285652398579713
Could not translate t.co card URL on tweet 1638651858318139393
Unavailable user in card on tweet 1640104469420552196
Unavailable user in card on tweet 1640104469420552196
User 15843059 not found in user refs in card on tweet 1640104469420552196
User 15843059 not found in user refs in card on tweet 1640104469420552196
Done!
```

```
C:\Users\Shiryon\AppData\Local\Temp\ipykernel_6628\3268144449.py:34: DeprecationWarning: content is deprecated, use rawContent instead
  tweets_list.append([tweet.date, tweet.id, tweet.content, tweet.user.username, t
weet.replyCount, tweet.retweetCount, tweet.likeCount, tweet.quoteCount])
Could not translate t.co card URL on tweet 1646616308039950337
Unavailable user in card on tweet 1647012745622044672
User 14204245 not found in user refs in card on tweet 1647012745622044672
Could not translate t.co card URL on tweet 1648086227084627970
Could not translate t.co card URL on tweet 1649915966652837889
Unavailable user in card on tweet 1650282775759708161
Unavailable user in card on tweet 1650282775759708161
User 1434052940 not found in user refs in card on tweet 1650282775759708161
User 1434052940 not found in user refs in card on tweet 1650282775759708161
Unavailable user in card on tweet 1650191546560688129
Unavailable user in card on tweet 1650191546560688129
User 1434052940 not found in user refs in card on tweet 1650191546560688129
User 1434052940 not found in user refs in card on tweet 1650191546560688129
Unavailable user in card on tweet 1652049108968808475
Unavailable user in card on tweet 1652049108968808475
User 1203690121373069312 not found in user refs in card on tweet 1652049108968808
475
User 1203690121373069312 not found in user refs in card on tweet 1652049108968808
475
Unavailable user in card on tweet 1652399858550992897
Unavailable user in card on tweet 1652399858550992897
User 1203690121373069312 not found in user refs in card on tweet 1652399858550992
897
User 1203690121373069312 not found in user refs in card on tweet 1652399858550992
897
```

Done!

## Wczytanie tweetów - z plików.

```
In [ ]: import pandas as pd

a1 = pd.read_csv('nuclearenergy-Tweets-1-2023.csv')
a2 = pd.read_csv('nuclearenergy-Tweets-2-2023.csv')
a3 = pd.read_csv('nuclearenergy-Tweets-3-2023.csv')
a4 = pd.read_csv('nuclearenergy-Tweets-4-2023.csv')
a12 = pd.read_csv('nuclearenergy-Tweets-12-2022.csv')

all_tweets = a1['Text'].tolist() + a2['Text'].tolist() + a3['Text'].tolist() + a
dece_tweets = a1['Text'].tolist()
```

## Analiza całościowa

### Tokenizacja tweetów przy użyciu nltk

```
In [ ]: import nltk

tokenized_tweets = []
for tweet in all_tweets:
    tokenized_tweet = nltk.word_tokenize(tweet.lower())
    tokenized_tweets.append(tokenized_tweet)
```

## Filtrowanie tweetów

Używają stopwords angielski domyślnych i dodanie swoich by usunąć znaki specjalne albo nie potrzebne słowa takie jak https czy 'm albo 're. albo hashtags

```
In [ ]: from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
stop_words.add('#')
stop_words.add('~')
stop_words.add('-')
stop_words.add(':')
stop_words.add('https')
stop_words.add('//t.co/ZTrU7v20Ef')
stop_words.add('?')
stop_words.add('24/7')
stop_words.add('@')
stop_words.add(',')
stop_words.add('.')
stop_words.add('!')
stop_words.add(';')
stop_words.add('$')
stop_words.add('&')
stop_words.add("'s")
stop_words.add(')')
stop_words.add('(')
stop_words.add("''")
stop_words.add('``')
stop_words.add('\'')
stop_words.add("'''")
stop_words.add('``')
stop_words.add("''")
stop_words.add('*')
stop_words.add('...')
stop_words.add('u')
stop_words.add('l')
stop_words.add('..')
stop_words.add('\'')
stop_words.add('\"')
stop_words.add('%')
stop_words.add("'re")
stop_words.add("nuclearenergy")
```

```
filtered_tweets = []
filtered_tweets = [[word for word in tweet if word not in stop_words] for tweet
```

## Lemanizacja tweetów

```
In [ ]: from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

lemmatized_tweets = [[lemmatizer.lemmatize(word) for word in tweet] for tweet in
```

## Najczęstsze słowa w tweetach

```
In [ ]: from collections import Counter
import matplotlib.pyplot as plt
```

```

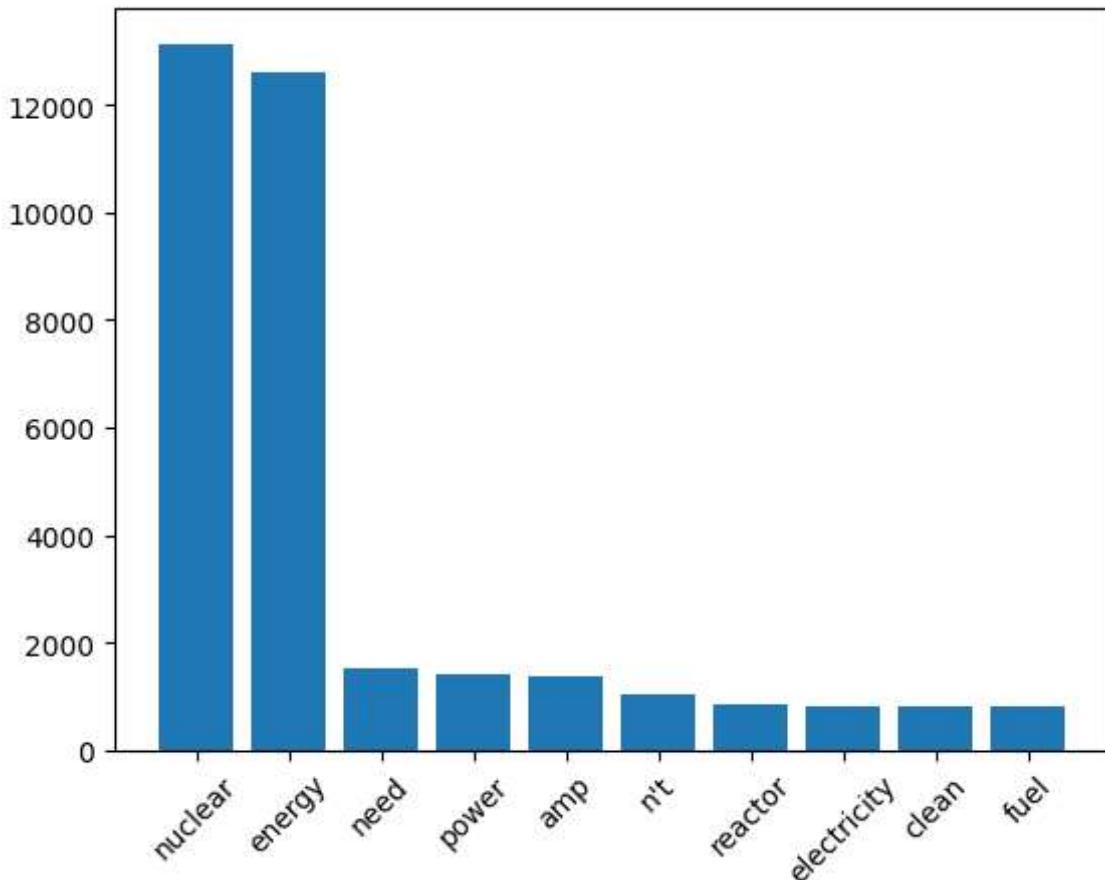
flattened_words = [word for tweet in lemmatized_tweets for word in tweet]

word_count = Counter(flattened_words)

most_common = word_count.most_common(10)
print(most_common)
labels, values = zip(*most_common)
plt.bar(labels, values)
plt.xticks(rotation=45)
plt.show()

```

[('nuclear', 13139), ('energy', 12603), ('need', 1532), ('power', 1392), ('amp', 1379), ("n't", 1015), ('reactor', 836), ('electricity', 822), ('clean', 819), ('fuel', 816)]



Najwięcej użyto słów takich jak nuclear, energy, power, need, amp, not, fuel, clean, reactor i climate. Co ciekawe słowa nuclear i energy pojawiają się więcej razy niż ilość tweetów nuclear o 1814 razy, energo 1278 razy. Kolejne słowa pojawiają się znacznie mniej.

### Analiza emocji za pomocą narzędzia nltk vader dla całości

