

In [1]:

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 %matplotlib inline
5 import seaborn as sns
6 from IPython import get_ipython
7 import warnings
8 warnings.filterwarnings("ignore")
```

In [4]:

```
1 data = pd.read_csv('stress.txt')
```

In [5]:

```
1 data.head()
```

Out[5]:

	subreddit	post_id	sentence_range	text	id	label	confidence	social_time
0	ptsd	8601tu	(15, 20)	He said he had not felt that way before, sugge...	33181	1	0.8	15216
1	assistance	8lbrx9	(0, 5)	Hey there r/assistance, Not sure if this is th...	2606	0	1.0	15270
2	ptsd	9ch1zh	(15, 20)	My mom then hit me with the newspaper and it s...	38816	1	0.8	15359
3	relationships	7rorpp	[5, 10]	until i met my new boyfriend, he is amazing, h...	239	1	0.6	15164
4	survivorsofabuse	9p2gbc	[0, 5]	October is Domestic Violence Awareness Month a...	1421	1	0.8	15398

5 rows × 116 columns



In [6]:

```
1 data.tail()
```

Out[6]:

	subreddit	post_id	sentence_range	text	id	label	confidence	social_time
2833	relationships	7oe1t	[35, 40]	* Her, a week ago: Precious, how are you? (I...	1713	0	1.000000	15151
2834	ptsd	9p4ung	[20, 25]	I don't have the ability to cope with it anymo...	1133	1	1.000000	15398
2835	anxiety	9nam6l	(5, 10)	In case this is the first time you're reading ...	10442	0	1.000000	15392
2836	almosthomeless	5y53ya	[5, 10]	Do you find this normal? They have a good rela...	1834	0	0.571429	14889
2837	ptsd	5y25cl	[0, 5]	I was talking to my mom this morning and she s...	961	1	0.571429	14889

5 rows × 116 columns



In [7]:

```
1 data.shape
```

Out[7]:

(2838, 116)

In [8]:



```
1 data.columns
```

Out[8]:

```
Index(['subreddit', 'post_id', 'sentence_range', 'text', 'id', 'label',  
      'confidence', 'social_timestamp', 'social_karma', 'syntax_ari',  
      ...  
      'lex_dal_min_pleasantness', 'lex_dal_min_activation',  
      'lex_dal_min_imagery', 'lex_dal_avg_activation', 'lex_dal_avg_image  
ry',  
      'lex_dal_avg_pleasantness', 'social_upvote_ratio',  
      'social_num_comments', 'syntax_fk_grade', 'sentiment'],  
      dtype='object', length=116)
```

In [9]:



```
1 data.duplicated().sum()
```

Out[9]:

```
0
```

In [10]:



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

Out[10]:

```
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
```

In [11]:



```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 2838 entries, 0 to 2837  
Columns: 116 entries, subreddit to sentiment  
dtypes: float64(106), int64(6), object(4)  
memory usage: 2.5+ MB
```

In [12]:



```
1 data.describe()
```

Out[12]:

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

8 rows × 112 columns

In [13]:



```
1 data['subreddit'].unique()
```

Out[13]:

```
array(['ptsd', 'assistance', 'relationships', 'survivorsofabuse',
      'domesticviolence', 'anxiety', 'homeless', 'stress',
      'almosthomeless', 'food_pantry'], dtype=object)
```

In [14]:



```
1 data['subreddit'].value_counts()
```

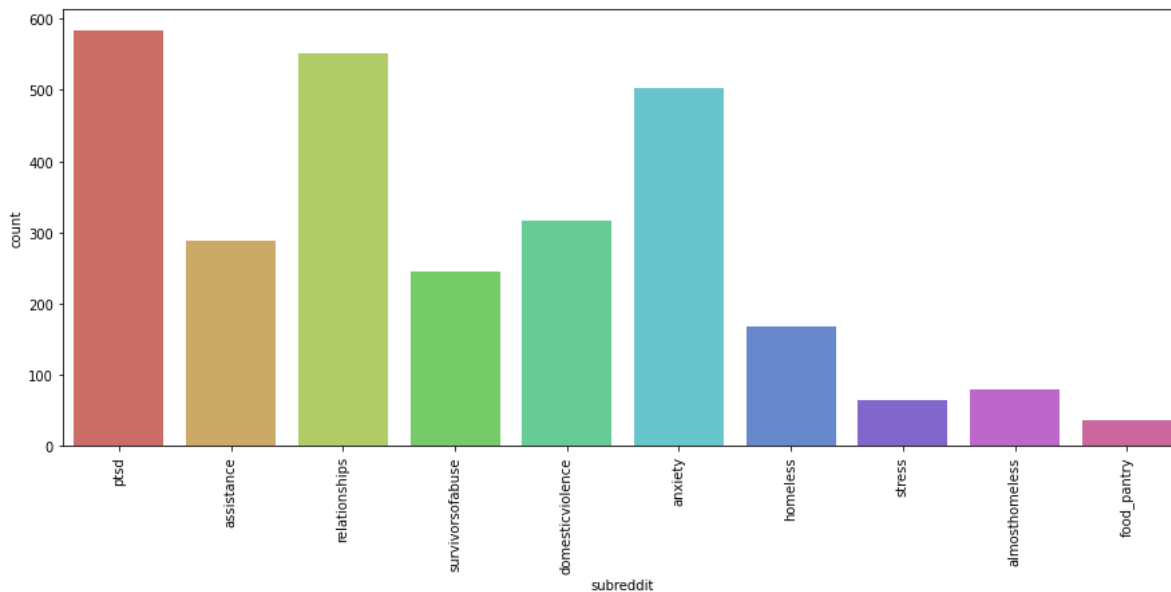
Out[14]:

```
ptsd          584
relationships  552
anxiety       503
domesticviolence  316
assistance    289
survivorsofabuse  245
homeless      168
almosthomeless  80
stress        64
food_pantry   37
Name: subreddit, dtype: int64
```

In [16]:



```
1 plt.figure(figsize=(15,6))
2 sns.countplot('subreddit', data = data, palette = 'hls')
3 plt.xticks(rotation = 90)
4 plt.show()
```



In [17]:



```
1 data['label'].unique()
```

Out[17]:

```
array([1, 0], dtype=int64)
```

In [18]:



```
1 data['label'].value_counts()
```

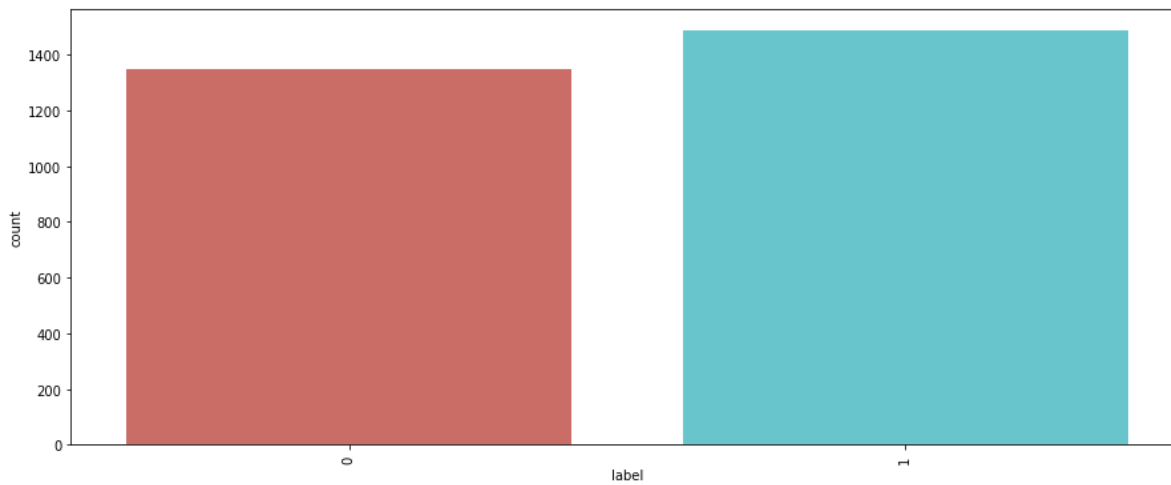
Out[18]:

```
1    1488
0    1350
Name: label, dtype: int64
```

In [19]:



```
1 plt.figure(figsize=(15,6))
2 sns.countplot('label', data = data, palette = 'hls')
3 plt.xticks(rotation = 90)
4 plt.show()
```

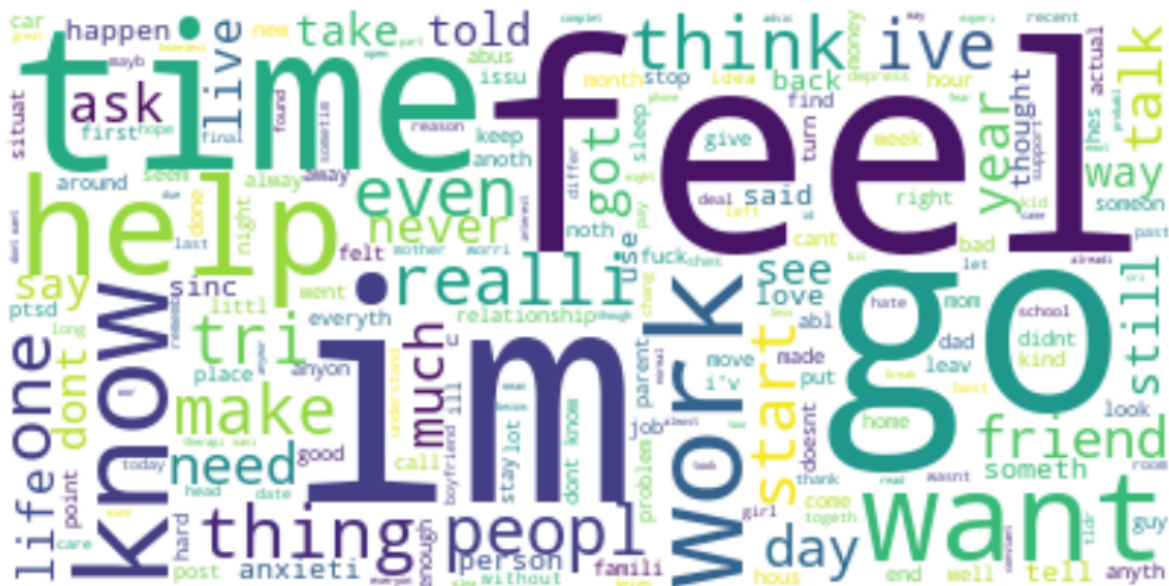


In [20]:



```
1 import nltk
2 import re
3 nltk.download('stopwords')
4 stemmer = nltk.SnowballStemmer("english")
5 from nltk.corpus import stopwords
6 import string
7 stopword=set(stopwords.words('english'))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\pc\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
```



In [23]:



```
1 data["label"] = data["label"].map({0: "No Stress", 1: "Stress"})
2 data = data[["text", "label"]]
3 print(data.head())
```

	text	label
0	said felt way sugget go rest trigger ahead you...	Stress
1	hey rassist sure right place post goe im curr...	No Stress
2	mom hit newspaper shock would know dont like pla...	Stress
3	met new boyfriend amaz kind sweet good student...	Stress
4	octob domest violenc awar month domest violenc...	Stress

In [24]:



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

In [25]:



```
1 x = np.array(data["text"])
2 y = np.array(data["label"])
```

In [26]:



```
1 cv = CountVectorizer()
2 X = cv.fit_transform(x)
3 xtrain, xtest, ytrain, ytest = train_test_split(X, y,
4                                                 test_size=0.33,
5                                                 random_state=42)
```

In [27]:



```
1 from sklearn.linear_model import LogisticRegression
2 from sklearn.tree import DecisionTreeClassifier
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.neighbors import KNeighborsClassifier
5 from sklearn.naive_bayes import BernoulliNB
```

In [28]:



```
1 model_log = LogisticRegression()
2 model_log.fit(xtrain, ytrain)
```

Out[28]:

LogisticRegression()



In [29]:

```
1 print("Score of the model with X-train and Y-train is : ", str(round(model_log.score(xtrain, ytrain), 2)))
2 print("Score of the model with X-test and Y-test is : ", str(round(model_log.score(xtest, ytest), 2)))
```

Score of the model with X-train and Y-train is : 99.63 %  
Score of the model with X-test and Y-test is : 71.5 %

In [30]:

```
1 model_dt = DecisionTreeClassifier()
2 model_dt.fit(xtrain, ytrain)
```

Out[30]:

DecisionTreeClassifier()

In [31]:

```
1 print("Score of the model with X-train and Y-train is : ", str(round(model_dt.score(xtrain, ytrain), 2)))
2 print("Score of the model with X-test and Y-test is : ", str(round(model_dt.score(xtest, ytest), 2)))
```

Score of the model with X-train and Y-train is : 100.0 %  
Score of the model with X-test and Y-test is : 60.83 %

In [32]:

```
1 model_rf = RandomForestClassifier(n_estimators=10,
2                                 criterion="entropy")
3 model_rf.fit(xtrain, ytrain)
```

Out[32]:

RandomForestClassifier(criterion='entropy', n\_estimators=10)

In [33]:

```
1 print("Score of the model with X-train and Y-train is : ", str(round(model_rf.score(xtrain, ytrain), 2)))
2 print("Score of the model with X-test and Y-test is : ", str(round(model_rf.score(xtest, ytest), 2)))
```

Score of the model with X-train and Y-train is : 99.21 %  
Score of the model with X-test and Y-test is : 66.28 %

In [34]:

```
1 model = BernoulliNB()
2 model.fit(xtrain, ytrain)
```

Out[34]:

BernoulliNB()

In [35]:



```
1 print("Score of the model with X-train and Y-train is : ", str(round(model.score(xt
2 print("Score of the model with X-test and Y-test is : ", str(round(model.score(xtes
```

Score of the model with X-train and Y-train is : 91.95 %  
Score of the model with X-test and Y-test is : 74.71 %

In [39]:



```
1 user = input("Enter a Text: ")
2 data = cv.transform([user]).toarray()
3 output = model_dt.predict(data)
4 print(output)
```

Enter a Text: i am mentally strong.  
['No Stress']

In [41]:



```
1 user = input("Enter a Text: ")
2 data = cv.transform([user]).toarray()
3 output = model_dt.predict(data)
4 print(output)
```

Enter a Text: i am in stress.  
['Stress']