In [1]:
```
import re
import string
import numpy as np
import random
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from plotly import graph_objs as go
import plotly.express as px
import plotly.figure_factory as ff
from collections import Counter

from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator


import nltk
from nltk.corpus import stopwords

from tqdm import tqdm
import os
import nltk
import spacy
import random
from spacy.util import compounding
from spacy.util import minibatch

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data = pd.read_csv("C:\\Users\\Ansh\\Downloads\\tweets.csv")
# here i am printing fisrt five line of dataset
data.head()
```

Out[2]:

	_key	tweet_author	tweet_text
0	1374140386071961602	Hematopoiesis News	⚕ Scientists conducted a Phase II study of ac...
1	1374032432173842437	Michael Wang, MD	This phase 2 Acalabrutinib-Venetoclax (AV) tri...
2	1373902876553048065	1stOncology	#NICE backs #AstraZenecas #Calquence for #CLL ...
3	1373656782367813635	Toby Eyre	#acalabrutinib is a valuable option in pts int...
4	1372941634334232586	Lymphoma Hub	NICE has recommended the use of acalabrutinib ...

In [3]:
```
data.shape
```

Out[3]:
```
(43347, 3)
```

In [4]:
```
data.describe()
```

Out[4]:

	_key
count	4.334700e+04
mean	1.003292e+18
std	2.313927e+17
min	5.505794e+17
25%	8.006620e+17

50%	1.011270e+18
75%	1.204047e+18
max	1.374141e+18

In [5]:
```
data.isna().sum()
```

Out[5]:
```
_key            0
tweet_author    0
tweet_text      0
dtype: int64
```

In [6]:
```
data.head()
```

Out[6]:

	_key	tweet_author	tweet_text
0	1374140386071961602	Hematopoiesis News	⚕ Scientists conducted a Phase II study of ac...
1	1374032432173842437	Michael Wang, MD	This phase 2 Acalabrutinib-Venetoclax (AV) tri...
2	1373902876553048065	1stOncology	#NICE backs #AstraZenecas #Calquence for #CLL ...
3	1373656782367813635	Toby Eyre	#acalabrutinib is a valuable option in pts int...
4	1372941634334232586	Lymphoma Hub	NICE has recommended the use of acalabrutinib ...

In [7]:
```
def edits1(word):
    letters='abcdefghijklmnopqrstuvwxyz'
    splits=[(word[:i], word[i:]) for i in range(len(word)+1)]
    deletes=[L+R[1:] for L,R in splits if R]
    transposes=[L+R[1] +R[0] + R[2:] for L,R in splits if len(R)>1]
    replaces = [L+c+R[1:] for L,R in splits if R for c in letters]
    inserts = [L+c+ R for L,R in splits for c in letters]
    return set(deletes+transposes+replaces+inserts)
def edits2(word):
    return(e2 for e1 in edits1(word) for e2 in edits1(e1))
```

In [8]:
```
def remove_spaces(text):
    text=text.strip()
    text=text.split()
    return ' '.join(text)
```

In [9]:
```
contraction = {'cause':'because',
               'aint': 'am not',
               'aren\'t': 'are not'}

def mapping_replacer(x,dic):
    for words in dic.keys():
        if ' ' + words + ' ' in x:
            x=x.replace(' '+ words +' ' ,' '+dic[words]+' ' )
    return x
```

In [10]:
```
import nltk
nltk.download('punkit')
from nltk.tokenize import word_tokenize
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.stem.lancaster import LancasterStemmer

nltk.LancasterStemmer
ls = LancasterStemmer()
lem = WordNetLemmatizer()
```

```
def lexicon_normalization(text):
    words = word_tokenize(text)


    # 1- Stemming
    words_stem = [ls.stem(w) for w in words]

    # 2- Lemmatization
    words_lem = [lem.lemmatize(w) for w in words_stem]
    return words_lem
```

```
[nltk_data] Error loading punkit: Package 'punkit' not found in index
```

In [11]:
```
import emoji
import re
#from emot.emo_unicode import UNICODE_EMO
def convert_emojis(text):
    for emot in emoji.UNICODE_EMOJI:
        text = re.sub(r'('+emot+')', "_".join(emoji.UNICODE_EMOJI[emot].replace(",","").
    return text
```

In [12]:
```
def clean_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuati
    and remove words containing numbers.'''
    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 = re.sub('\'','', text)

    return text
```

In [13]:
```
from collections import Counter
def remove_stopword(text):
    stop_words = stopwords.words('english')
    stopwords_dict = Counter(stop_words)
    text = ' '.join([word for word in text.split() if word not in stopwords_dict])
    return text
```

In [14]:
```
def tokenise(text):
    words = word_tokenize(text)
    return words
```

In [15]:
```
import re
data['tweet_text'] = data['tweet_text'].map(lambda x: re.sub(r'\W+', ' ', x))
data['tweet_text'] = data['tweet_text'].replace(r'\W+', ' ', regex=True)
```

In [16]:
```
data.tweet_text
```

Out[16]:
```
0          Scientists conducted a Phase II study of acal...
1         This phase 2 Acalabrutinib Venetoclax AV trial...
2          NICE backs AstraZenecas Calquence for CLL htt...
3          acalabrutinib is a valuable option in pts int...
4         NICE has recommended the use of acalabrutinib ...
                                   ...
43342     Hanging out with Friends FF CLL Happiness http...
43343     Hanging out with Friends FF CLL Happiness http...
43344     Zusatznutzen von Idelalisib ist weder für CLL ...
43345      Hematología PTK2 EXPRESSION AND IMMUNOCHEMOTH...
43346      Hematología MUTATIONS IN TLR MYD88 PATHWAY ID...
Name: tweet_text, Length: 43347, dtype: object
```

In [17]:
```
data.head()
```

Out[17]:

	_key	tweet_author	tweet_text
0	1374140386071961602	Hematopoiesis News	Scientists conducted a Phase II study of acal...
1	1374032432173842437	Michael Wang, MD	This phase 2 Acalabrutinib Venetoclax AV trial...
2	1373902876553048065	1stOncology	NICE backs AstraZenecas Calquence for CLL htt...
3	1373656782367813635	Toby Eyre	acalabrutinib is a valuable option in pts int...
4	1372941634334232586	Lymphoma Hub	NICE has recommended the use of acalabrutinib ...

In [18]:
```
df = data.groupby(by=['tweet_author']).count()
```

In [19]:
```
df.rename(columns = {'tweet_author':'Author', 'tweet_text':''}, inplace = True)
```

In [20]:
```
df
```

Out[20]:

	_key	
tweet_author		
Camilla White	2	2
Emilie Thompson	2	2
Hannah Wright	2	2
Yvianna ð	1	1
#DestroyTheAadhaar TwiLightOFTheGODS	1	1
...	...	...
😩	1	1
🦋NasRaf🦋	1	1
🐄🐄 Hömicidal Barcode 🐄🐄	1	1
🦠 Adam Grant	1	1
🌶 Manuela	1	1

9292 rows × 2 columns

In [21]:
```
data['tweet_text']=data['tweet_text'].apply(lambda x: mapping_replacer(x, contraction))
```

In [22]:
```
data['tweet_text'] = data['tweet_text'].apply(lambda x:clean_text(x))
```

In [23]:
```
data['tweet_text']=data['tweet_text'].apply(lambda x: remove_stopword(x))
```

In [24]:
```
data['tweet_text']=data['tweet_text'].apply(lambda x: lexicon_normalization(x))
```

In [25]:
```
data.head()
```

Out[25]:

	_key	tweet_author	tweet_text
0	1374140386071961602	Hematopoiesis News	[sci, conduc, phas, ii, study, acalabrutinib, ...
1	1374032432173842437	Michael Wang, MD	[phas, acalabrutinib, venetoclax, av, tri, sti...

2	1373902876553048065	1stOncology	[nic, back, astrazeneca, calqu, cll, http, co]
3	1373656782367813635	Toby Eyre	[acalabrutinib, valu, opt, pt, intol, ibrutini...
4	1372941634334232586	Lymphoma Hub	[nic, recommend, u, acalabrutinib, paty, tre, ...

