```
from google.colab import drive
drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
%cd /content/drive/MyDrive/DSC680/Weeks1-4/Week2/data/

    /content/drive/MyDrive/DSC680/Weeks1-4/Week2/data
```

```
%ls

    aggression_parsed_dataset.csv/          glove.6B.50d.txt
    antidepressant/                         glove.6B.zip
    antidepressants/                        hopeless/
    archive/                                kaggle_parsed_dataset.csv/
    attack_parsed_dataset.csv/              lonely/
    cyberbullying_tweets.csv                suicide/
    depressed/                              toxicity_parsed_dataset.csv/
    depression/                             twitter_parsed_dataset.csv/
    depression_dataset_reddit_cleaned.csv/  twitter_racism_parsed_dataset.csv/
    depressive_unigram_tweets_final.csv     twitter_sexism_parsed_dataset.csv/
    glove.6B.100d.txt                       vader_processed_final.csv
    glove.6B.200d.txt                       youtube_parsed_dataset.csv/
    glove.6B.300d.txt
```

```
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
    Collecting contractions
      Downloading contractions-0.1.73-py2.py3-none-any.whl (8.7 kB)
    Collecting textsearch>=0.0.21
      Downloading textsearch-0.0.24-py2.py3-none-any.whl (7.6 kB)
    Collecting pyahocorasick
      Downloading pyahocorasick-2.0.0-cp39-cp39-manylinux_2_5_x86_64.manylinux1_x86_64.whl (103 kB)
         ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 103.2/103.2 KB 6.1 MB/s eta 0:00:00
    Collecting anyascii
      Downloading anyascii-0.3.2-py3-none-any.whl (289 kB)
         ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 289.9/289.9 KB 23.4 MB/s eta 0:00:00
    Installing collected packages: pyahocorasick, anyascii, textsearch, contractions
    Successfully installed anyascii-0.3.2 contractions-0.1.73 pyahocorasick-2.0.0 textsearch-0.0.24
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
    Collecting autocorrect
      Downloading autocorrect-2.6.1.tar.gz (622 kB)
         ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 622.8/622.8 KB 12.5 MB/s eta 0:00:00
      Preparing metadata (setup.py) ... done
    Building wheels for collected packages: autocorrect
      Building wheel for autocorrect (setup.py) ... done
      Created wheel for autocorrect: filename=autocorrect-2.6.1-py3-none-any.whl size=622380 sha256=0e9ee06b1ca2485c1fca5dc6a48a0f938a84c387
      Stored in directory: /root/.cache/pip/wheels/ab/0f/23/3c010c3fd877b962146e7765f9e9b08026cac8b035094c5750
    Successfully built autocorrect
    Installing collected packages: autocorrect
    Successfully installed autocorrect-2.6.1
```

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
import emoji
import string
import nltk
from PIL import Image
from collections import Counter
from wordcloud import WordCloud, ImageColorGenerator, STOPWORDS
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk.stem.snowball import SnowballStemmer
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classification_report
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.svm import SVC,LinearSVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.pipeline import Pipeline
import pickle
```

```
data = pd.read_csv('cyberbullying_tweets.csv')
data.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
```

```
data.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

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

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

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

```
religion               7998
age                    7992
gender                 7973
ethnicity              7961
not_cyberbullying      7945
other_cyberbullying    7823
Name: cyberbullying_type, dtype: int64
```

```
data = data.rename(columns={'tweet_text': 'text', 'cyberbullying_type': 'sentiment'})
```

```
data.head()
```

	text	sentiment
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

```
data["sentiment_encoded"] = data['sentiment'].replace({"religion": 1, "age": 2, "ethnicity": 3, "gender": 4, "other_cyberbullying": 5,"not_cy
data.head()
```