```

In [ ]: from nltk.sentiment.vader import SentimentIntensityAnalyzer

sid = SentimentIntensityAnalyzer()

positive = []
negative = []
neutral = []

for tweet in all_tweets:

```

```

score = sid.polarity_scores(tweet)
if score['compound'] > 0:
    positive.append(tweet)
elif score["compound"] < 0:
    negative.append(tweet)
else:
    neutral.append(tweet)

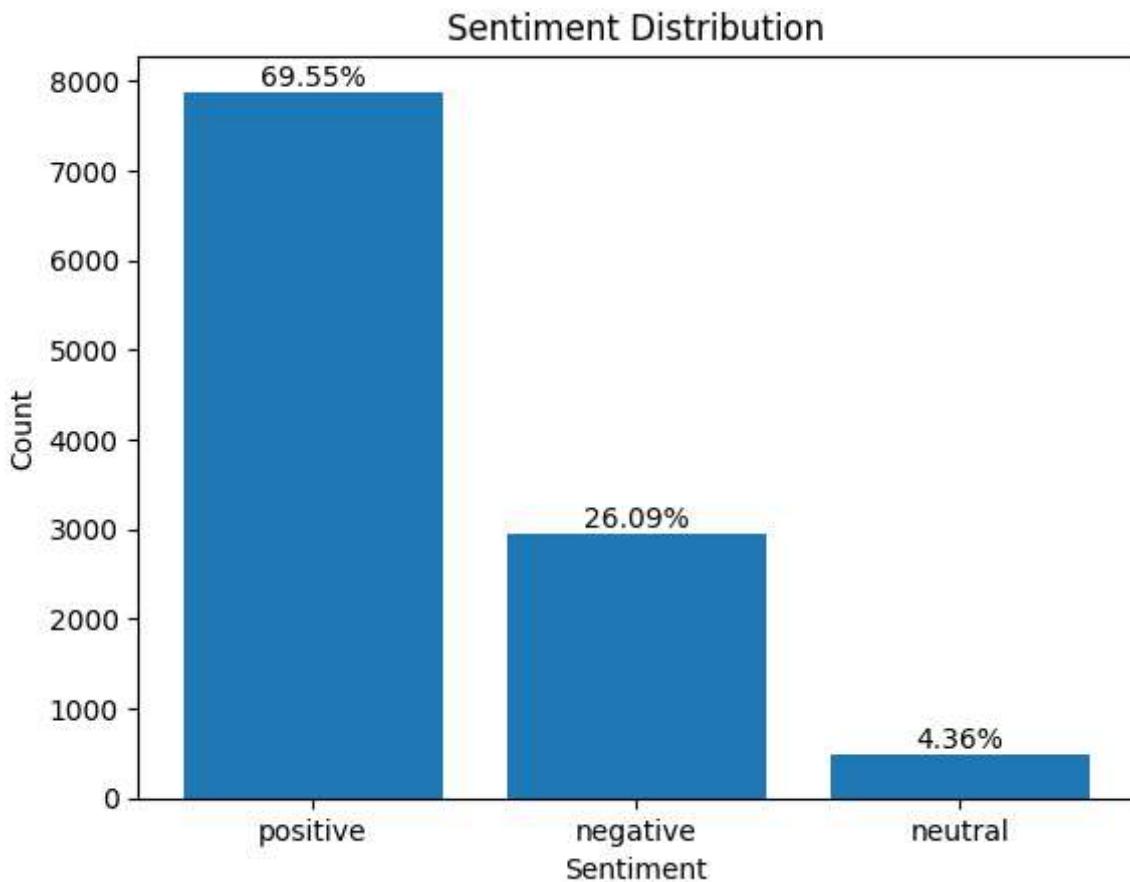
numbers = [len(positive), len(negative), len(neutral)]
labels = ['positive', 'negative', 'neutral']

plt.bar(labels, numbers)
for i, num in enumerate(numbers):
    percentage = str(round(num/len(all_tweets) * 100, 2)) + "%"
    plt.text(i, num, percentage, ha='center', va='bottom')

plt.title('Sentiment Distribution')
plt.xlabel('Sentiment')
plt.ylabel('Count')

plt.show()

```



Jak można zauważyć przy sprawdzeniu każdego tweeta, średnie emocji wychodzą, że ok 70% jest pozytywnie nastawiona, ok 26% negatywnie i ok 4% neutralnie. Jest to dosyć zaskakujące gdyż mogło się wydawać, że negatywnie nastawionych jest trochę więcej

### Analiza emocji za pomocą narzędzia text2emotion dla całości

In [ ]: `import text2emotion as te`

```
emotion_results = { "Happy": 0, "Angry": 0, "Surprise": 0, "Sad": 0, "Fear": 0 }
```

```

emotion_tweets = []
i = 0
for tweet in all_tweets:
    print(i)
    tweet_emotions = te.get_emotion(tweet)
    dominating_emotion = max(tweet_emotions, key=tweet_emotions.get)
    emotion_results[dominating_emotion] += 1
    i += 1

