```
import numpy as np
import pandas as pd
```

```
from google.colab import drive

drive.mount("/content/gdrive")
```

```
    Mounted at /content/gdrive
```

```
df=pd.read_csv("/content/gdrive/MyDrive/Colab Notebooks/stress.csv")
df.head()
```

	subreddit	post_id	sentence_range	text	id	label	confidence	social_timestamp	soc
0	ptsd	8601tu	(15, 20)	He said he had not felt that way before, sugge...	33181	1	0.8	1521614353	
1	assistance	8lbrx9	(0, 5)	Hey there r/assistance, Not sure if this is th...	2606	0	1.0	1527009817	
2	ptsd	9ch1zh	(15, 20)	My mom then hit me with the newspaper and it s...	38816	1	0.8	1535935605	
3	relationships	7rorpp	[5, 10]	until i met my new boyfriend, he is amazing, h...	239	1	0.6	1516429555	
4	survivorsofabuse	9p2gbc	[0, 5]	October is Domestic Violence Awareness Month a...	1421	1	0.8	1539809005	

5 rows × 116 columns

```
df.describe()
```

	id	label	confidence	social_timestamp	social_karma	syntax_ari	lex_liwc_WC
count	2838.000000	2838.000000	2838.000000	2.838000e+03	2838.000000	2838.000000	2838.000000
mean	13751.999295	0.524313	0.808972	1.518107e+09	18.262156	4.684272	85.996124
std	17340.161897	0.499497	0.177038	1.552209e+07	79.419166	3.316435	32.334887
min	4.000000	0.000000	0.428571	1.483274e+09	0.000000	-6.620000	5.000000
25%	926.250000	0.000000	0.600000	1.509698e+09	2.000000	2.464243	65.000000
50%	1891.500000	1.000000	0.800000	1.517066e+09	5.000000	4.321886	81.000000
75%	25473.750000	1.000000	1.000000	1.530898e+09	10.000000	6.505657	101.000000
max	55757.000000	1.000000	1.000000	1.542592e+09	1435.000000	24.074231	310.000000

8 rows × 112 columns

```
df.isnull().sum()
```

```
    subreddit                   0
    post_id                     0
    sentence_range              0
    text                        0
    id                          0
                               ..
    lex_dal_avg_pleasantness    0
    social_upvote_ratio         0
```

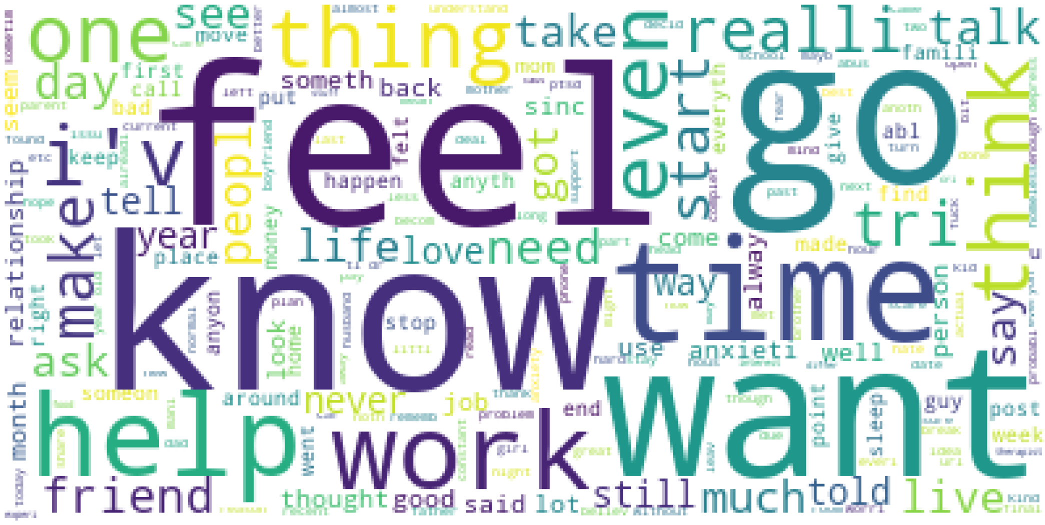
```
social_num_comments         0
syntax_fk_grade             0
sentiment                   0
Length: 116, dtype: int64
```

```
import nltk
import re
from nltk. corpus import stopwords
import string
nltk. download( 'stopwords' )
stemmer = nltk. SnowballStemmer("english")
stopword=set (stopwords . words ( 'english' ))
```