	text	sentiment	sentiment_encoded
0	In other words #katandandre, your food was cra...	not_cyberbullying	6
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	6
2	@XochitlSuckkks a classy whore? Or more red ve...	not_cyberbullying	6
3	@Jason_Gio meh. :P thanks for the heads up, b...	not_cyberbullying	6
4	@RudhoeEnglish This is an ISIS account pretend...	not_cyberbullying	6

```
stop_words = set(stopwords.words('english'))
```

```
def strip_emoji(text):
    return emoji.replace_emoji(text,replace="")
```

```
def strip_all_entities(text):
    text = text.replace('\r', '').replace('\n', ' ').lower()
    text = re.sub(r"(?:\@|https?\://)\S+", "", text)
    text = re.sub(r'[^\x00-\x7f]',r'', text)
    text = re.sub(r'(.)1+', r'1', text)
    text = re.sub('[0-9]+', '', text)
    stopchars= string.punctuation
    table = str.maketrans('', '', stopchars)
    text = text.translate(table)
    text = [word for word in text.split() if word not in stop_words]
    text = ' '.join(text)
    return text
```

```
def decontract(text):
    text = re.sub(r"can\'t", "can not", text)
    text = re.sub(r"n\'t", " not", text)
    text = re.sub(r"\'re", " are", text)
    text = re.sub(r"\'s", " is", text)
    text = re.sub(r"\'d", " would", text)
    text = re.sub(r"\'ll", " will", text)
    text = re.sub(r"\'t", " not", text)
    text = re.sub(r"\'ve", " have", text)
    text = re.sub(r"\'m", " am", text)
    return text
```

```
def clean_hashtags(tweet):
    new_tweet = " ".join(word.strip() for word in re.split('#(?!(?:hashtag)\b)[\w-]+(?=(?:\s+#[\w-]+)*\s*$)', tweet))
    new_tweet2 = " ".join(word.strip() for word in re.split('#|_', new_tweet))
    return new_tweet2
```

```
def filter_chars(a):
    sent = []
    for word in a.split(' '):
        if ('$' in word) | ('&' in word):
            sent.append('')
        else:
            sent.append(word)
    return ' '.join(sent)
```

```
def remove_mult_spaces(text):
    return re.sub("\s\s+" , " ", text)
```

```
def stemmer(text):
    tokenized = nltk.word_tokenize(text)
    ps = PorterStemmer()
    return ' '.join([ps.stem(words) for words in tokenized])
```

```
def lemmatize(text):
    tokenized = nltk.word_tokenize(text)
    lm = WordNetLemmatizer()
    return ' '.join([lm.lemmatize(words) for words in tokenized])
```

```
def preprocess(text):
    text = strip_emoji(text)
    text = decontract(text)
    text = strip_all_entities(text)
    text = clean_hashtags(text)
    text = filter_chars(text)
    text = remove_mult_spaces(text)
    text = stemmer(text)
    text = lemmatize(text)
    return text
```

```
data['cleaned_text'] = data['text'].apply(preprocess)
data.head()
```

	text	sentiment	sentiment_encoded	cleaned_text
0	In other words #katandandre, your food was cra...	not_cyberbullying	6	word katandandr food crapilici mkr
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	6	aussietv white mkr theblock imacelebrityau tod...
2	@XochitlSuckkks a classy whore? Or more red ve...	not_cyberbullying	6	classi whore red velvet cupcak

```
data["cleaned_text"].duplicated().sum()
```

```
3041
```

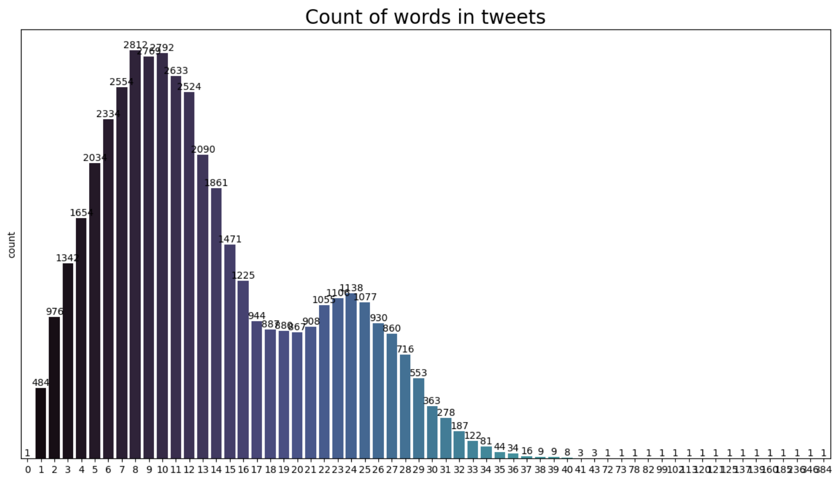
```
data.drop_duplicates("cleaned_text", inplace=True)
```

```
data['tweet_list'] = data['cleaned_text'].apply(word_tokenize)
data.head()
```