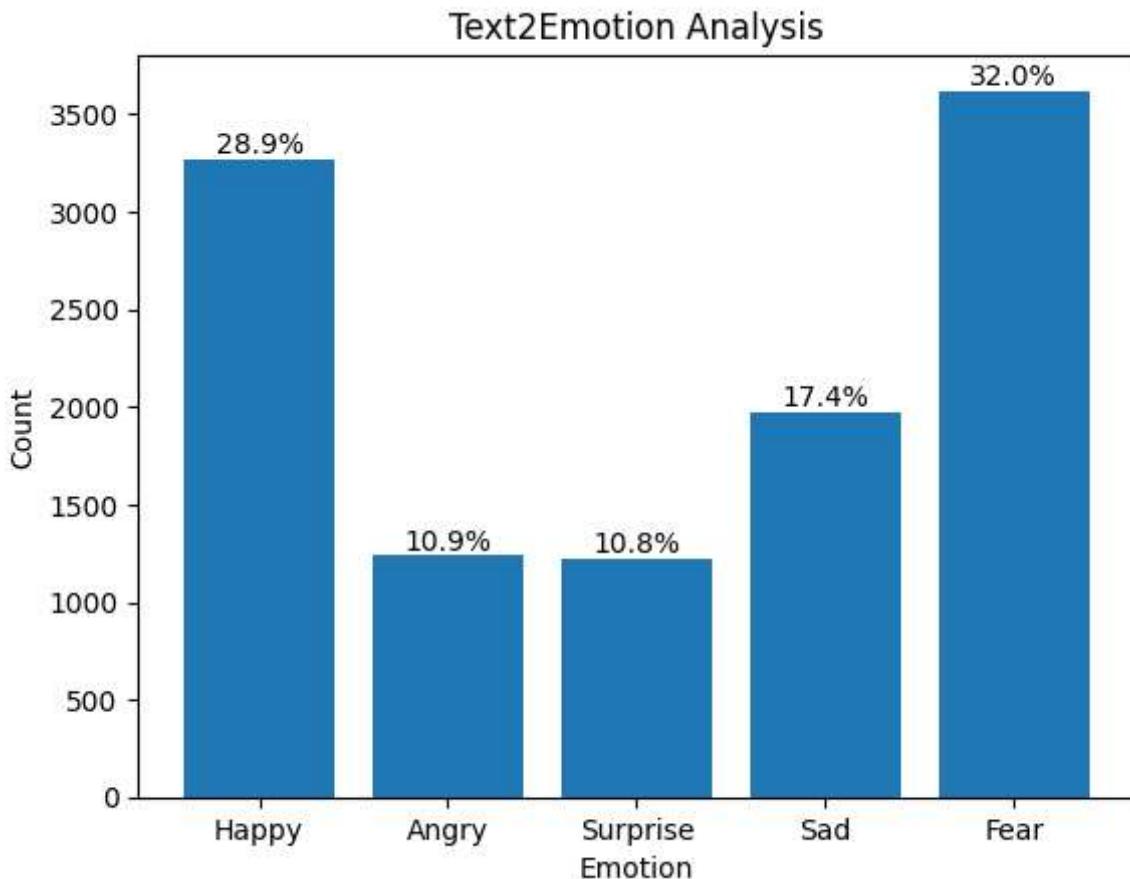
numbers = [emotion_results["Happy"], emotion_results["Angry"], emotion_results["Surprise"], emotion_results["Sad"], emotion_results["Fear"]]
labels = ['Happy', 'Angry', 'Surprise', 'Sad', 'Fear']

plt.bar(labels, numbers)
for i, num in enumerate(numbers):
    percentage = str(round(num/len(all_tweets) * 100, 1)) + "%"
    plt.text(i, num, percentage, ha='center', va='bottom')

plt.title('Text2Emotion Analysis')
plt.xlabel('Emotion')
plt.ylabel('Count')
plt.show()

```

[nltk\_data] Downloading package stopwords to  
[nltk\_data] C:\Users\Shiryon\AppData\Roaming\nltk\_data...  
[nltk\_data] Package stopwords is already up-to-date!  
[nltk\_data] Downloading package punkt to  
[nltk\_data] C:\Users\Shiryon\AppData\Roaming\nltk\_data...  
[nltk\_data] Package punkt is already up-to-date!  
[nltk\_data] Downloading package wordnet to  
[nltk\_data] C:\Users\Shiryon\AppData\Roaming\nltk\_data...  
[nltk\_data] Package wordnet is already up-to-date!



Średnie emocje przy użyciu text2emotion dają ok 29% szczęśliwa, ok 11% zła, ok 11% jest również zaskoczona, ok 17% jest smutna i ok 32% przestraszona.

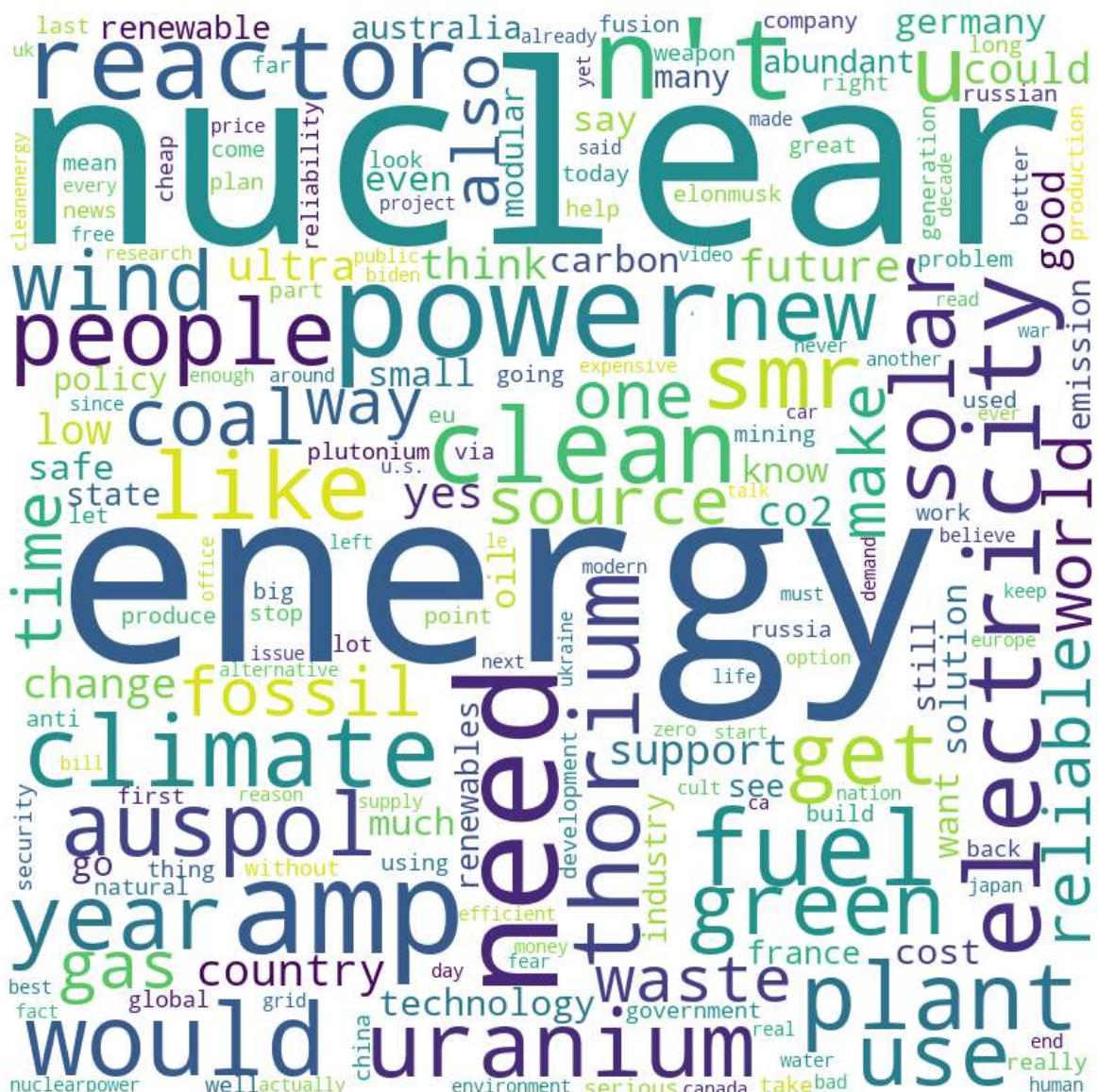
Porównanie dla całości emocji danych przez nltk vader i text2emotion.

Nltk vader daje, że większość jest pozytywna, w mniejszości jest negatywna i znacznej większości jest neutralna. W text2emotion najczęściej jest przestraszonych co jest raczej negatywną emocją i na drugim miejscu jest szczęście - pozytywna emocja, później smutek - negatywna emocja, i potem po równo zła (negatywna) i zaskoczona co może być i negatywną i pozytywną emocią. Jak można wyraźnie zauważycie nie pokrywają się zbytnio

## Chmura słów

```
In [ ]: from wordcloud import WordCloud
```

```
wordcloud = WordCloud(width=800, height=800, background_color='white').generate_
plt.figure(figsize=(8,8), facecolor=None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```



Słowa tutaj występujące są raczej zgodnie z oczekiwaniami takie jak energy, nuclear, plant, electricity, need, clean reactor, powet itp. chodziaż mogło się by wydawać że Niemcy pojawią się wiecej razy gdyż zamkneli swoje elektrownie i nie oczekiwanym słowem pojawiającym się jest Biden.