Polarity Of Sentiment

In [26]:
```
from wordcloud import WordCloud, STOPWORDS , ImageColorGenerator

from textblob import TextBlob

def get_tweet_sentiment(tweet):
    '''
    Utility function to classify sentiment of passed tweet
    using textblob's sentiment method
    '''
    # create TextBlob object of passed tweet text
    analysis = TextBlob(tweet)

    # set sentiment
    if analysis.sentiment.polarity > 0:
        return 'positive'
    elif analysis.sentiment.polarity == 0:
        return 'neutral'
    else:
        return 'negative'
```

In [27]:
```
data['sentiment']=data['tweet_text'].apply(lambda x: get_tweet_sentiment(' '.join(x)))
```

In [28]:
```
data.to_csv(path_or_buf='C:\\Users\\Ansh\\Desktop\\Akaike NLP Assignment//Objective1.csv
```

In [29]:
```
Positive_sent = data[data['sentiment']=='positive']
Negative_sent = data[data['sentiment']=='negative']
Neutral_sent = data[data['sentiment']=='neutral']
```

In [30]:
```
print('Number of tweets with positive sentiment', Positive_sent['sentiment'].count())
print('Number of tweets with negative sentiment', Negative_sent['sentiment'].count())
print('Number of tweets with neutral sentiment', Neutral_sent['sentiment'].count())
```

```
Number of tweets with positive sentiment 12832
Number of tweets with negative sentiment 3207
Number of tweets with neutral sentiment 27308
```

Here we got to know that for finding out the polarity we need to clean the data, remove stopwords and tokenise the tweets

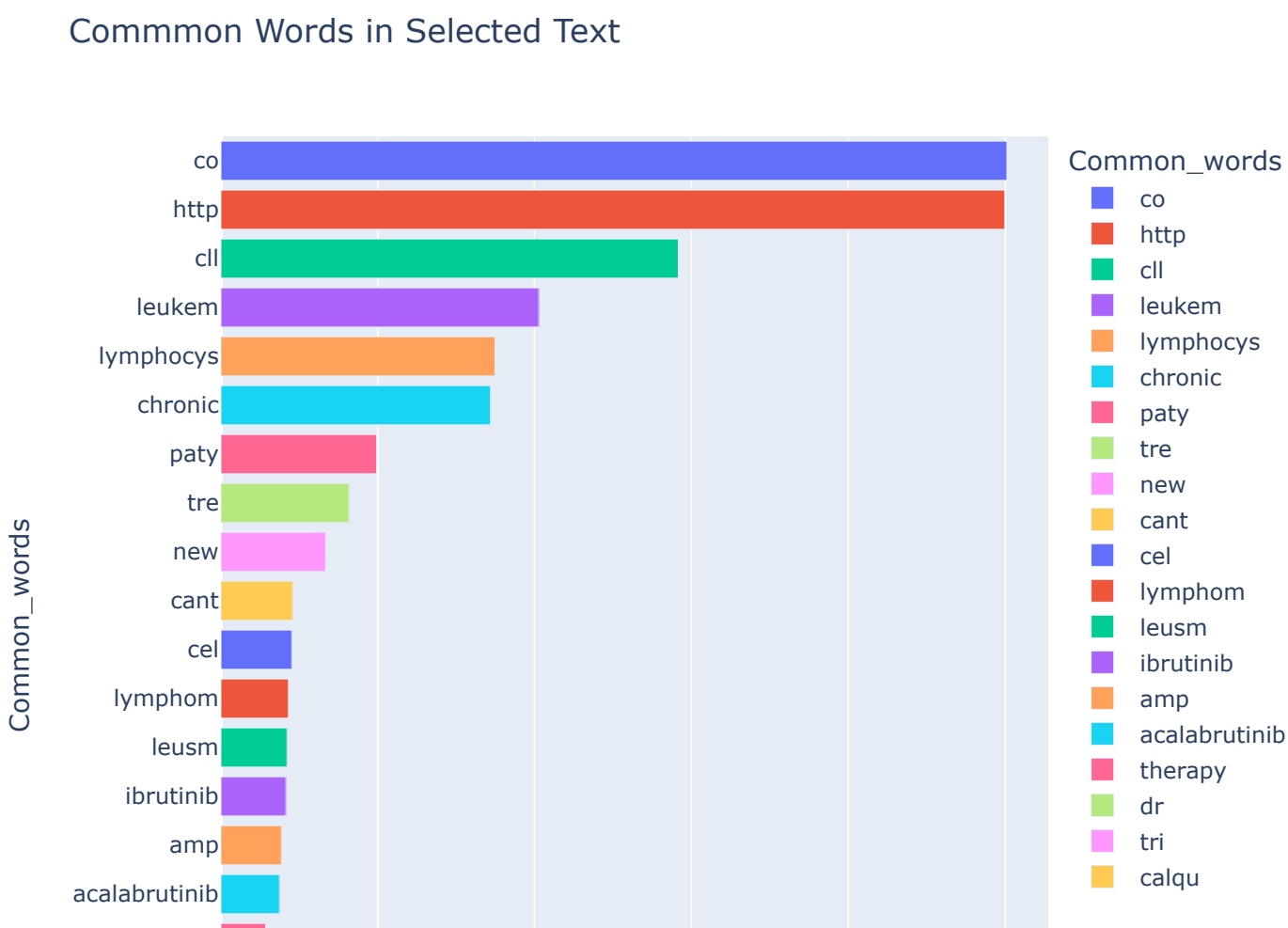
In [31]:
```
top = Counter([item for sublist in data['tweet_text'] for item in sublist])
temp = pd.DataFrame(top.most_common(20))
temp.columns = ['Common_words','count']
temp.style.background_gradient(cmap='Blues')
```

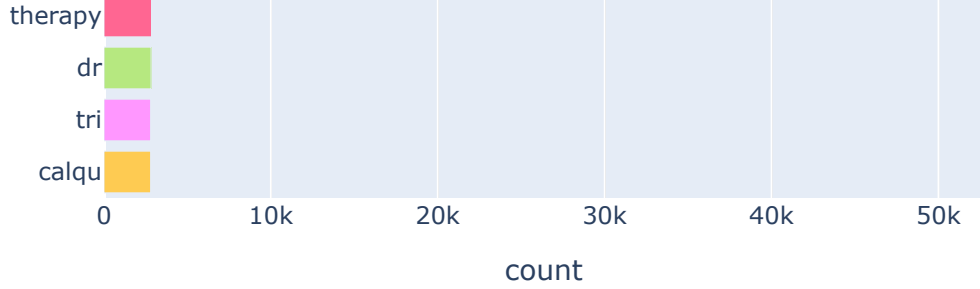
Out[31]:

	Common_words	count
0	co	50086
1	http	49950
2	cll	29147
3	leukem	20255

4	lymphocys	17459
5	chronic	17181
6	paty	9923
7	tre	8165
8	new	6675
9	cant	4565
10	cel	4511
11	lymphom	4303
12	leusm	4209
13	ibrutinib	4142
14	amp	3850
15	acalabrutinib	3727
16	therapy	2846
17	dr	2821
18	tri	2791
19	calqu	2791

In [32]:
```
fig = px.bar(temp, x="count", y="Common_words", title='Commmon Words in Selected Text',
             width=700, height=700,color='Common_words')
fig.show()
```



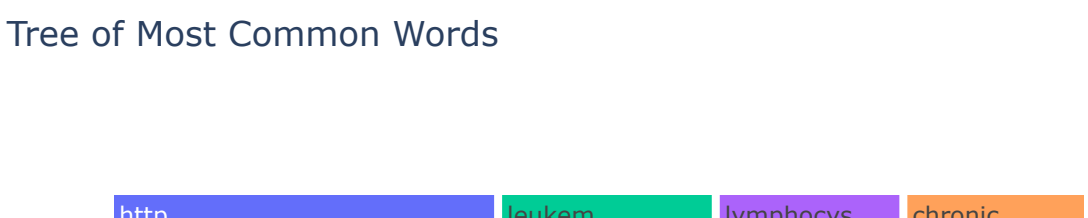


In [33]:
```
top = Counter([item for sublist in data['tweet_text'] for item in sublist])
temp = pd.DataFrame(top.most_common(20))
temp = temp.iloc[1:,:]
temp.columns = ['Common_words','count']
temp.style.background_gradient(cmap='Purples')
```

Out[33]:

	Common_words	count
1	http	49950
2	cll	29147
3	leukem	20255
4	lymphocys	17459
5	chronic	17181
6	paty	9923
7	tre	8165
8	new	6675
9	cant	4565
10	cel	4511
11	lymphom	4303
12	leusm	4209
13	ibrutinib	4142
14	amp	3850
15	acalabrutinib	3727
16	therapy	2846
17	dr	2821
18	tri	2791
19	calqu	2791