	text	sentiment	sentiment_encoded	cleaned_text	tweet_list
0	In other words #katandandre, your food was cra...	not_cyberbullying	6	word katandandr food crapilici mkr	[word, katandandr, food, crapilici, mkr]
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	6	aussietv white mkr theblock imacelebrityau tod...	[aussietv, white, mkr, theblock, imacelebritya...

```
text_len = []
for text in data.tweet_list:
    tweet_len = len(text)
    text_len.append(tweet_len)
data['text_len'] = text_len
```

```
plt.figure(figsize=(15,8))
ax = sns.countplot(x='text_len', data=data, palette='mako')
plt.title('Count of words in tweets', fontsize=20)
plt.yticks([])
ax.bar_label(ax.containers[0])
plt.ylabel('count')
plt.xlabel('')
plt.show()
```



```
data = data[data['text_len']!=0]
```

```
data.shape
```

```
     (44650, 6)
```

```
def plot_wordcloud(cyberbullying_type):
    string = ""
    for i in data[data.sentiment == cyberbullying_type].cleaned_text.values:
        string = string + " " + i.strip()

    custom_mask = np.array(Image.open('/kaggle/input/twitter-image/twitter.png'))
    mask_colors = ImageColorGenerator(custom_mask)
    wordcloud = WordCloud(background_color ='white',max_words=2000, max_font_size=256,
               random_state=42, width=custom_mask.shape[1],height=custom_mask.shape[0],
                mask = custom_mask,min_font_size = 10,color_func=mask_colors).generate(string)

    # plot the WordCloud image
    plt.figure(figsize = (8, 8), facecolor = None)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.tight_layout(pad = 0)
    plt.title(cyberbullying_type)
    plt.show()
    del string
```

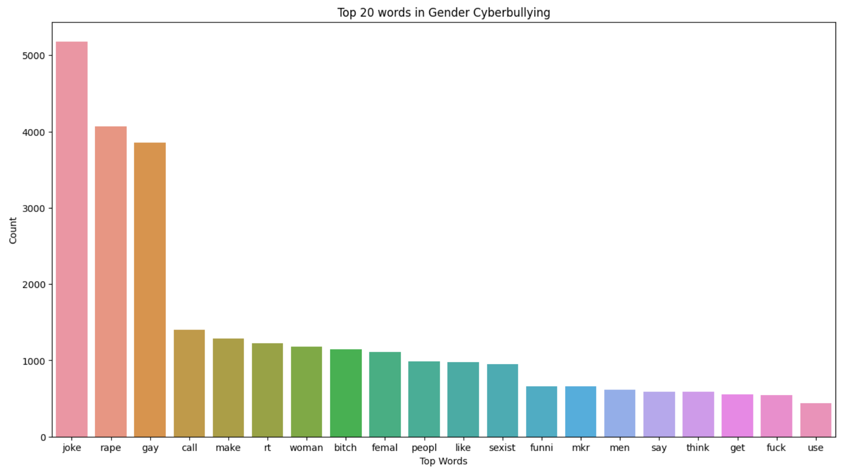
```
not_cyberbullying_type = data[data['sentiment']=='not_cyberbullying']
gender_type = data[data['sentiment']=='gender']
religion_type = data[data['sentiment']=='religion']
other_cyberbullying_type = data[data['sentiment']=='other_cyberbullying']
age_type = data[data['sentiment']=='age']
ethnicity_type = data[data['sentiment']=='ethnicity']
```

```
gender = Counter([item for sublist in gender_type['tweet_list'] for item in sublist])
top20_gender = pd.DataFrame(gender.most_common(20))
top20_gender.columns = ['Top Words','Count']
top20_gender.style.background_gradient(cmap='Greens')
```