## Czasowa analiza emocji

Podział na miesiące od grudnia 2022 roku do kwietnia 2023 roku

### Przygotowanie danych

```
In [ ]: g = a12['Text'].tolist()
s = a1['Text'].tolist()
l = a2['Text'].tolist()
m = a3['Text'].tolist()
k = a4['Text'].tolist()
```

### Analiza emocji przy pomocy narzędzia nltk vader.

```
In [ ]: def vaderEmotion(tw, month):
    sid = SentimentIntensityAnalyzer()

    positive = []
    negative = []
    neutral = []

    for tweet in tw:
        score = sid.polarity_scores(tweet)
        if score['compound'] > 0:
            positive.append(tweet)
        elif score["compound"] < 0:
            negative.append(tweet)
        else:
            neutral.append(tweet)

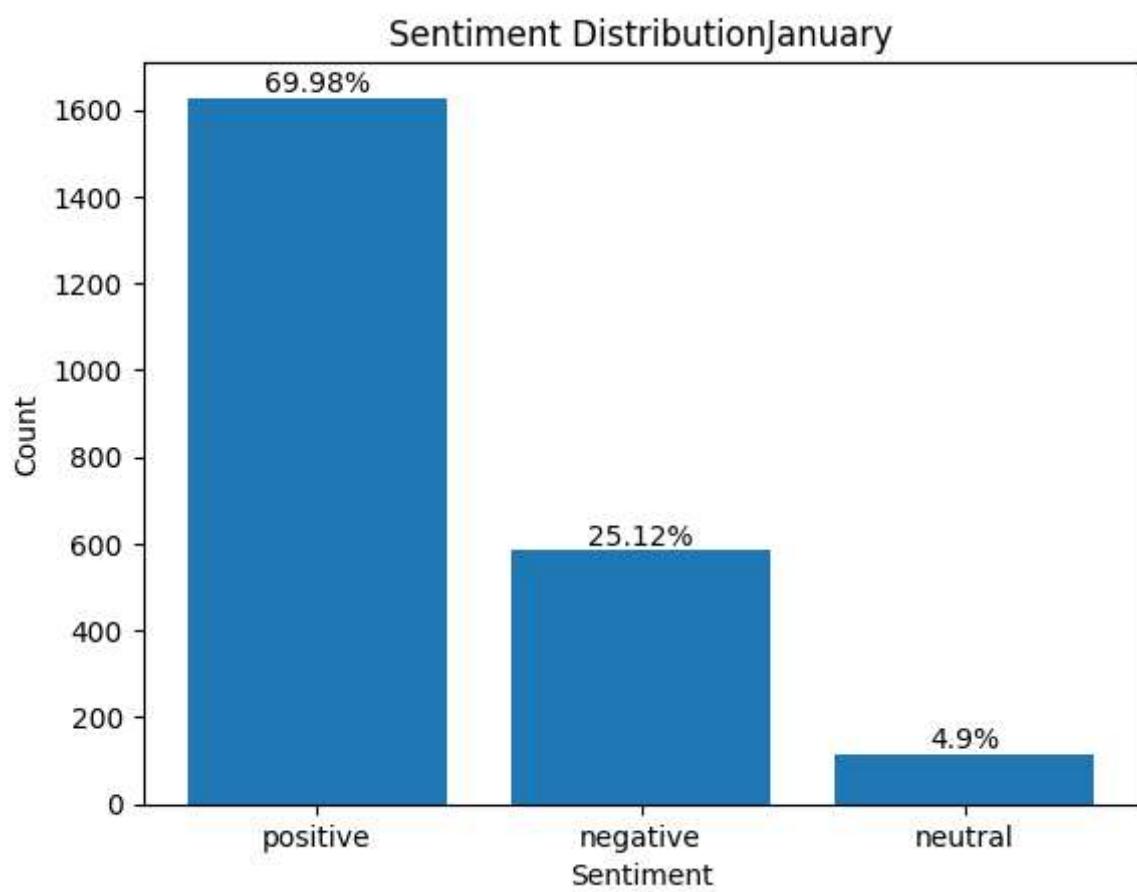
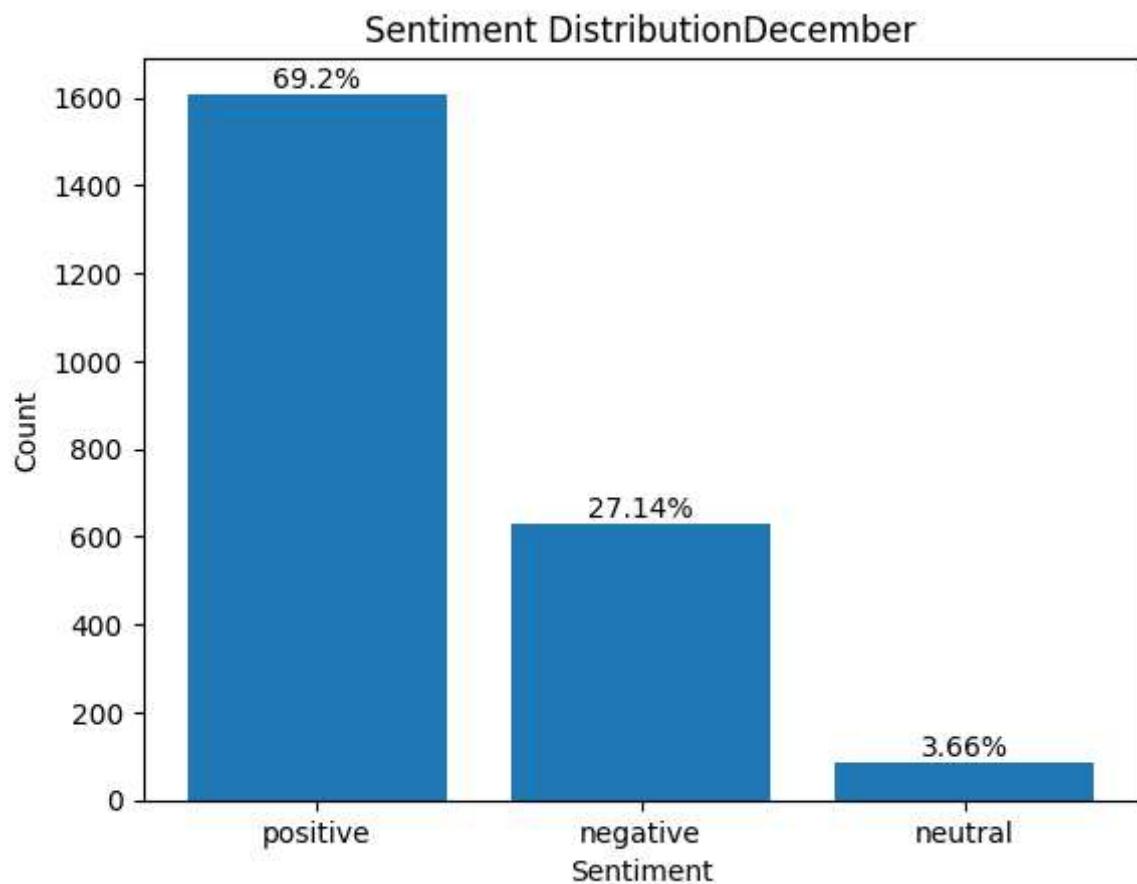
    numbers = [len(positive), len(negative), len(neutral)]
    labels = ['positive', 'negative', 'neutral']

    plt.bar(labels, numbers)
    for i, num in enumerate(numbers):
        percentage = str(round(num/len(tw) * 100, 2)) + "%"
        plt.text(i, num, percentage, ha='center', va='bottom')

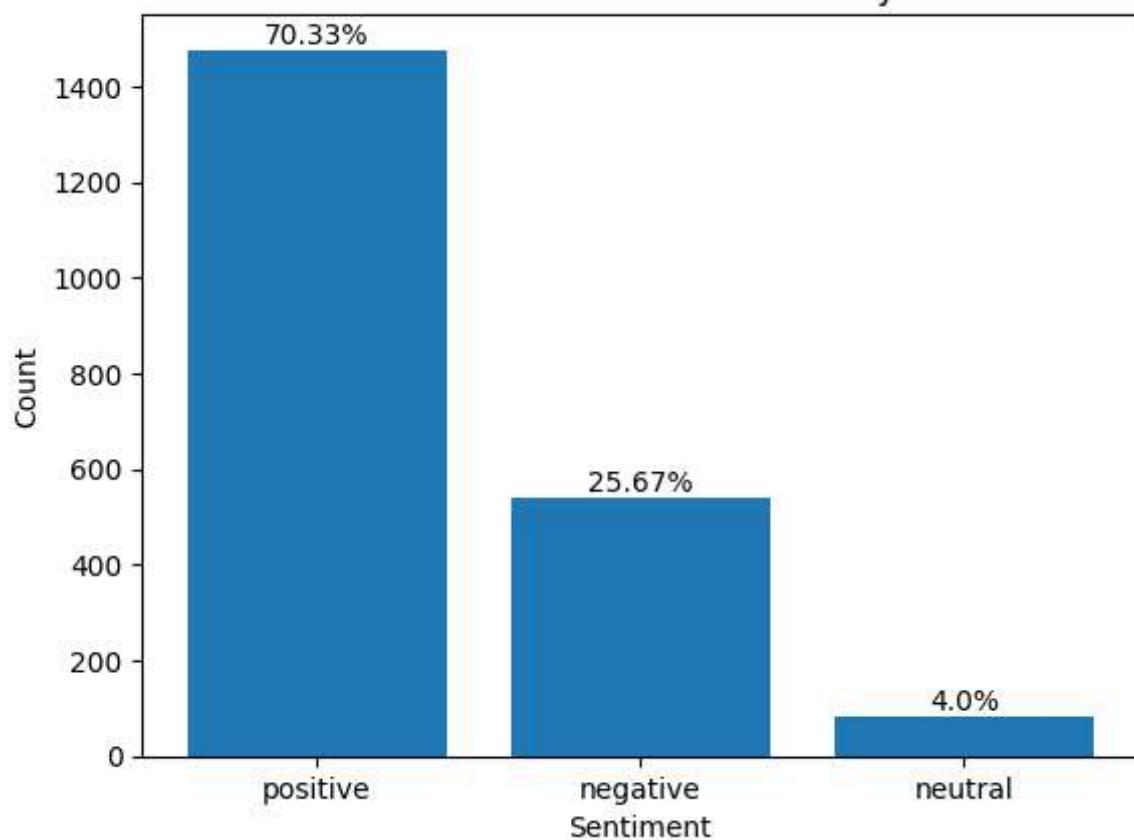
    plt.title('Sentiment Distribution' + month)
    plt.xlabel('Sentiment')
    plt.ylabel('Count')

    plt.show()

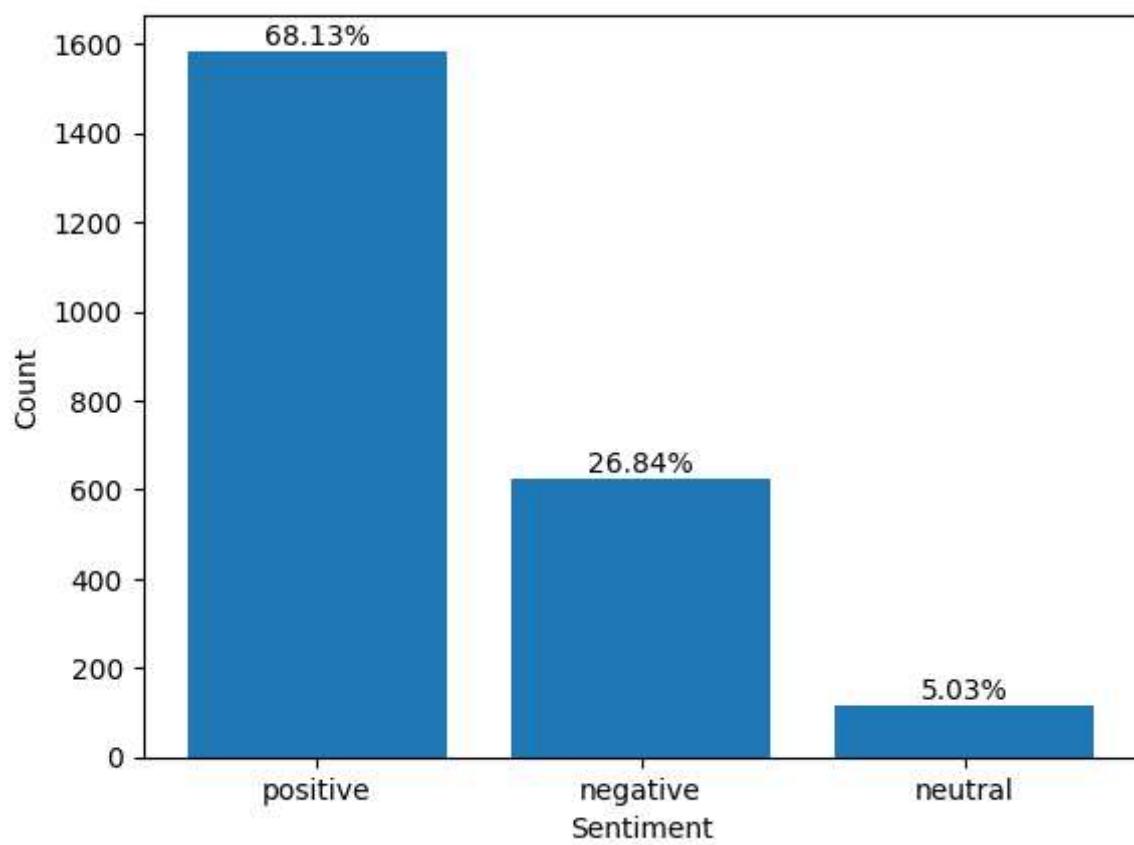
vaderEmotion(g, "December")
vaderEmotion(s, "January")
vaderEmotion(l, "February")
vaderEmotion(m, "March")
vaderEmotion(k, "April")
```