In [34]:
```
fig = px.treemap(temp, path=['Common_words'], values='count',title='Tree of Most Common
fig.show()
```



In [35]:
```
from wordcloud import WordCloud, STOPWORDS , ImageColorGenerator

from textblob import TextBlob

def get_tweet_sentiment(tweet):
    '''
    Utility function to classify sentiment of passed tweet
    using textblob's sentiment method
    '''
    # create TextBlob object of passed tweet text
    analysis = TextBlob(tweet)

    # set sentiment
    if analysis.sentiment.polarity > 0:
        return 'positive'
    elif analysis.sentiment.polarity == 0:
        return 'neutral'
    else:
        return 'negative'
```

In [36]:
```
data['sentiment']=data['tweet_text'].apply(lambda x: get_tweet_sentiment(' '.join(x)))
```

In [37]:
```
data
```

Out[37]:

	_key	tweet_author	tweet_text	sentiment
0	1374140386071961602	Hematopoiesis News	[sci, conduc, phas, ii, study, acalabrutinib, ...	neutral
1	1374032432173842437	Michael Wang, MD	[phas, acalabrutinib, venetoclax, av, tri, sti...	neutral
2	1373902876553048065	1stOncology	[nic, back, astrazeneca, calqu, cll, http, co]	neutral
3	1373656782367813635	Toby Eyre	[acalabrutinib, valu, opt, pt, intol, ibrutini...	neutral
4	1372941634334232586	Lymphoma Hub	[nic, recommend, u, acalabrutinib, paty, tre, ...	neutral
...	...	...	...	...
43342	551103473643945985	Joy is a Lifestyle	[hang, friend, ff, cll, happy, http, co]	positive
43343	551102786675290112	𝒞𝓇𝒾𝓏𝓏𝓎 𝒫𝑒𝓇𝓇𝓎 ♀	[hang, friend, ff, cll, happy, http, co]	positive

43344	550969541186953217	IQWiG	[zusatznutz, von, idelalisib, ist, wed, für, c...	neutral
43345	550941480525635584	Medibooks	[hematologí, express, immunochemotherapy, outc...	neutral
43346	550579446537678849	Medibooks	[hematologí, mut, tlr, pathway, ident, subset,...	positive

43347 rows × 4 columns

In [ ]:
```
data.to_csv("C:\\Users\\Ansh\Desktop\\Akaike NLP Assignment\\Objective2.csv")
```

In [38]:
```
Positive_sent = data[data['sentiment']=='positive']
Negative_sent = data[data['sentiment']=='negative']
Neutral_sent = data[data['sentiment']=='neutral']
```

In [39]:
```
print('Number of tweets with positive sentiment', Positive_sent['sentiment'].count())
print('Number of tweets with negative sentiment', Negative_sent['sentiment'].count())
print('Number of tweets with neutral sentiment', Neutral_sent['sentiment'].count())
```

```
Number of tweets with positive sentiment 12832
Number of tweets with negative sentiment 3207
Number of tweets with neutral sentiment 27308
```

In [40]:
```
#MosT common positive words
top = Counter([item for sublist in Positive_sent['tweet_text'] for item in sublist])
temp_positive = pd.DataFrame(top.most_common(20))
temp_positive.columns = ['Common_words','count']
temp_positive.style.background_gradient(cmap='Greens')
```

Out[40]:

	Common_words	count
0	co	15067
1	http	15016
2	cll	9587
3	new	6383
4	leukem	6121
5	lymphocys	5281
6	chronic	5172
7	paty	3612
8	tre	2945
9	leusm	2077
10	hematolog	1748
11	cant	1414
12	artic	1408
13	ibrutinib	1383
14	cel	1342
15	lymphom	1285
16	high	1263
17	drug	1205
18	amp	1106

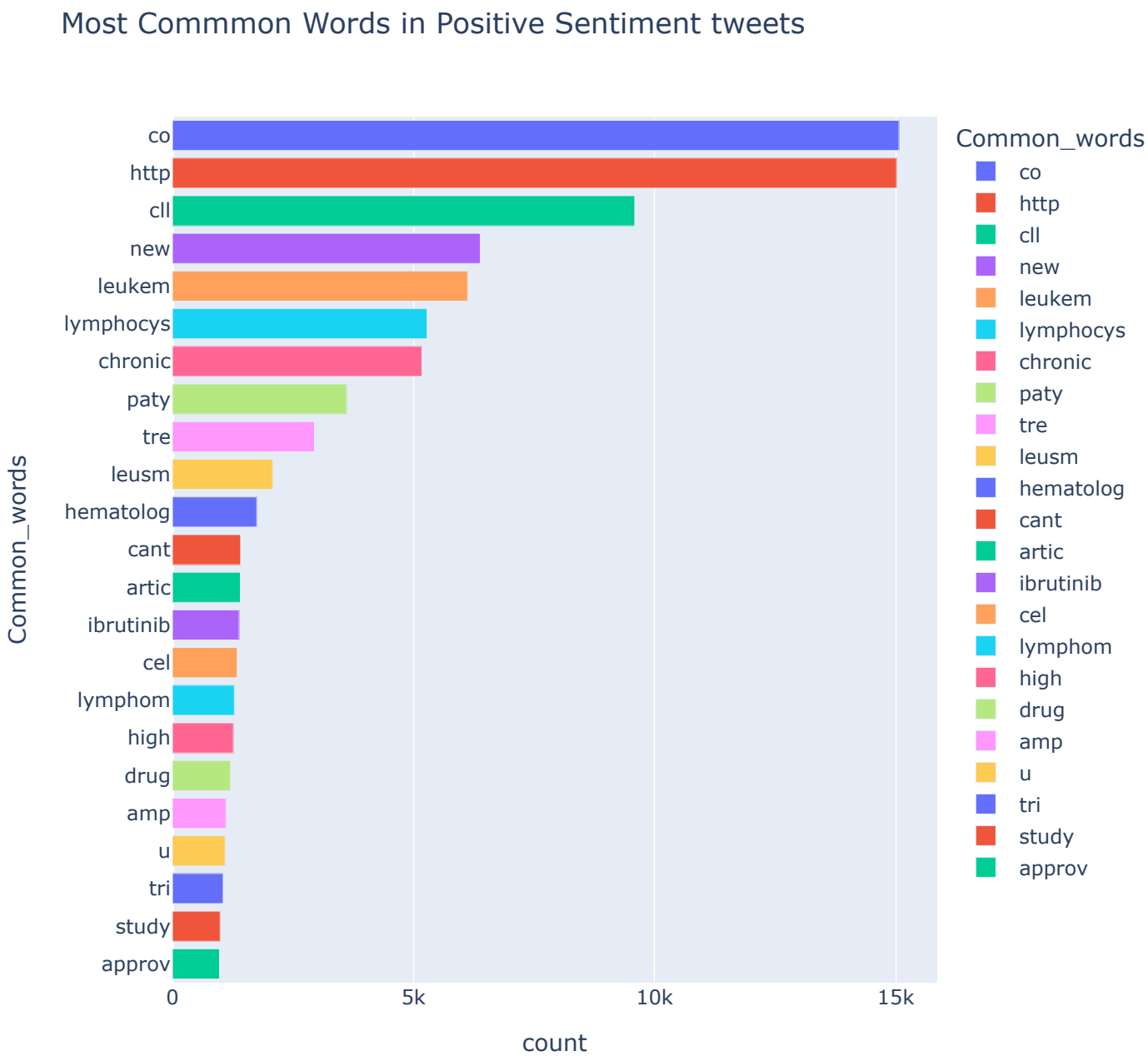
19	u	1090

In [41]:
```
import numpy as np
top = Counter([item for sublist in Positive_sent['tweet_text'] for item in sublist])
temp_positive = pd.DataFrame(top.most_common(23))
temp_positive.columns = ['Common_words','count']
temp_positive['Common_words'] = temp_positive['Common_words'].map(lambda x: re.sub(r'\W+
temp_positive['Common_words'] = temp_positive['Common_words'].replace(r'\W+', '', regex=
temp_positive['Common_words'] = temp_positive['Common_words'].apply(lambda x:remove_spac
temp_positive=temp_positive[~temp_positive['Common_words'].isin(['s','gre','`',' * '])]
mask1 = temp_positive.Common_words.str.contains('[a-zA-Z]')
mask2 = temp_positive.Common_words.notna()
temp_positive = temp_positive[mask1 | mask2]
temp_positive.Common_words =  temp_positive.Common_words.str.replace(r"\s+", "").replace
temp_positive=temp_positive.dropna()


temp_positive.style.background_gradient(cmap='Greens')
```