	Top Words	Count
0	joke	5179
1	rape	4070
2	gay	3852
3	call	1401
4	make	1283
5	rt	1221
6	woman	1182
7	bitch	1146
8	femal	1108
9	peopl	988
10	like	978
11	sexist	953
12	funni	662
13	mkr	659

```
fig = plt.figure(figsize=(15,8))
sns.barplot(data=top20_gender, y="Count", x="Top Words")
plt.title("Top 20 words in Gender Cyberbullying")
```

```
Text(0.5, 1.0, 'Top 20 words in Gender Cyberbullying')
```

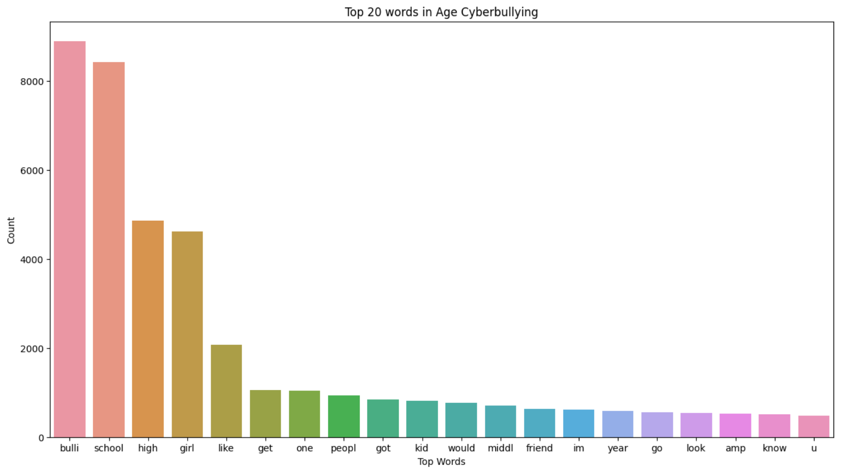


```
age = Counter([item for sublist in age_type['tweet_list'] for item in sublist])
top20_age = pd.DataFrame(age.most_common(20))
top20_age.columns = ['Top Words','Count']
top20_age.style.background_gradient(cmap='Greens')
```

	Top Words	Count
0	bulli	8899
1	school	8426
2	high	4861
3	girl	4616
4	like	2076
5	get	1051
6	one	1044
7	peopl	934
8	got	842
9	kid	809
10	would	762
11	middl	714
12	friend	632
13	im	625
14	year	586
15	go	555
16	look	545
17	amp	527

```
fig = plt.figure(figsize=(15,8))
sns.barplot(data=top20_age, y="Count", x="Top Words")
plt.title("Top 20 words in Age Cyberbullying")
```

```
Text(0.5, 1.0, 'Top 20 words in Age Cyberbullying')
```

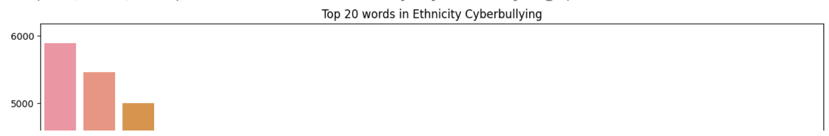


```
ethnicity = Counter([item for sublist in ethnicity_type['tweet_list'] for item in sublist])
ethnicity = Counter([item for sublist in ethnicity_type['tweet_list'] for item in sublist])
top20_ethnicity = pd.DataFrame(ethnicity.most_common(20))
top20_ethnicity.columns = ['Top Words','Count']
top20_ethnicity.style.background_gradient(cmap='Greens')
```