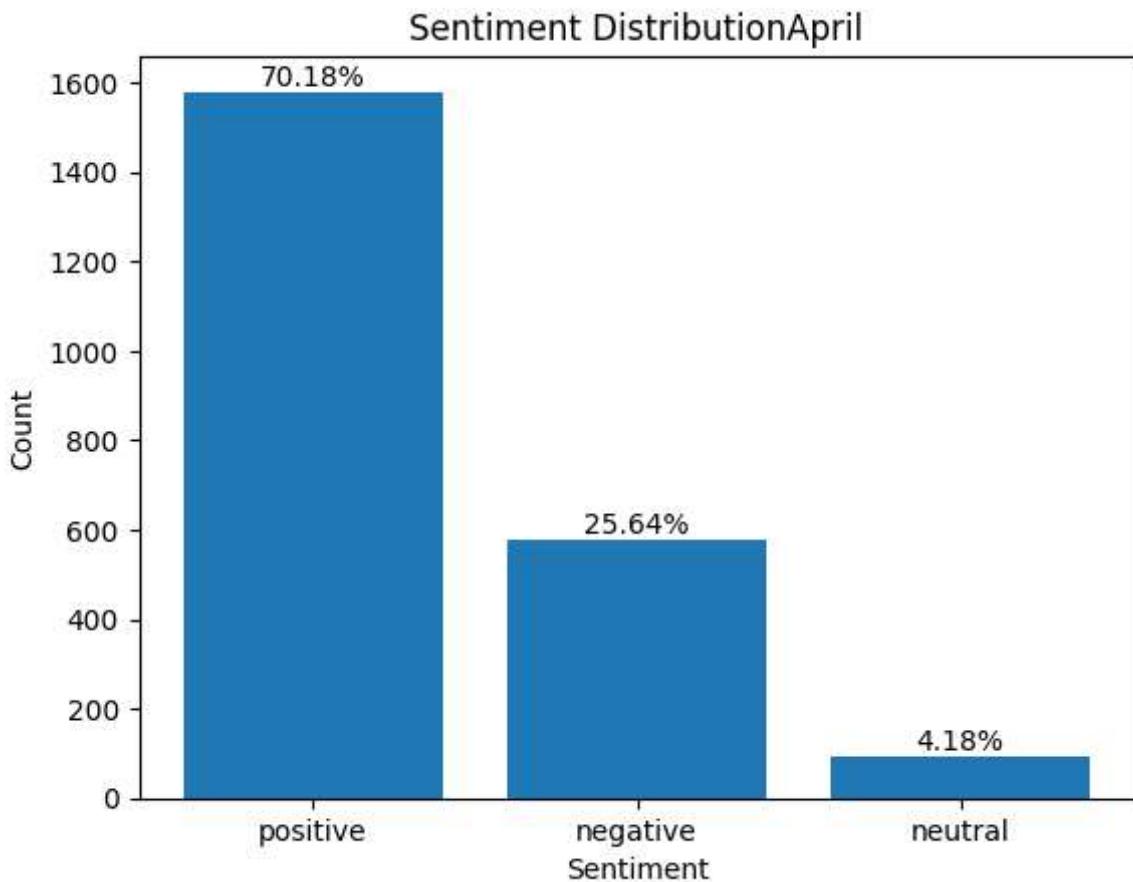


### Sentiment DistributionFebruary



### Sentiment DistributionMarch





Pozytywne, negatywne i neutralne emocje mniej więcej się utrzymują mogą się różnić o ok 2% w mniej lub więcej tych emocji.

### Analiza emocji przy pomocy narzędzia text2emotion

```
In [ ]: def t2e(allt, month):
    emotion_results = { "Happy": 0, "Angry": 0, "Surprise": 0, "Sad": 0, "Fear": 0}
    emotion_tweets = []
    for tweet in allt:
        tweet_emotions = te.get_emotion(tweet)
        dominating_emotion = max(tweet_emotions, key=tweet_emotions.get)
        emotion_results[dominating_emotion] += 1

    numbers = [emotion_results["Happy"], emotion_results["Angry"], emotion_results["Surprise"], emotion_results["Sad"], emotion_results["Fear"]]
    labels = ['Happy', 'Angry', 'Surprise', 'Sad', 'Fear']

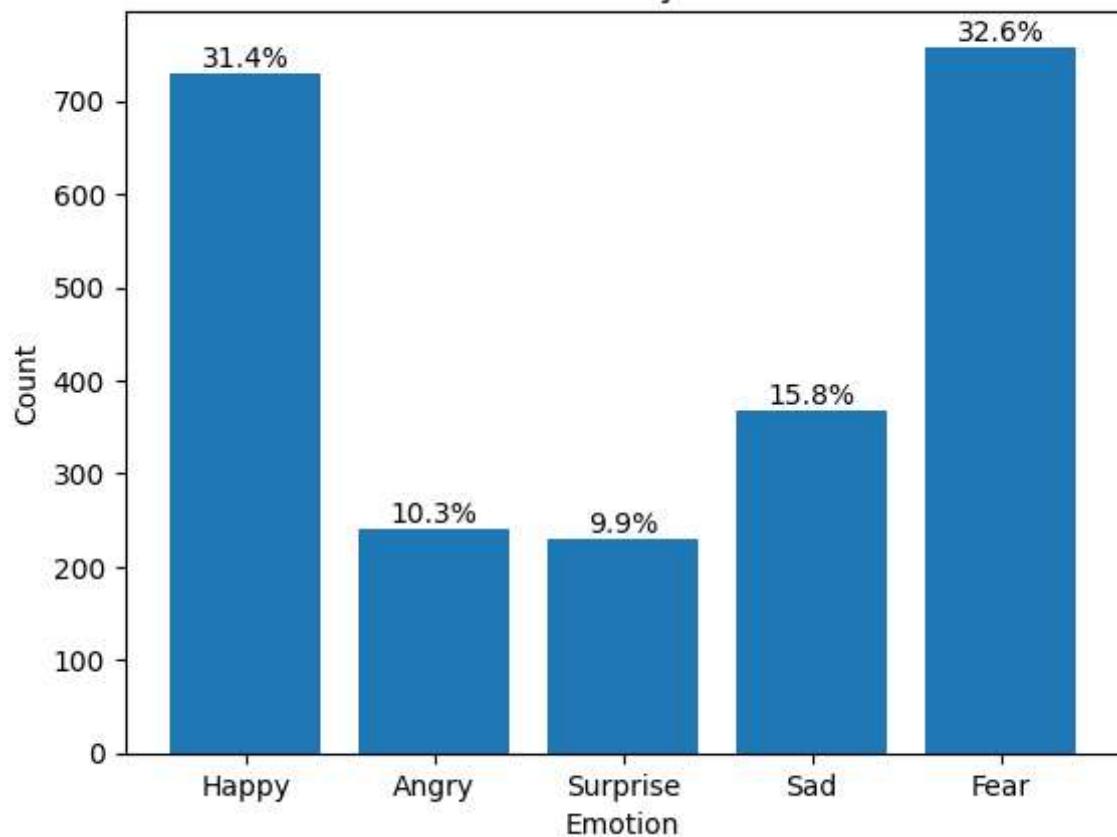
    plt.bar(labels, numbers)
    for i, num in enumerate(numbers):
        percentage = str(round(num/len(allt) * 100, 1)) + "%"
        plt.text(i, num, percentage, ha='center', va='bottom')

    plt.title('Text2Emotion Analysis' + month)
    plt.xlabel('Emotion')
    plt.ylabel('Count')
    plt.show()

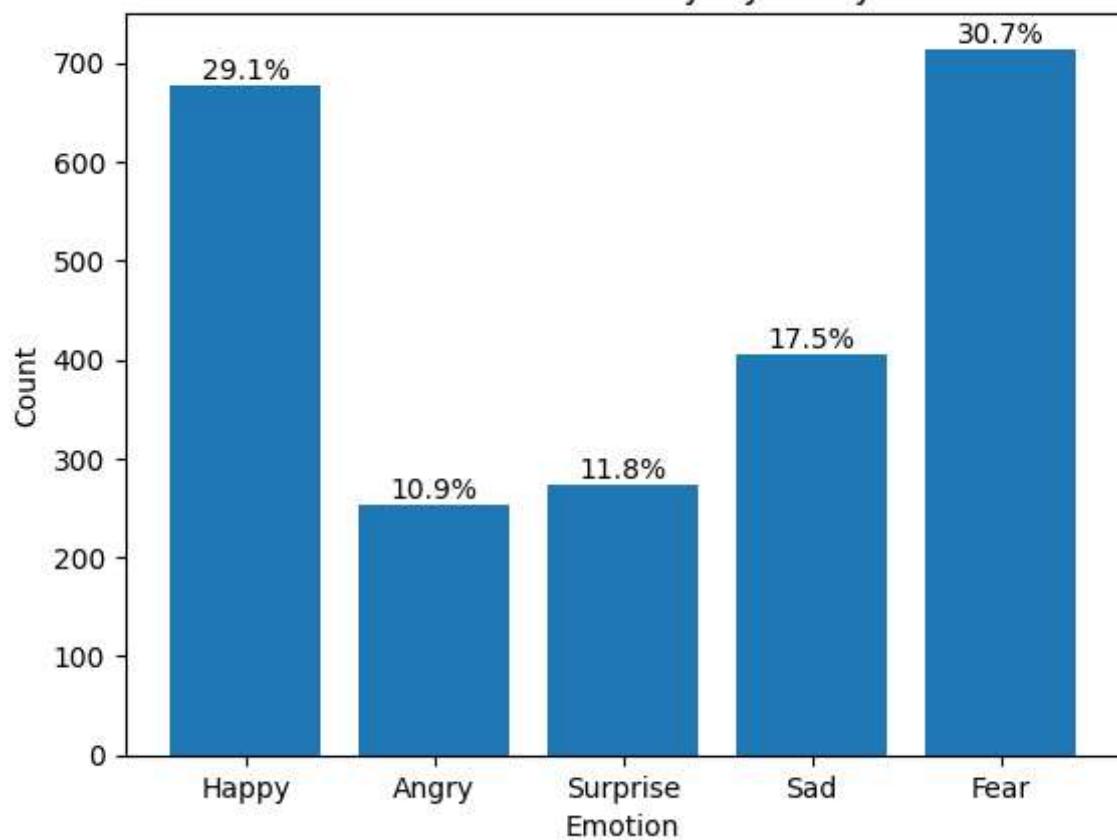
t2e(g, "December")
t2e(s, "January")
t2e(l, "February")
```

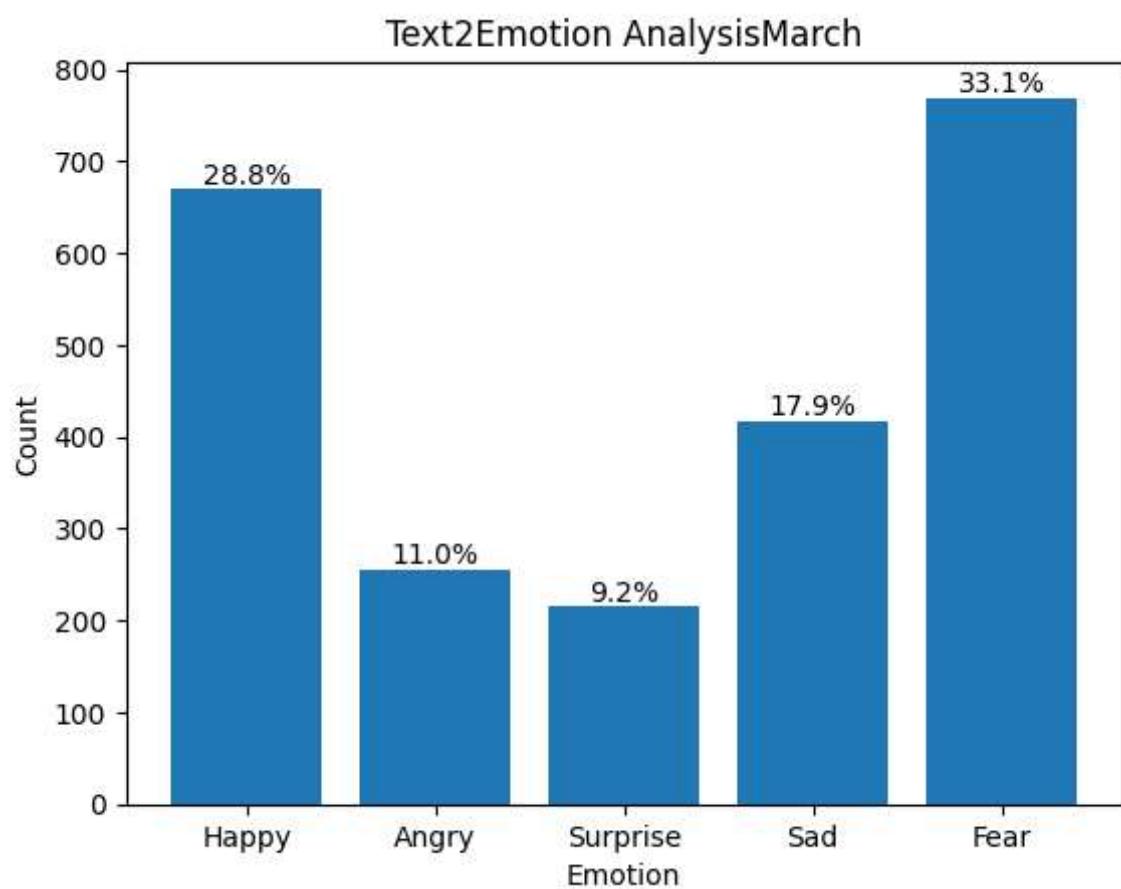
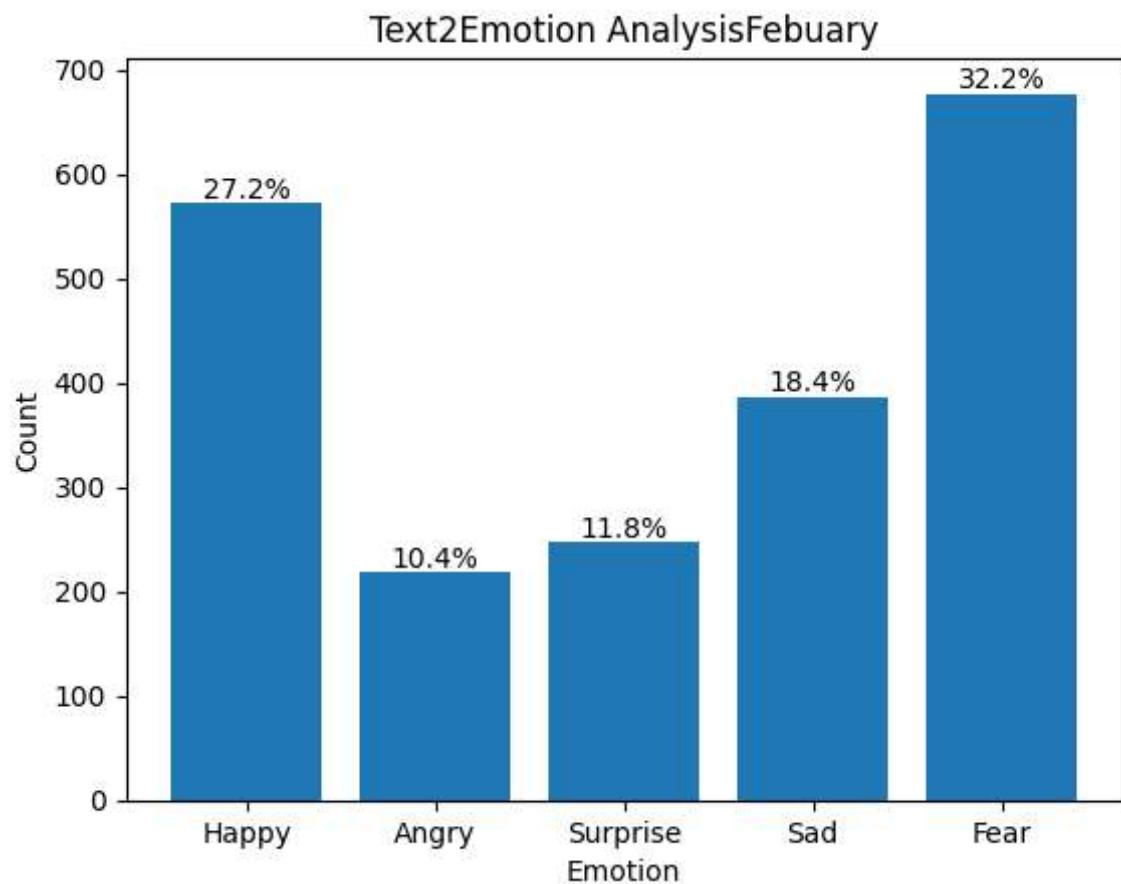
```
t2e(m, "March")
t2e(k, "April")
```