Out[41]:

	Common_words	count
0	co	15067
1	http	15016
2	cll	9587
3	new	6383
4	leukem	6121
5	lymphocys	5281
6	chronic	5172
7	paty	3612
8	tre	2945
9	leusm	2077
10	hematolog	1748
11	cant	1414
12	artic	1408
13	ibrutinib	1383
14	cel	1342
15	lymphom	1285
16	high	1263
17	drug	1205
18	amp	1106
19	u	1090
20	tri	1052
21	study	990
22	approv	976

In [42]:
```
fig = px.bar(temp_positive, x="count", y="Common_words", title='Most Commmon Words in Po
             width=700, height=700,color='Common_words')
fig.show()
```



In [43]:

```
#MosT common negative words
top = Counter([item for sublist in Negative_sent['tweet_text'] for item in sublist])
temp_negative = pd.DataFrame(top.most_common(20))
temp_negative = temp_negative.iloc[1:,:]
temp_negative.columns = ['Common_words','count']
temp_negative.style.background_gradient(cmap='Reds')
```

Out[43]:

	Common_words	count
1	http	3834
2	cll	2338
3	leukem	1819
4	lymphocys	1556
5	chronic	1551
6	cant	1167

7	paty	973
8	tre	682
9	amp	670
10	adult	597
11	long	463
12	slow	438
13	cel	434
14	grow	414
15	lymphocyt	408
16	prim	401
17	ibrutinib	384
18	involv	380
19	common	364

In [44]:

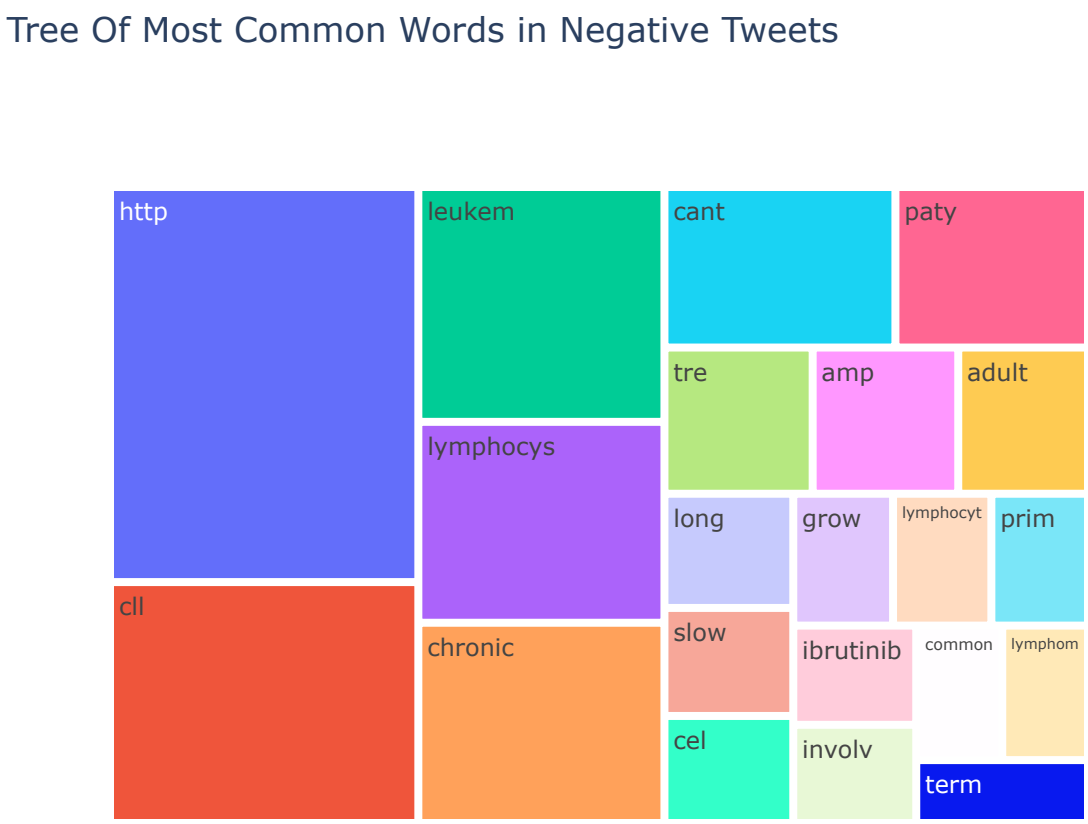
```
#MosT common negative words
top = Counter([item for sublist in Negative_sent['tweet_text'] for item in sublist])
temp_negative = pd.DataFrame(top.most_common(22))
temp_negative = temp_negative.iloc[1:,:]
temp_negative.columns = ['Common_words','count']

#Data cleaning
temp_negative['Common_words'] = temp_negative['Common_words'].map(lambda x: re.sub(r'\W+
temp_negative['Common_words'] = temp_negative['Common_words'].replace(r'\W+', '', regex=
temp_negative=temp_negative[~temp_negative['Common_words'].isin(['s','t'])] #new line re
#mask1 = temp_negative.Common_words.str.contains('[a-zA-Z]')
#mask2 = temp_negative.Common_words.notna()
#temp_negative = temp_negative[mask1 | mask2]

temp_negative.Common_words =  temp_negative.Common_words.replace("", np.nan)
temp_negative = temp_negative.dropna(subset=['Common_words'])

temp_negative.style.background_gradient(cmap='Reds')
```

Out[44]:

	Common_words	count
1	http	3834
2	cll	2338
3	leukem	1819
4	lymphocys	1556
5	chronic	1551
6	cant	1167
7	paty	973
8	tre	682
9	amp	670
10	adult	597
11	long	463
12	slow	438

13	cel	434
14	grow	414
15	lymphocyt	408
16	prim	401
17	ibrutinib	384
18	involv	380
19	common	364
20	lymphom	363
21	term	340

In [45]:
```
fig = px.treemap(temp_negative, path=['Common_words'], values='count',title='Tree Of Mos
fig.show()
```



In [46]:
```
#MosT common Neutral words
top = Counter([item for sublist in Neutral_sent['tweet_text'] for item in sublist])
temp_neutral = pd.DataFrame(top.most_common(20))
temp_neutral = temp_neutral.loc[1:,:]
temp_neutral.columns = ['Common_words','count']
temp_neutral.style.background_gradient(cmap='YlOrBr')
```

Out[46]:

	Common_words	count
1	http	31100
2	cll	17222

3	leukem	12315
4	lymphocys	10622
5	chronic	10458
6	paty	5338
7	tre	4538
8	cel	2735
9	acalabrutinib	2706
10	lymphom	2655
11	ibrutinib	2375
12	amp	2074
13	cant	1984
14	leusm	1876
15	calqu	1869
16	dr	1814
17	therapy	1714
18	venetoclax	1602
19	relaps	1438

In [47]:

