```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

```
df = pd.read_csv('cyberbullying.csv.zip')
```

```
df.head()
```

	tweet_text	cyberbullying_type
0	In other words #katandandre, your food was cra...	not_cyberbullying
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying
2	@XochitlSuckkks a classy whore? Or more red ve...	not_cyberbullying
3	@Jason_Gio meh. :P thanks for the heads up, b...	not_cyberbullying
4	@RudhoeEnglish This is an ISIS account pretend...	not_cyberbullying

+ Code + Text

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47692 entries, 0 to 47691
Data columns (total 2 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   tweet_text          47692 non-null  object
 1   cyberbullying_type  47692 non-null  object
dtypes: object(2)
memory usage: 745.3+ KB
```

```
!pip install demoji
```

```
Collecting demoji
  Downloading demoji-1.1.0-py3-none-any.whl (42 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.9/42.9 kB 1.2 MB/s eta 0:00:00
Installing collected packages: demoji
Successfully installed demoji-1.1.0
```

```
import re
from nltk.corpus import stopwords
from nltk.stem.snowball import SnowballStemmer
import demoji
import string
```

```
import warnings
warnings.filterwarnings("ignore")
from warnings import simplefilter
from sklearn.exceptions import ConvergenceWarning
simplefilter("ignore", category=ConvergenceWarning)
```

```
import nltk
nltk.download('stopwords')
STOPWORDS = set(stopwords.words('english'))
STOPWORDS.update(['rt', 'mkr', 'didn', 'bc', 'n', 'm',
                  'im', 'll', 'y', 've', 'u', 'ur', 'don',
                  'p', 't', 's', 'aren', 'kp', 'o', 'kat',
                  'de', 're', 'amp', 'will', 'wa', 'e', 'like'])
stemmer = SnowballStemmer('english')
def clean_text(text):
    pattern = re.compile(r"(#[A-Za-z0-9]+|@[A-Za-z0-9]+|https?://\S+|www\.\S+|\S+\.[a-z]+|RT @)")
    text = pattern.sub('', text)
    text = " ".join(text.split())
    text = text.lower()
    text = " ".join([stemmer.stem(word) for word in text.split()])
    remove_punc = re.compile(r"[%s]" % re.escape(string.punctuation))
    text = remove_punc.sub('', text)
    text = " ".join([word for word in str(text).split() if word not in STOPWORDS])
    emoji = demoji.findall(text)
    for emot in emoji:
        text = re.sub(r"(%s)" % (emot), "_".join(emoji[emot].split()), text)

    return text
```

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

```
df['cleaned_text'] = df['tweet_text'].apply(lambda text: clean_text(text))
```

```
df.head()
```

	tweet_text	cyberbullying_type	cleaned_text
0	In other words #katandandre, your food was cra...	not_cyberbullying	word food crapilicious
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	whi white
2	@XochitlSuckkks a classy whore? Or more red ve...	not_cyberbullying	classi whore red velvet cupcakes
3	@Jason_Gio meh. :P thanks for the heads up, b...	not_cyberbullying	gio meh thank head concern anoth angri dude tw...
4	@RudhoeEnglish This is an ISIS account pretend...	not_cyberbullying	isi account pretend kurdish account islam lies

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

```
tweet_text            0
cyberbullying_type    0
cleaned_text          0
dtype: int64
```

```
df['cleaned_text'].duplicated().sum()
```

```
2887
```

```
df.drop_duplicates("cleaned_text", inplace = True)
```

```
df['cleaned_text'].str.isspace().sum()
```

```
0
```