	Top Words	Count
0	fuck	5892
1	nigger	5456
2	dumb	5003
3	as	2241
4	black	2134
5	u	2107
6	white	1557
7	call	1381
8	peopl	1182
9	rt	1166
10	obama	1112
11	one	1058
12	like	1035
13	bitch	999
14	ur	987
15	say	817
16	first	745
17	get	740
18	anyth	708
19	racism	689

```
fig = plt.figure(figsize=(15,8))
sns.barplot(data=top20_ethnicity, y="Count", x="Top Words")
plt.title("Top 20 words in Ethnicity Cyberbullying")
```

```
Text(0.5, 1.0, 'Top 20 words in Ethnicity Cyberbullying')
```

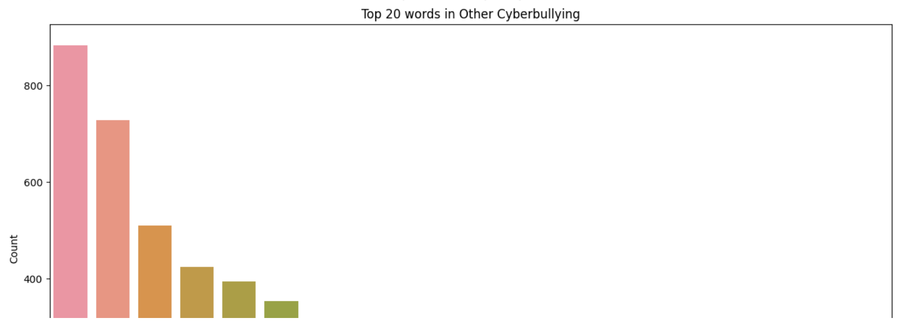


```
other_cyberbullying = Counter([item for sublist in other_cyberbullying_type['tweet_list'] for item in sublist])
top20_other_cyberbullying = pd.DataFrame(other_cyberbullying.most_common(20))
top20_other_cyberbullying.columns = ['Top Words','Count']
top20_other_cyberbullying.style.background_gradient(cmap='Greens')
```

	Top Words	Count
0	bulli	882
1	rt	727
2	fuck	509
3	like	424
4	get	393
5	peopl	353
6	go	277
7	idiot	261
8	know	241
9	think	237
10	would	235
11	u	223
12	make	215
13	one	205
14	time	202
15	hate	199
16	want	183
17	thing	182
18	need	182
19	see	178

```
fig = plt.figure(figsize=(15,8))
sns.barplot(data=top20_other_cyberbullying, y="Count", x="Top Words")
plt.title("Top 20 words in Other Cyberbullying")
```

```
Text(0.5, 1.0, 'Top 20 words in Other Cyberbullying')
```

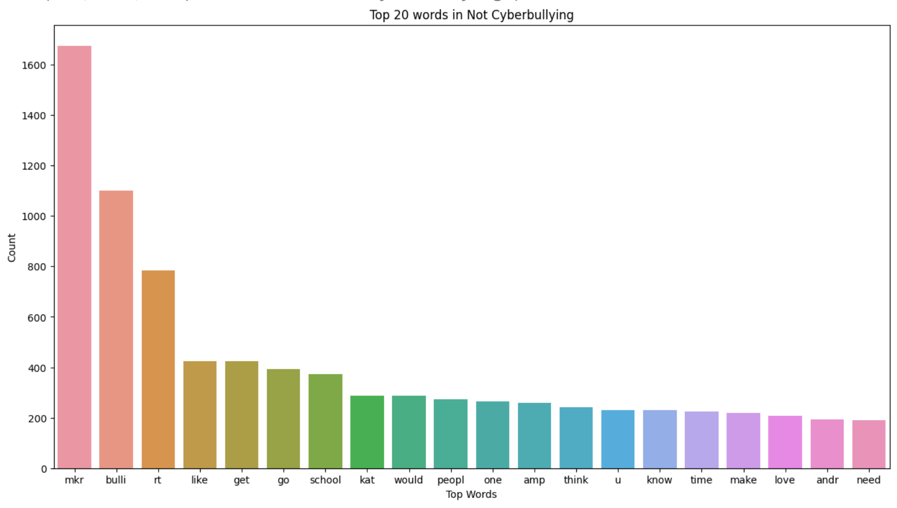


```
not_cyberbullying = Counter([item for sublist in not_cyberbullying_type['tweet_list'] for item in sublist])
top20_not_cyberbullying = pd.DataFrame(not_cyberbullying.most_common(20))
top20_not_cyberbullying.columns = ['Top Words','Count']
top20_not_cyberbullying.style.background_gradient(cmap='Greens')
```