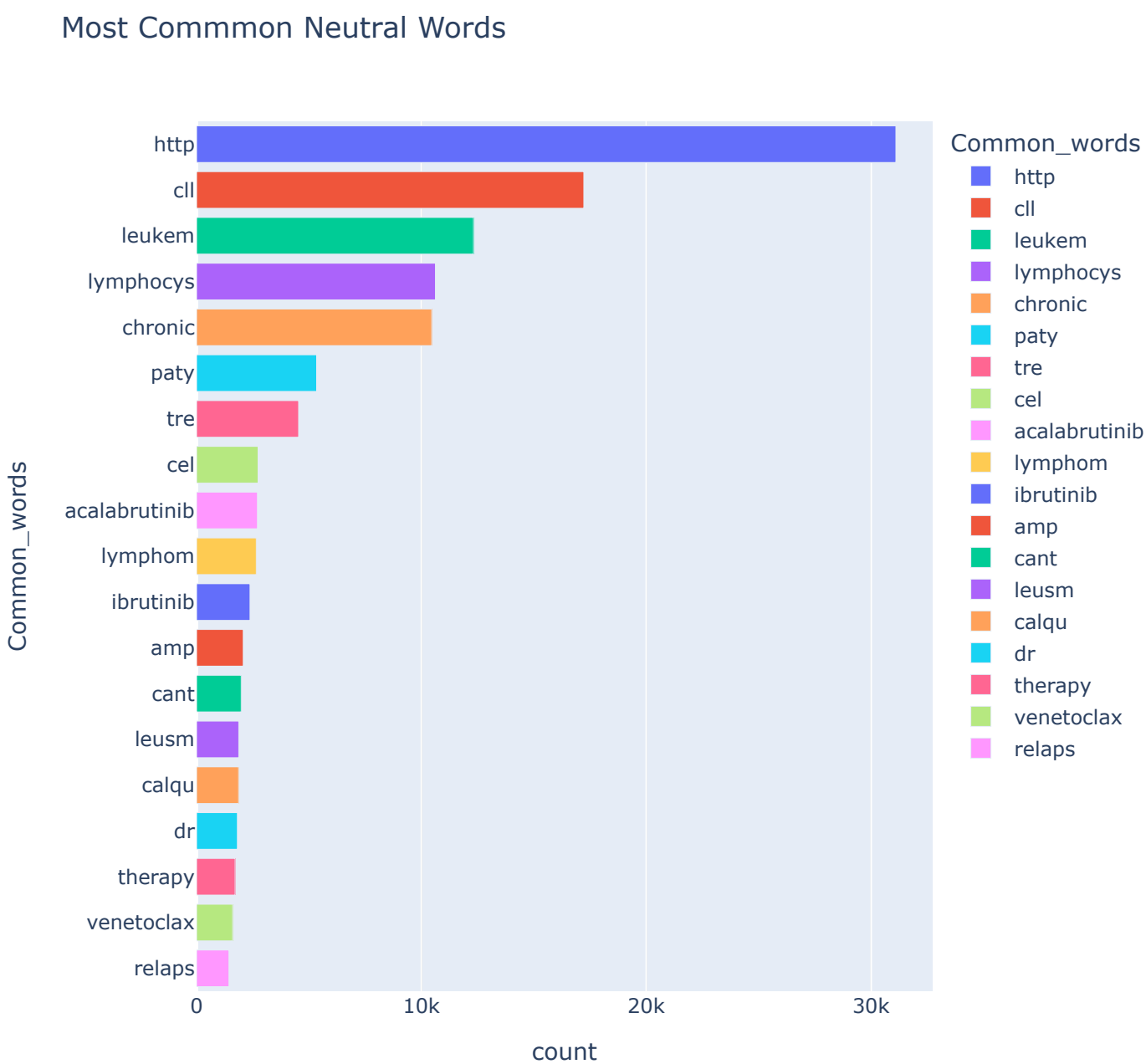
```
top = Counter([item for sublist in Neutral_sent['tweet_text'] for item in sublist])
temp_neutral = pd.DataFrame(top.most_common(20))
temp_neutral = temp_neutral.loc[1:,:]
temp_neutral.columns = ['Common_words','count']

#Data cleaning
temp_neutral['Common_words'] = temp_neutral['Common_words'].map(lambda x: re.sub(r'\W+',
temp_neutral['Common_words'] = temp_neutral['Common_words'].replace(r'\W+', '', regex=Tr
temp_neutral=temp_neutral[~temp_neutral['Common_words'].isin(['s'])] #new line removing

temp_neutral.Common_words =  temp_neutral.Common_words.replace("", np.nan)
temp_neutral = temp_neutral.dropna(subset=['Common_words'])

temp_neutral.style.background_gradient(cmap='YlOrBr')
```

Out[47]:

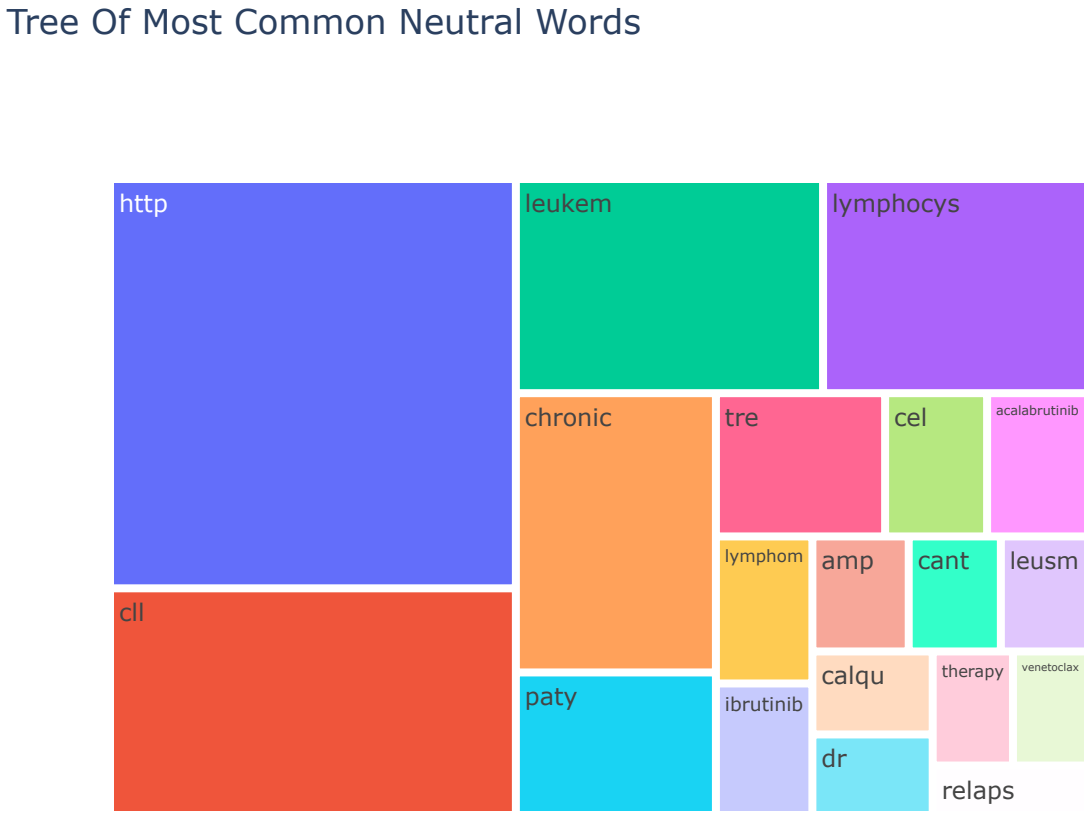
	Common_words	count
1	http	31100
2	cll	17222
3	leukem	12315
4	lymphocys	10622
5	chronic	10458
6	paty	5338
7	tre	4538
8	cel	2735
9	acalabrutinib	2706
10	lymphom	2655

11	ibrutinib	2375
12	amp	2074
13	cant	1984
14	leusm	1876
15	calqu	1869
16	dr	1814
17	therapy	1714
18	venetoclax	1602
19	relaps	1438

In [48]:
```
fig = px.bar(temp_neutral, x="count", y="Common_words", title='Most Commmon Neutral Word
             width=700, height=700,color='Common_words')
fig.show()
```



In [49]:
```
fig = px.treemap(temp_neutral, path=['Common_words'], values='count',title='Tree Of Most
fig.show()
```



In [50]:
```
raw_text = [word for word_list in data['tweet_text'] for word in word_list]
```

In [51]:
```
def words_unique(sentiment,numwords,raw_words):
    '''
    Input:
        segment - Segment category (ex. 'Neutral');
        numwords - how many specific words do you want to see in the final result;
        raw_words - list  for item in train_data[train_data.segments == segments]['temp_
    Output:
        dataframe giving information about the name of the specific ingredient and how m

    '''
    allother = []
    for item in data[data.sentiment != sentiment]['tweet_text']:
        for word in item:
            allother .append(word)
    allother  = list(set(allother ))

    specificnonly = [x for x in raw_text if x not in allother]

    mycounter = Counter()

    for item in data[data.sentiment == sentiment]['tweet_text']:
        for word in item:
            mycounter[word] += 1
    keep = list(specificnonly)

    for word in list(mycounter):
```

```
        if word not in keep:
            del mycounter[word]

    Unique_words = pd.DataFrame(mycounter.most_common(numwords), columns = ['words','cou

    return Unique_words
```