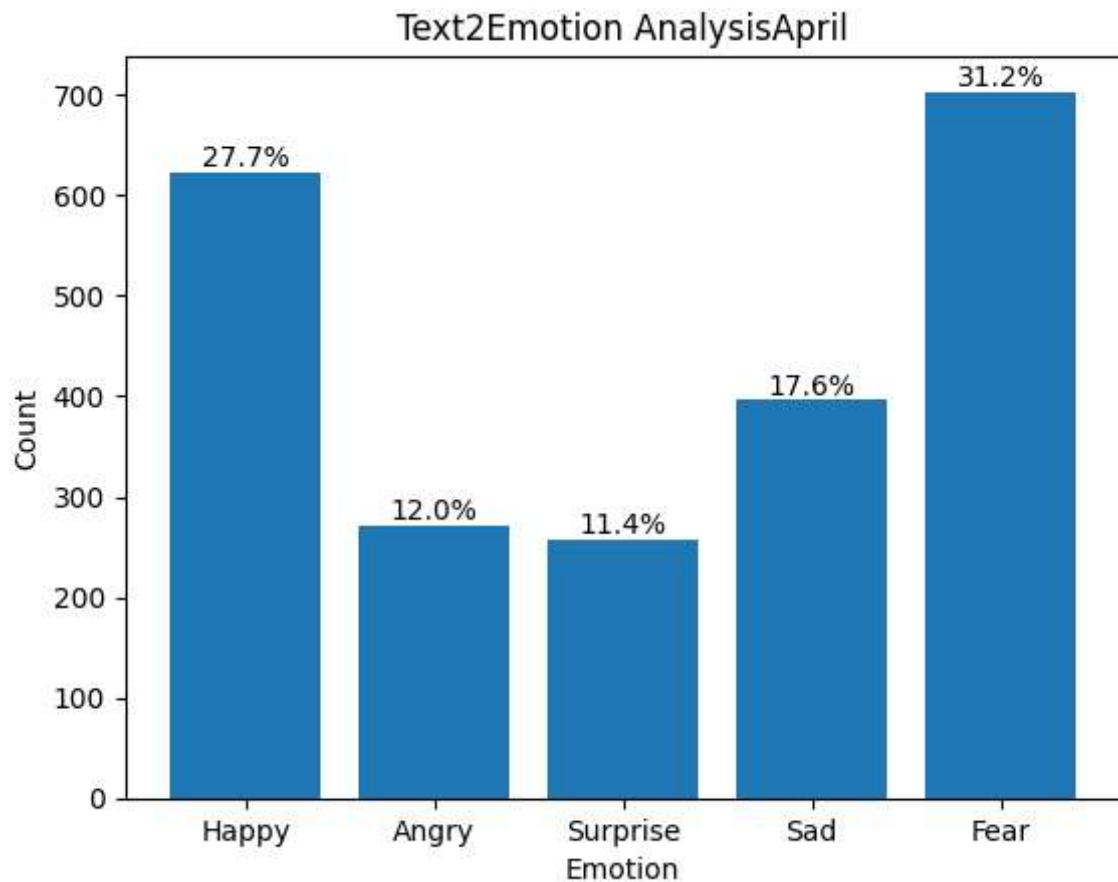
Text2Emotion Analysis December



Text2Emotion Analysis January







W przypadku text2emotion również sytuacja wygląda podobnie ilość średniej emocji są mniej więcej takie same mogą się maksymalnie różnić o ok 3%.

### **Podsumowanie porównanie czasowego**

Oby dwa narzędzia dają średnie emocje w prawie identycznych proporcjach mogą się różnić o ok 2-3% do analiz czasowych i całościowej. Ale tak samo jak w przypadku analizy całościowej nie pokrywają się bardzo text2emotion dają najczęściej strach czyli negatywną emocję potem pozytywna emocija szczęście i potem znowu negatywna emocija smutek. W przypadku nltk vader daje tak jak w przypadku całościowej najwięcej pozytywnych, potem znacznie mniej negatywnych i bardzo mało neutralnych. I tu znowu narzędzia zbytnio się nie pokrywają ze sobą.

### **Podsumowanie**

Oby dwa narzędzia przy porównaniu i analizy dla całościowej czy czasowej analizują podobnie, ale gdy się narzędzia prówna się ze sobą to w ten sam sposób się nie pokrywają choć o ile to zostaje bardzo podobnie dla analizy całościowej i czasowej. Jeśli chodzi o czas analizy tych narzędzi to nltk vader analizował pare sekund, a text2emotion znacznie więcej, analizwał przez ponad godzinę co prawda ma więcej emocji do analizy ale nadal różnica jest bardzo duża. Pozo staje pytanie o ich dokładność, którą jest trudną stwierdzić dokładnie ale nie wydaje mi się że ok 1/3 tweetów wyjawiała strach, większość z nich jest albo za albo przeciw i kłótnie pomiędzy nimi. Najczęściej występują słowa są większość takie o jakiś by ktoś pomyślał słysząc nuclear energy, choć zdarzają się jakieś

mniej oczekiwane jak Biden, choć również mogło się wydawać, że decyzja Niemiec w kwietniu 2023 by na coś wypłyneła ale wygląda na to że nie.

## Bibliografia

- wkłady/laboratoria
- dane z twitter.com
- <https://www.datacamp.com/tutorial/text-analytics-beginners-nltk>
- <https://www.datacamp.com/tutorial/wordcloud-python>
- [https://www.nltk.org/\\_modules/nltk/sentiment/vader.html](https://www.nltk.org/_modules/nltk/sentiment/vader.html)
- <https://betterprogramming.pub/how-to-scrape-tweets-with-snsrape-90124ed006af>
- <https://towardsdatascience.com/text2emotion-python-package-to-detect-emotions-from-textual-data-b2e7b7ce1153>