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

```
def clean(text):
    text = str(text) . lower()  #returns a string where all characters are lower case. Symbols and Numbers are ignored.
    text = re. sub('\[.*?\]',' ',text)  #substring and returns a string with replaced values.
    text = re. sub('https?://\S+/www\. \S+', ' ', text)#whitespace char with pattern
    text = re. sub('<. *?>+', ' ', text)#special char enclosed in square brackets
    text = re. sub(' [%s]' % re. escape(string. punctuation), ' ', text)#eliminate punctuation from string
    text = re. sub(' \n',' ', text)
    text = re. sub(' \w*\d\w*' ,' ', text)#word character ASCII punctuation
    text = [word for word in text. split(' ') if word not in stopword]  #removing stopwords
    text =" ". join(text)
    text = [stemmer . stem(word) for word in text. split(' ') ]#remove morphological affixes from words
    text = " ". join(text)
    return text
df [ "text"] = df["text"]. apply(clean)
```

```
import matplotlib. pyplot as plt
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
text = " ". join(i for i in df. text)
stopwords = set (STOPWORDS)
wordcloud = WordCloud( stopwords=stopwords,background_color="white") . generate(text)
plt. figure(figsize=(50, 50) )
plt. imshow(wordcloud )
plt. axis("off")
plt. show( )
```



```
from sklearn. feature_extraction. text import CountVectorizer
from sklearn. model_selection import train_test_split
```

```
x = np.array (df["text"])
y = np.array (df["label"])
```

```
cv = CountVectorizer ()
X = cv. fit_transform(x)
print(X)
xtrain, xtest, ytrain, ytest = train_test_split(X, y,test_size=0.33)
```

```
  (0, 7400)	1
  (0, 3274)	1
  (0, 9448)	1
  (0, 857)	1
  (0, 8354)	1
  (0, 3746)	1
  (0, 7209)	1
  (0, 8903)	1
  (0, 297)	1
  (0, 9743)	1
  (0, 4299)	1
  (0, 5030)	1
  (0, 5321)	1
  (0, 2184)	1
  (0, 5114)	1
  (0, 3261)	1
  (0, 2589)	3
  (0, 4184)	1
  (0, 5312)	1
  (0, 3693)	1
  (0, 8334)	1
  (0, 6856)	1
  (0, 4146)	1
  (0, 5170)	1
  (0, 1827)	1
  :	:
  (2836, 873)	1
  (2836, 4551)	1
  (2836, 2924)	1
  (2836, 4611)	1
  (2836, 4781)	1
  (2836, 4507)	1
  (2837, 7400)	2
  (2837, 3014)	1
  (2837, 5529)	2
  (2837, 8779)	1
  (2837, 8497)	1
  (2837, 6765)	1
  (2837, 4314)	1
  (2837, 9664)	1
  (2837, 5565)	1
  (2837, 8876)	1
  (2837, 5709)	1
  (2837, 2584)	1
  (2837, 7463)	1
  (2837, 2347)	1
  (2837, 7799)	1
  (2837, 2754)	1
  (2837, 8875)	1
  (2837, 5455)	1
  (2837, 3016)	1
```

```
from sklearn.naive_bayes import BernoulliNB
model=BernoulliNB()
model.fit(xtrain,ytrain)
```

```
BernoulliNB()
```

```
user=input("Enter the text")
data=cv.transform([user]).toarray()
output=model.predict(data)
print(output)
```

```
Enter the textsometimes i feel i need some hellp
[1]
```

✓ 5s completed at 7:19 PM ● ×