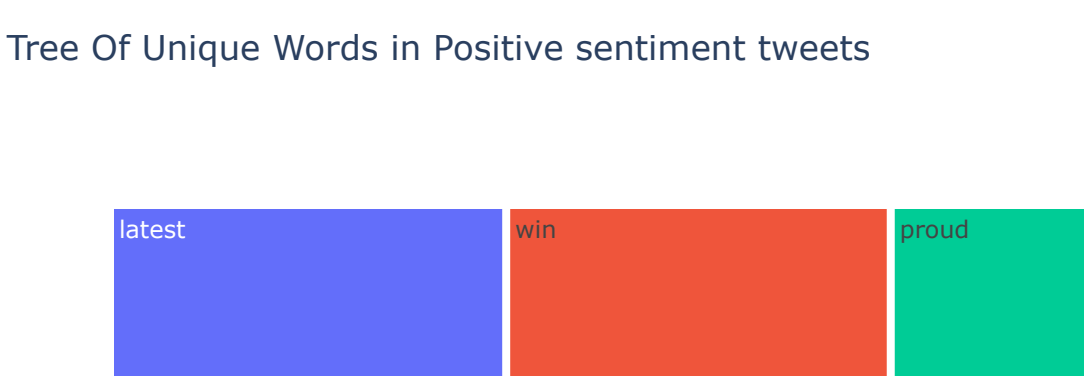
In [52]:
```
Unique_Positive= words_unique('positive', 20, raw_text)
print("The top 20 unique words in Positive Tweets are:")
Unique_Positive.style.background_gradient(cmap='Greens')
```

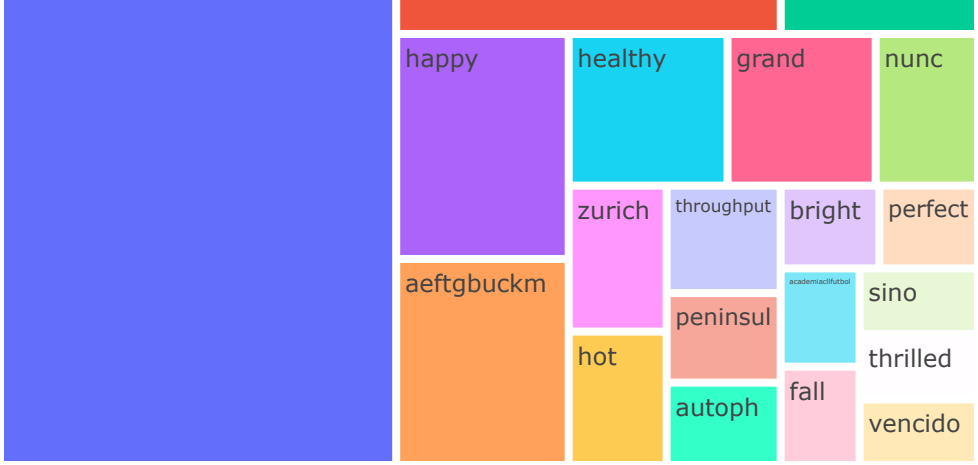
```
The top 20 unique words in Positive Tweets are:
```

Out[52]:

	words	count
0	latest	617
1	win	193
2	proud	99
3	happy	95
4	aeftgbuckm	87
5	healthy	59
6	grand	55
7	nunc	38
8	zurich	35
9	hot	32
10	throughput	30
11	peninsul	25
12	autoph	23
13	bright	20
14	perfect	20
15	thrilled	19
16	fall	19
17	sino	19
18	vencido	19
19	academiacllfutbol	19

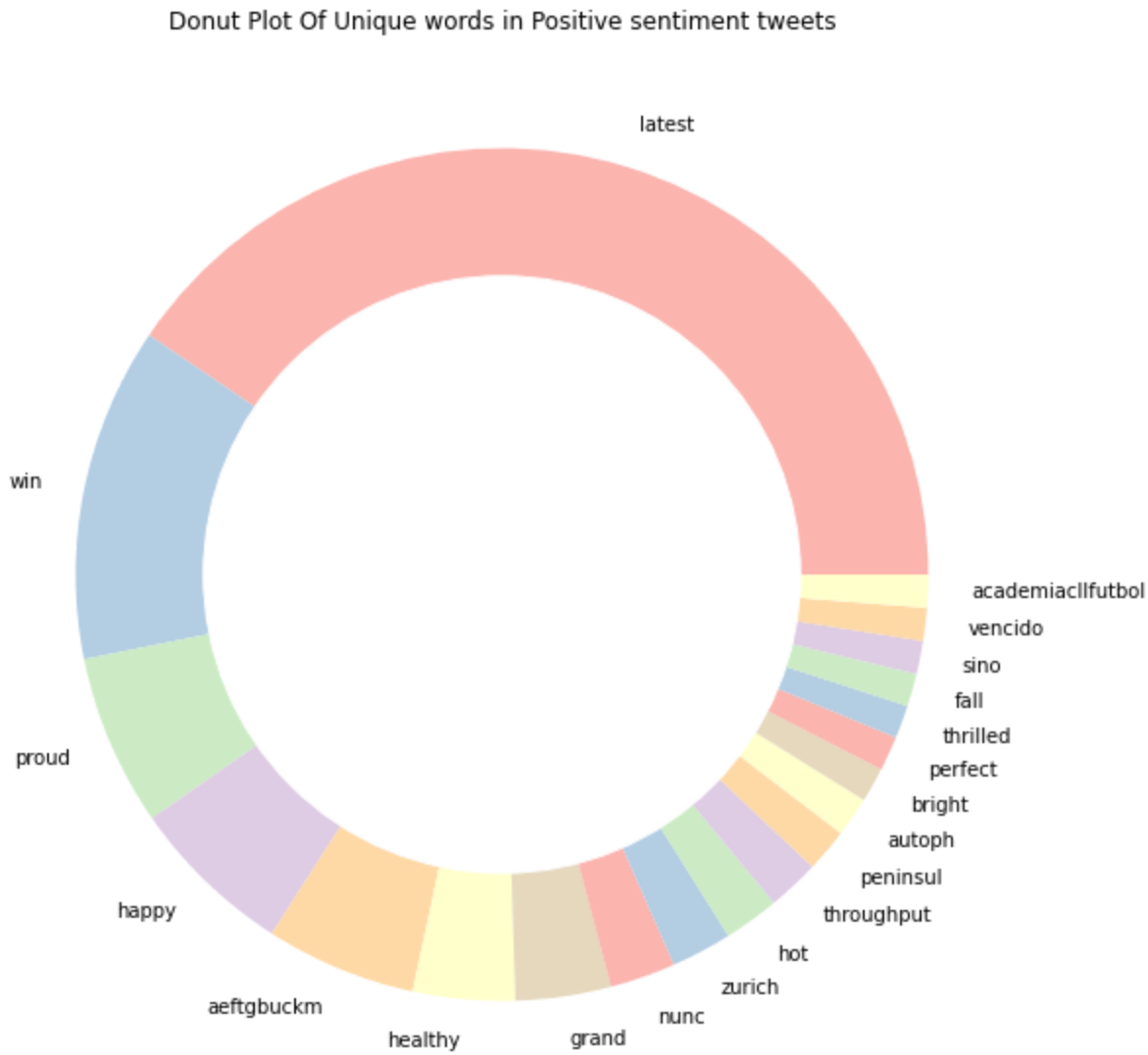
In [53]:
```
fig = px.treemap(Unique_Positive, path=['words'], values='count',title='Tree Of Unique W
fig.show()
```





In [54]:

```
from palettable.colorbrewer.qualitative import Pastel1_7
plt.figure(figsize=(16,10))
my_circle=plt.Circle((0,0), 0.7, color='white')
plt.pie(Unique_Positive['count'], labels=Unique_Positive.words, colors=Pastel1_7.hex_col
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.title('Donut Plot Of Unique words in Positive sentiment tweets')
plt.show()
```



In [55]:
```
Unique_Negative= words_unique('negative', 10, raw_text)
print("The top 10 unique words in Negative Tweets are:")
Unique_Negative.style.background_gradient(cmap='Reds')
```