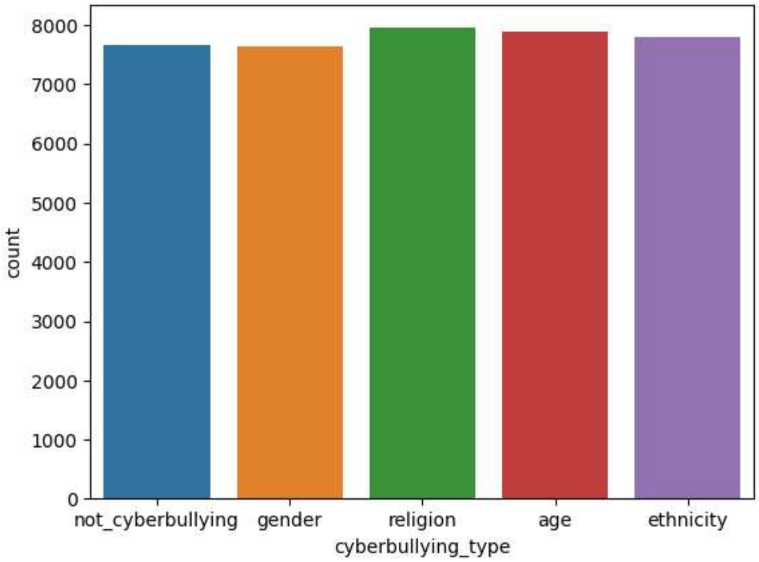
```
df = df[df["cyberbullying_type"]!="other_cyberbullying"]
```

```
df['cyberbullying_type'].value_counts()
```

```
religion             7946
age                  7887
ethnicity            7797
not_cyberbullying    7670
gender               7637
Name: cyberbullying_type, dtype: int64
```

```
sns.countplot(data = df, x = 'cyberbullying_type')
```

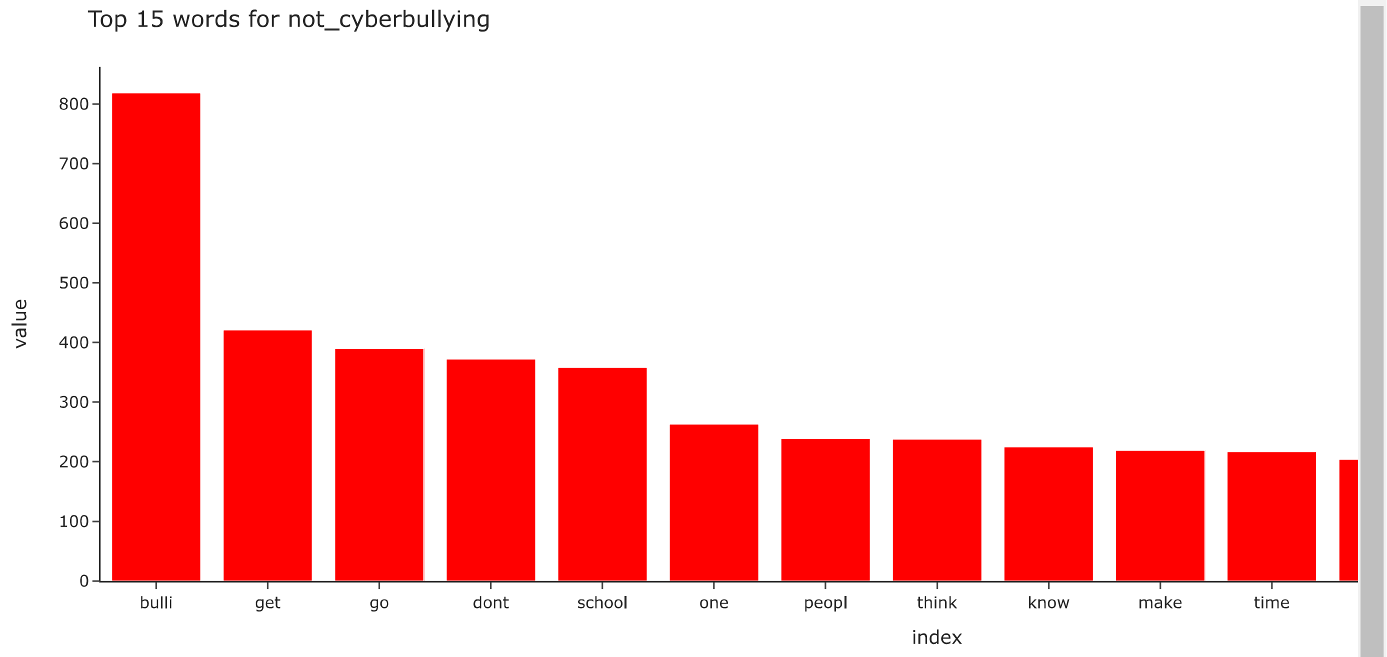
```
<Axes: xlabel='cyberbullying_type', ylabel='count'>
```

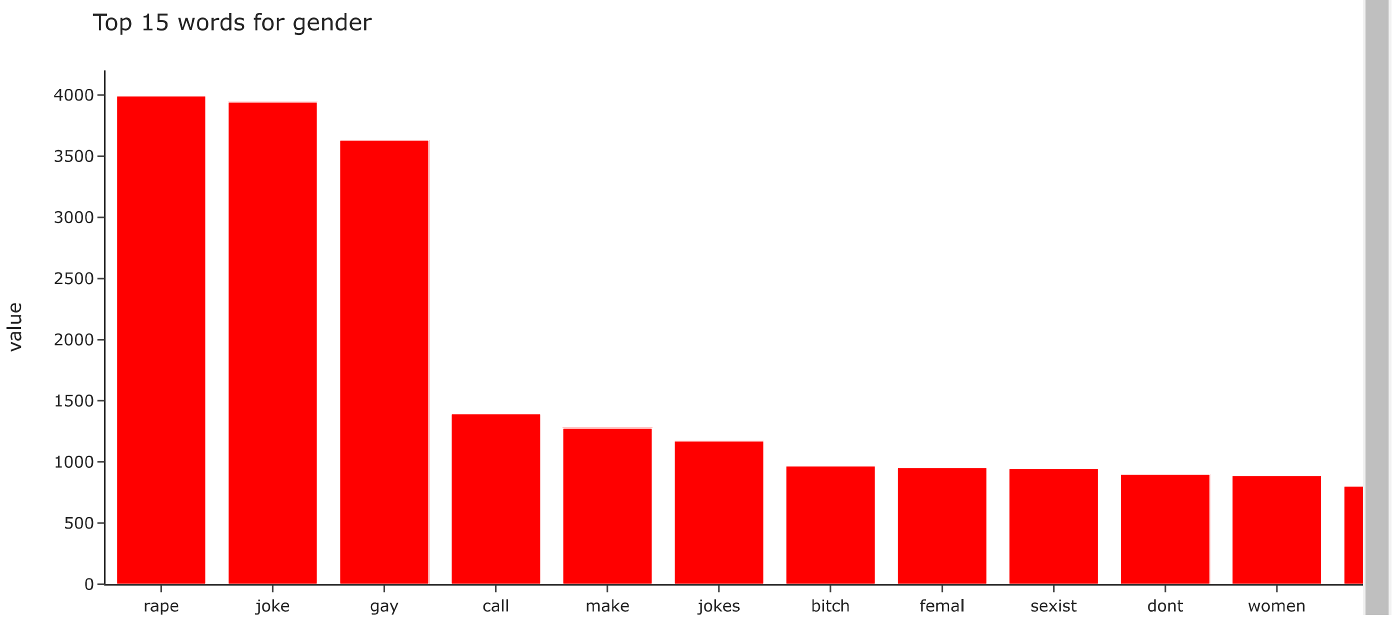


```
for cyber_type in df.cyberbullying_type.unique():

    top50_word = df.cleaned_text[df.cyberbullying_type==cyber_type].str.split(expand=True).stack().value_counts()[:15]

    fig = px.bar(top50_word, color=top50_word.values, color_continuous_scale=px.colors.sequential.RdPu, custom_data=[top50_word.values])
    fig.update_traces(marker_color='red')
    fig.update_traces(hovertemplate='<b>Count: </b>%{customdata[0]}')
    fig.update_layout(title=f"Top 15 words for {cyber_type}",
                     template='simple_white',
                     hovermode='x unified')
    fig.show()
```





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

```
X = df['cleaned_text']
y = df['cyberbullying_type']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.1, random_state = 42)
```

```
tfidf = TfidfVectorizer(max_features = 5000)
```

```
X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)
```

```
X_train_tfidf
```

```
<35043x5000 sparse matrix of type '<class 'numpy.float64'>'
        with 403374 stored elements in Compressed Sparse Row format>
```

```
X_test_tfidf
```

```
<3894x5000 sparse matrix of type '<class 'numpy.float64'>'
	with 44224 stored elements in Compressed Sparse Row format>
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
tfidf_array_train = X_train_tfidf.toarray()
tfidf_array_test = X_test_tfidf.toarray()
scaled_X_train = scaler.fit_transform(tfidf_array_train)
scaled_X_test = scaler.transform(tfidf_array_test)
```

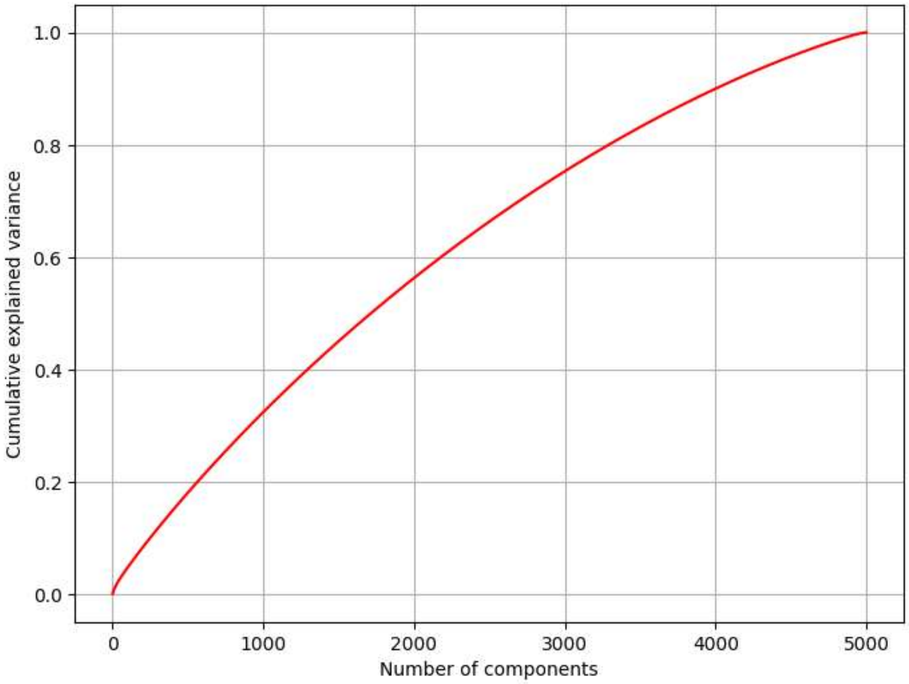
8000

```
from sklearn.decomposition import PCA
NUM_COMPONENTS = 5000
pca = PCA(NUM_COMPONENTS)
reduced = pca.fit(scaled_X_train)
```

```
variance_explained = np.cumsum(pca.explained_variance_ratio_)
```

```
fig, ax = plt.subplots(figsize=(8, 6))
plt.plot(range(NUM_COMPONENTS),variance_explained, color='r')
ax.grid(True)
plt.xlabel("Number of components")
plt.ylabel("Cumulative explained variance")
```

```
Text(0, 0.5, 'Cumulative explained variance')
```



```
final_pca = PCA(0.9)
reduced_90 = final_pca.fit_transform(scaled_X_train)
```

index

```
reduced_90_test = final_pca.transform(scaled_X_test)
```

```
reduced_90.shape
```

```
(35043, 3999)
```

```
final_pca = PCA(0.8)
reduced_80 = final_pca.fit_transform(scaled_X_train)
```

```
reduced_80.shape
```

```
(35043, 3290)
```

```
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.linear_model import LogisticRegression
log_model_pca = LogisticRegression()
log_model_pca.fit(reduced_90, y_train)
preds_log_model_pca = log_model_pca.predict(reduced_90_test)
print(classification_report(y_test, preds_log_model_pca))
confusion_matrix(y_test, preds_log_model_pca)
```

```
                   precision    recall  f1-score   support

              age       0.86      0.83      0.85       766
        ethnicity       0.90      0.85      0.87       801
           gender       0.77      0.79      0.78       788
not_cyberbullying       0.65      0.67      0.66       783
         religion       0.84      0.86      0.85       756

         accuracy                           0.80      3894
        macro avg       0.80      0.80      0.80      3894
     weighted avg       0.80      0.80      0.80      3894

array([[638,  10,  28,  76,  14],
       [ 19, 682,  24,  50,  26],
       [ 20,  19, 622, 100,  27],
       [ 47,  30, 119, 528,  59],
       [ 14,  18,  14,  60, 650]])
```

```
from sklearn.experimental import enable_halving_search_cv
from sklearn.model_selection import HalvingGridSearchCV
log_model = LogisticRegression(solver = 'saga')
param_grid = {'C': np.logspace(0, 10, 5)}
grid_log_model = HalvingGridSearchCV(log_model, param_grid = param_grid, n_jobs = -1, min_resources = 'exhaust', factor = 3)
grid_log_model.fit(X_train_tfidf, y_train)
preds_grid_log_model = grid_log_model.predict(X_test_tfidf)
print(classification_report(y_test, preds_grid_log_model))
confusion_matrix(y_test, preds_grid_log_model)
```

```
                   precision    recall  f1-score   support

              age       0.96      0.97      0.96       766
        ethnicity       0.98      0.98      0.98       801
           gender       0.92      0.84      0.88       788
not_cyberbullying       0.80      0.85      0.82       783
         religion       0.94      0.96      0.95       756

         accuracy                           0.92      3894
        macro avg       0.92      0.92      0.92      3894
     weighted avg       0.92      0.92      0.92      3894

array([[743,   1,   3,  18,   1],
       [  2, 782,   2,  13,   2],
       [  1,   6, 660, 113,   8],
       [ 28,   7,  49, 666,  33],
       [  1,   2,   2,  25, 726]])
```

```
grid_log_model.best_estimator_
```

```
▾      LogisticRegression
LogisticRegression(solver='saga')
```

```
from sklearn.svm import LinearSVC
svm_model = LinearSVC()
C = [1e-5, 1e-4, 1e-2, 1e-1, 1]
param_grid = {'C': C}
grid_svm_model = HalvingGridSearchCV(svm_model, param_grid = param_grid, n_jobs = -1, min_resources = 'exhaust', factor = 3)
grid_svm_model.fit(X_train_tfidf, y_train)
preds_grid_svm_model = grid_svm_model.predict(X_test_tfidf)
print(classification_report(y_test, preds_grid_svm_model))
confusion_matrix(y_test, preds_grid_svm_model)
```

```
                   precision    recall  f1-score   support

              age       0.94      0.98      0.96       766
        ethnicity       0.97      0.98      0.98       801
           gender       0.94      0.81      0.87       788
not_cyberbullying       0.79      0.85      0.82       783
         religion       0.95      0.96      0.96       756

         accuracy                           0.92      3894
```

```
       macro avg       0.92      0.92      0.92      3894
    weighted avg       0.92      0.92      0.92      3894
```

```
array([[754,   1,   1,  10,   0],
       [  2, 783,   2,  13,   1],
       [  3,   7, 637, 133,   8],
       [ 39,  11,  36, 665,  32],
       [  1,   2,   1,  23, 729]])
```

```
grid_svm_model.best_estimator_
```

```
▾   LinearSVC
LinearSVC(C=0.1)
```