	Top Words	Count
0	mkr	1674
1	bulli	1102
2	rt	785
3	like	424
4	get	423
5	go	394
6	school	374
7	kat	286
8	would	286
9	peopl	272
10	one	264
11	amp	258
12	think	242
13	u	231
14	know	229
15	time	224
16	make	219
17	love	207
18	andr	194
19	need	191

```
fig = plt.figure(figsize=(15,8))
sns.barplot(data=top20_not_cyberbullying, y="Count", x="Top Words")
plt.title("Top 20 words in Not Cyberbullying")
```

```
Text(0.5, 1.0, 'Top 20 words in Not Cyberbullying')
```



```
data.head()
```

	text	sentiment	sentiment_encoded	cleaned_text	tweet_list
0	In other words #katandandre, your food was cra...	not_cyberbullying	6	word katandandr food crapilici mkr	[word, katandandr, food, crapilici, mkr]
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	6	aussietv white mkr theblock imacelebrityau	[aussietv, white, mkr, theblock,

```
sentiments = ["religion", "age", "ethnicity", "gender", "other_cyberbullying","not_cyberbullying"]
```

data into training and test sets

```
X,Y = data['cleaned_text'],data['sentiment_encoded']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, stratify =Y, random_state = 42)
print(X_train.shape, y_train.shape, X_test.shape, y_test.shape)
```

```
    (31255,) (31255,) (13395,) (13395,)
```

```
tf_idf = TfidfVectorizer()
X_train_tf = tf_idf.fit_transform(X_train)
X_test_tf = tf_idf.transform(X_test)
print(X_train_tf.shape)
print(X_test_tf.shape)
```

```
    (31255, 29111)
    (13395, 29111)
```

```
log_reg = LogisticRegression()
```

```
log_cv_score = cross_val_score(log_reg,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
```

```
mean_log_cv = np.mean(log_cv_score)
mean_log_cv
```

```
    0.8228272280708214
```

```
lin_svc = LinearSVC()
```

```
lin_svc_cv_score = cross_val_score(lin_svc,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
mean_lin_svc_cv = np.mean(lin_svc_cv_score)
mean_lin_svc_cv
```

```
    0.8220066371295554
```

```
multiNB = MultinomialNB()
```

```
multiNB_cv_score = cross_val_score(multiNB,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
mean_multiNB_cv = np.mean(multiNB_cv_score)
mean_multiNB_cv
```

```
dtree = DecisionTreeClassifier()
```

```
dtree_cv_score = cross_val_score(dtree,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
mean_dtree_cv = np.mean(dtree_cv_score)
mean_dtree_cv
```

```
0.8064850614081852
```

```
rand_forest = RandomForestClassifier()
rand_forest_cv_score = cross_val_score(rand_forest,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
mean_rand_forest_cv = np.mean(rand_forest_cv_score)
mean_rand_forest_cv
```

```
0.8288706779795108
```

```
adab = AdaBoostClassifier()
adab_cv_score = cross_val_score(adab,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
mean_adab_cv = np.mean(adab_cv_score)
mean_adab_cv
```

```
0.7600867316662788
```

```
svc1 = LinearSVC()
param_grid = {'C':[0.0001,0.001,0.01,0.1,1,10],
             'loss':['hinge','squared_hinge'],
             'fit_intercept':[True,False]}
grid_search = GridSearchCV(svc1,param_grid,cv=5,scoring='f1_macro',n_jobs=-1,verbose=0,return_train_score=True)
grid_search.fit(X_train_tf,y_train)
```

```
▸ GridSearchCV
▸ estimator: LinearSVC
    ▸ LinearSVC
```

```
grid_search.best_estimator_
```

```
▾ LinearSVC
LinearSVC(C=1, loss='hinge')
```

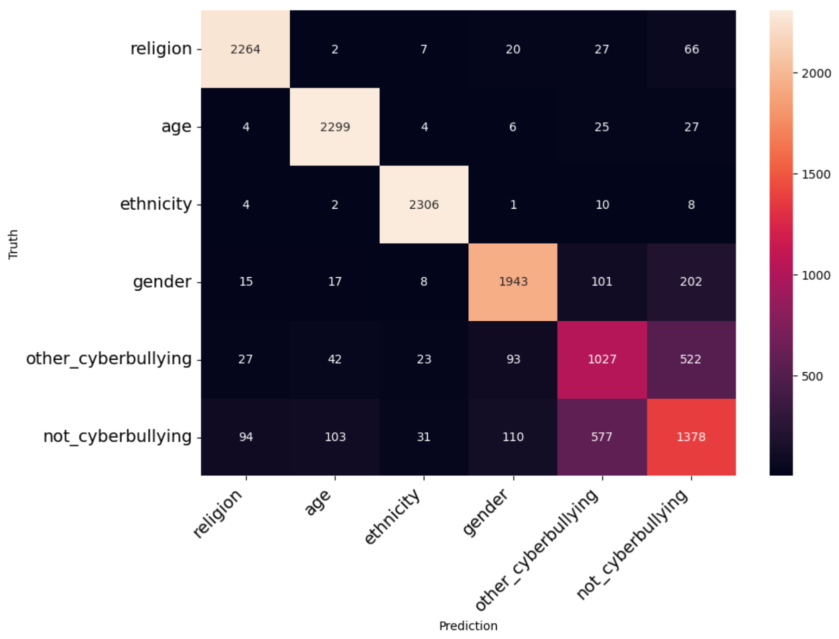
```
grid_search.best_score_
```

```
0.8244012024546198
```

```
lin_svc.fit(X_train_tf,y_train)
y_pred = lin_svc.predict(X_test_tf)
```

```
def print_confusion_matrix(confusion_matrix, class_names, figsize = (10,7), fontsize=14):
    df_cm = pd.DataFrame(confusion_matrix, index=class_names, columns=class_names)
    fig = plt.figure(figsize=figsize)
    try:
        heatmap = sns.heatmap(df_cm, annot=True, fmt="d")
    except ValueError:
        raise ValueError("Confusion matrix values must be integers.")
    heatmap.yaxis.set_ticklabels(heatmap.yaxis.get_ticklabels(), rotation=0, ha='right', fontsize=fontsize)
    heatmap.xaxis.set_ticklabels(heatmap.xaxis.get_ticklabels(), rotation=45, ha='right', fontsize=fontsize)
    plt.ylabel('Truth')
    plt.xlabel('Prediction')
```

```
cm = confusion_matrix(y_test,y_pred)
print_confusion_matrix(cm,sentiments)
print_confusion_matrix(cm,sentiments)
```



```
print('Classification Report:\n',classification_report(y_test, y_pred, target_names=sentiments))
```

```
 Classification Report:
                      precision    recall  f1-score   support

           religion       0.94      0.95      0.94      2386
                age       0.93      0.97      0.95      2365
          ethnicity       0.97      0.99      0.98      2331
             gender       0.89      0.85      0.87      2286
other_cyberbullying       0.58      0.59      0.59      1734
  not_cyberbullying       0.63      0.60      0.61      2293

           accuracy                           0.84     13395
          macro avg       0.82      0.83      0.82     13395
       weighted avg       0.84      0.84      0.84     13395
```

✓ 0s completed at 4:56 PM