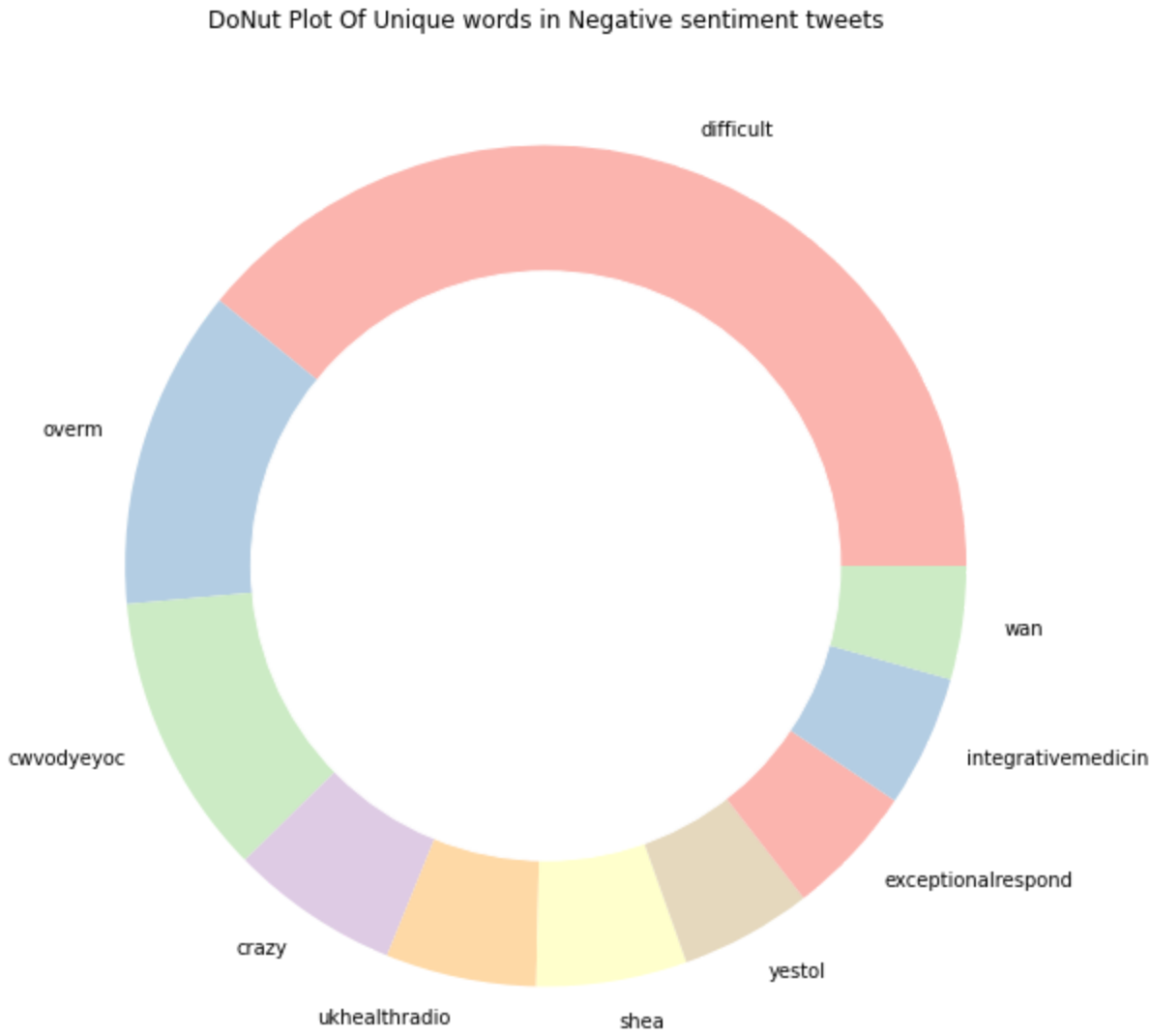
```
The top 10 unique words in Negative Tweets are:
```

Out[55]:

	words	count
0	difficult	54
1	overm	17
2	cwvodyeyoc	15
3	crazy	9
4	ukhealthradio	8
5	shea	8
6	yestol	7
7	exceptionalrespond	7
8	integrativemedicin	7
9	wan	6

In [56]:
```
from palettable.colorbrewer.qualitative import Pastel1_7
plt.figure(figsize=(16,10))
my_circle=plt.Circle((0,0), 0.7, color='white')
```

```
plt.rcParams['text.color'] = 'black'
plt.pie(Unique_Negative['count'], labels=Unique_Negative.words, colors=Pastel1_7.hex_col
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.title('DoNut Plot Of Unique words in Negative sentiment tweets')
plt.show()
```



In [57]:
```
Unique_Neutral= words_unique('neutral', 10, raw_text)
print("The top 10 unique words in Neutral Tweets are:")
Unique_Neutral.style.background_gradient(cmap='YlOrBr')
```

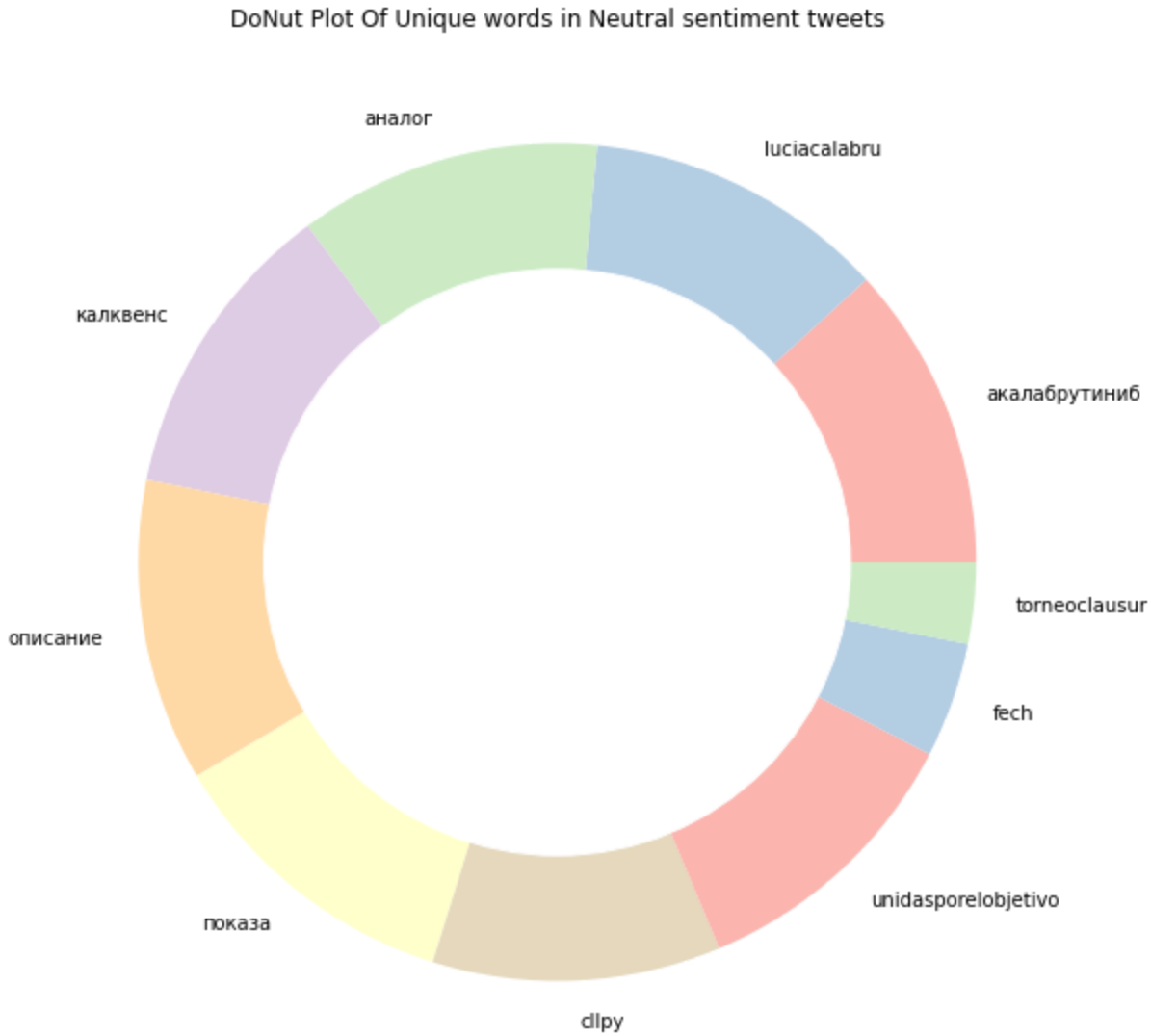
```
The top 10 unique words in Neutral Tweets are:
```

Out[57]:

	words	count
0	акалабрутиниб	528
1	luciacalabru	523
2	аналог	523
3	калквенс	523
4	описание	523
5	показа	523
6	cllpy	498
7	unidasporelobjetivo	497
8	fech	199

9	torneoclausur	140

In [58]:
```
from palettable.colorbrewer.qualitative import Pastel1_7
plt.figure(figsize=(16,10))
my_circle=plt.Circle((0,0), 0.7, color='white')
plt.pie(Unique_Neutral['count'], labels=Unique_Neutral.words, colors=Pastel1_7.hex_color
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.title('DoNut Plot Of Unique words in Neutral sentiment tweets')
plt.show()
```



Word Cloud

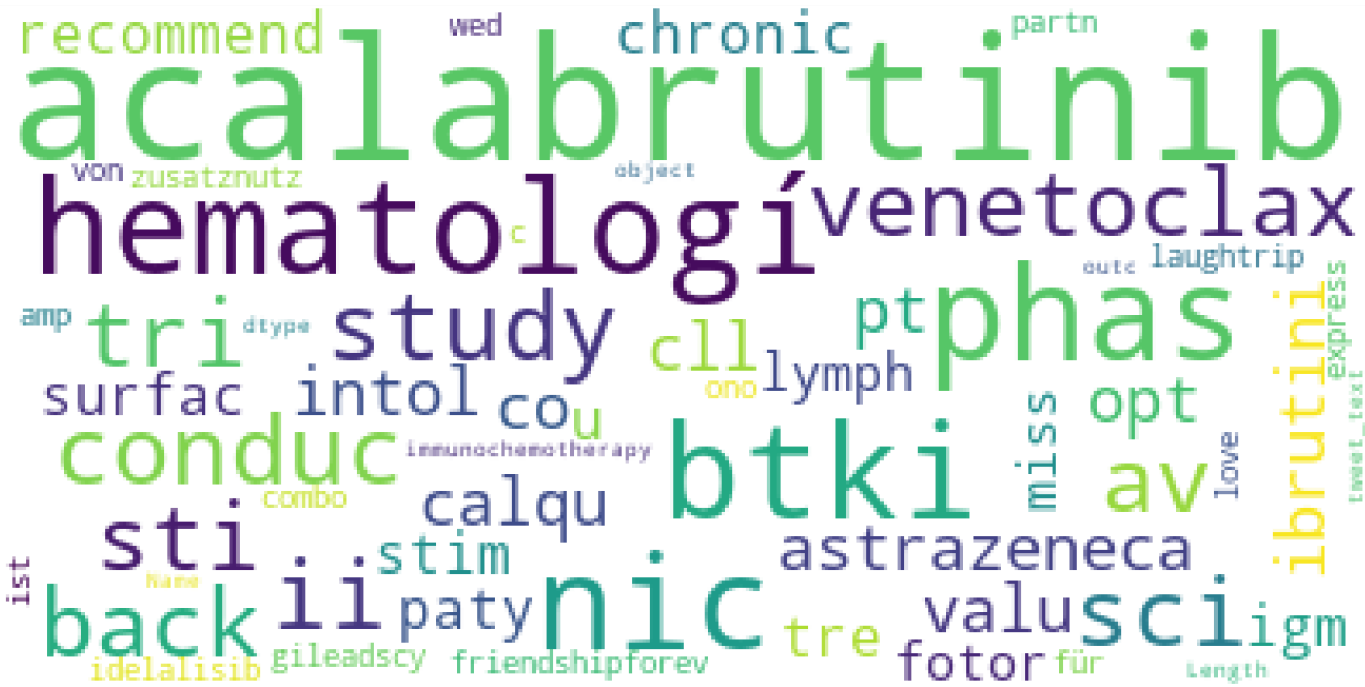
For Positive, Neutral, Negative Sentiments

In [59]:
```
def plot_wordcloud(text, mask=None, max_words=200, max_font_size=100, figure_size=(24.0,
                   title = None, title_size=40, image_color=False):

    wordcloud = WordCloud(background_color=color,
                          max_words = max_words,
                          max_font_size = max_font_size,
                          random_state = 42,
                          width=400,
                          height=200,
                          mask = mask)
```

```
    wordcloud.generate(str(text))

    plt.figure(figsize=figure_size)
    if image_color:
        image_colors = ImageColorGenerator(mask);
        plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation="bilinear")
        plt.title(title, fontdict={'size': title_size,
                                   'verticalalignment': 'bottom'})
    else:
        plt.imshow(wordcloud);
        plt.title(title, fontdict={'size': title_size, 'color': 'black',
                                   'verticalalignment': 'bottom'})
    plt.axis('off');
    plt.tight_layout()
```

In [60]:
```
plot_wordcloud(Neutral_sent.tweet_text,color='white',max_font_size=100,title_size=30,tit
```

WordCloud of Neutral Tweets

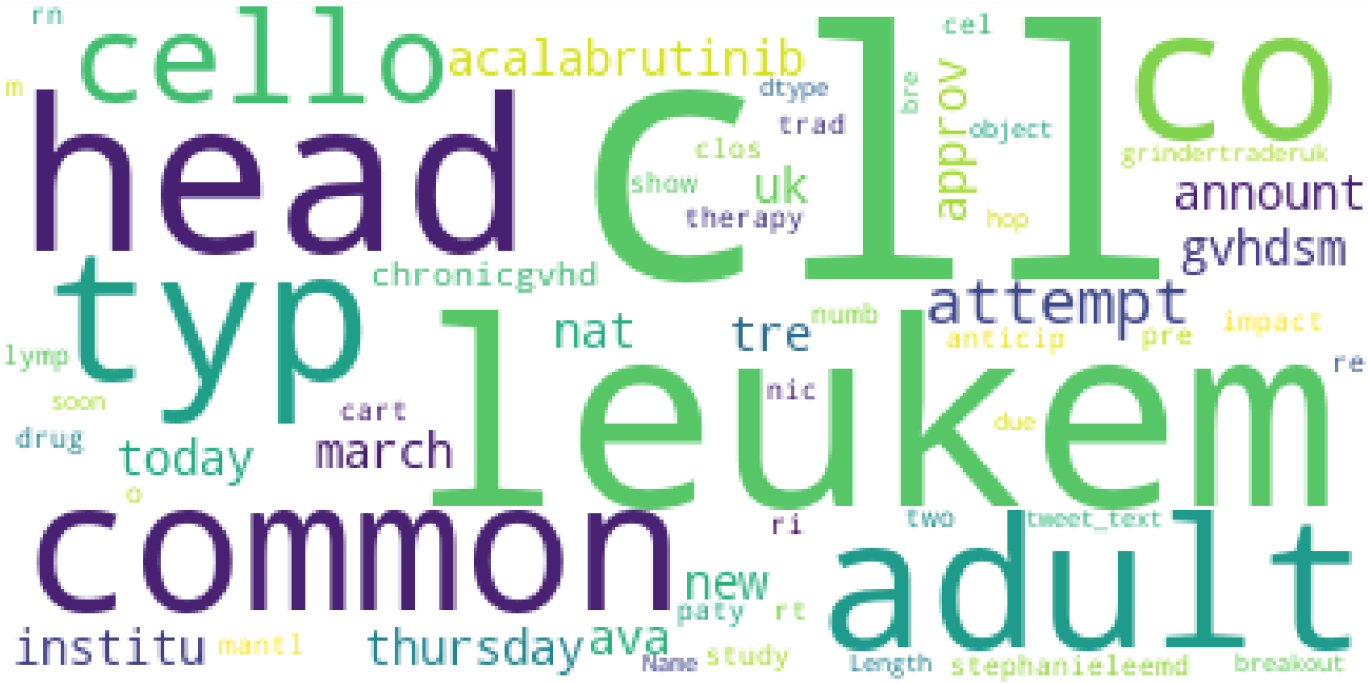


In [61]:
```
plot_wordcloud(Positive_sent.tweet_text,title="Word Cloud Of Positive tweets",title_size
```

Word Cloud Of Positive tweets

In [62]:
```
plot_wordcloud(Negative_sent.tweet_text,title="Word Cloud Of Negative tweets",title_size
```

Word Cloud Of Negative tweets



In [ ]:

In [ ]: