HATE SPEECH DETECTION USING TRANSFORMERS (DEEP LEARNİNG)

Bora Engin Deniz

2200356078

Hacettepe University

Problem Definition

The term hate speech is understood as any type of verbal, written or behavioural communication that attacks or uses derogatory or discriminatory language against a person or group based on what they are, in other words, based on their religion, ethnicity, nationality, race, colour, ancestry, sex or another identity factor.

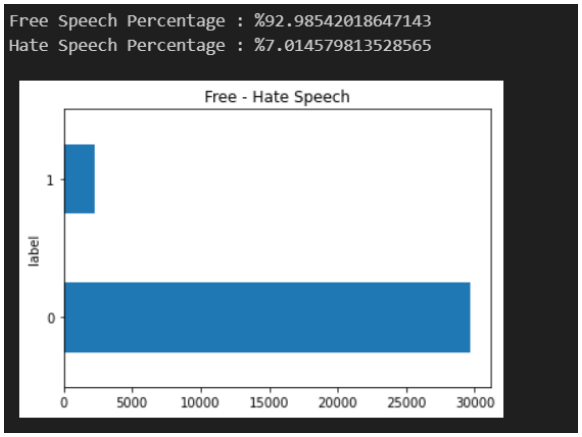
Hate Speech Detection is a task of sentiment classification. So for training, a model that can classify hate speech from a certain piece of text can be achieved by training it on a data that is used to classify sentiments. So, for the task of hate speech detection model, we will use the Twitter tweets to identify tweets containing Hate speech.

Datasets

I used two datasets to train my model and generating new predictions. The first dataset contains 31962 rows and 3 columns. The columns are “id,” “label” and “tweet.” The label column gives us the information about the tweet that if it contains hate speech (1) or not contains hate speech (0). I dropped the “id” column because of I did not use it so here is the information about the dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31962 entries, 0 to 31961
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   label   31962 non-null  int64 
 1   tweet   31962 non-null  object
dtypes: int64(1), object(1)
memory usage: 499.5+ KB
```

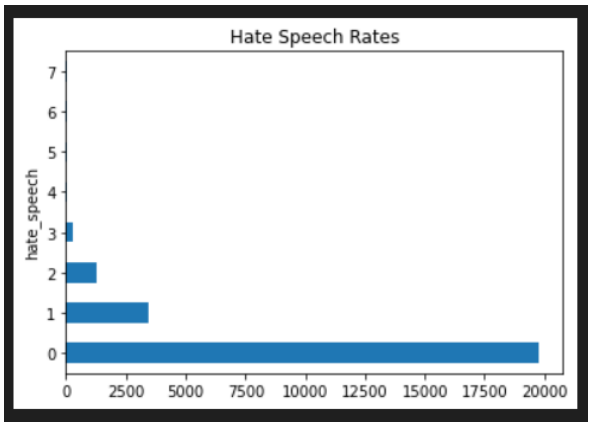
Here are the percentages of the labels we have:



As we can see above, free speech rate of our dataset is about 13 times bigger than hate speech rate. I wanted to improve the examples of the hate speech tweets, so I used another dataset. Here is the information about the dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24783 entries, 0 to 24782
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Unnamed: 0          24783 non-null  int64
 1   count               24783 non-null  int64
 2   hate_speech         24783 non-null  int64
 3   offensive_language  24783 non-null  int64
 4   neither             24783 non-null  int64
 5   class               24783 non-null  int64
 6   tweet               24783 non-null  object
dtypes: int64(6), object(1)
memory usage: 1.3+ MB
```

And here are the hate speech rates of our dataset:



After checking the tweets with different hate speech scores, I decided to use the tweets that has hate speech score equal or bigger than 2. I eliminated other but “hate speech” and “tweet” then I deleted the rows that has “hate speech” score smaller than 2. Here is the final “raw data” I used:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1574 entries, 0 to 1573
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   label   1574 non-null   int64
 1   tweet   1574 non-null   object
dtypes: int64(1), object(1)
memory usage: 24.7+ KB
```

Data Preprocessing

Text Cleaning

- Lowercase (NLP -> nlp)
- Removing Punctuation
- Removing URLs and tags

- Removing Special Characters

Tokenizing text

Splitting text into smaller parts to manipulate it better.

Removing Stop words

Removing words that have no meaning by themselves. We can give it examples like “the,” “is,” “are” etc. I used nltk “stop words” library to detect these words and remove them.

Lemmatization

For this process, we can briefly say that we identify words with similar meanings and gather them under a word to represent these words.

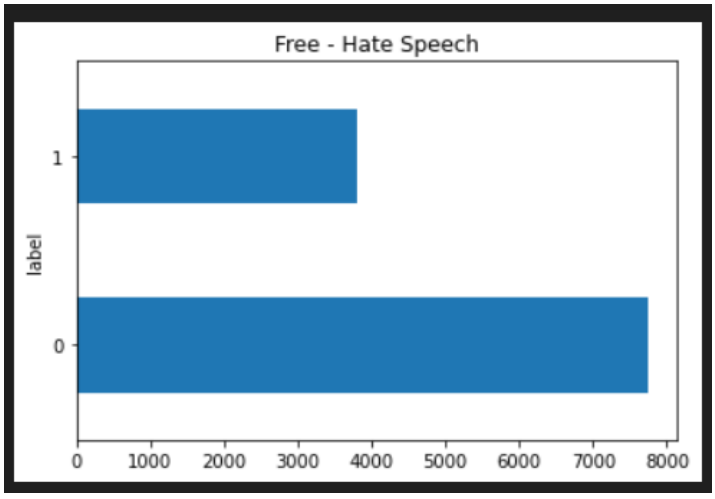
(“good” - “better” - “best”) --> “good”

WORDCLOUD

Wordcloud of First Dataset

Wordcloud of Second Dataset

I used all the hate marked tweets from the first dataset and approximately %26 of the free speech marked tweets, and all the tweets from second dataset to use it in our model. Here is the result after preprocess and concatenate these tweets:



Defining Dataset Class and Tokenizers

For Tokenizing and defining attributes, I created a class that takes dataset and tokenizer as an input. For tokenizer to do sub-word tokenizing, I used Bert Tokenizer. I also defined Model as Bert Sequence Classification with number of label equals to two because we have two options for tweets that it is either hate or free speech.

Class getting every tweet and it label, tokenizing the tweet and returning dictionary contains its “input_ids,” “attention_mask” and “label” values.

Model Building using PyTorch and Comparing Results.

After building model, here is the results of our model:

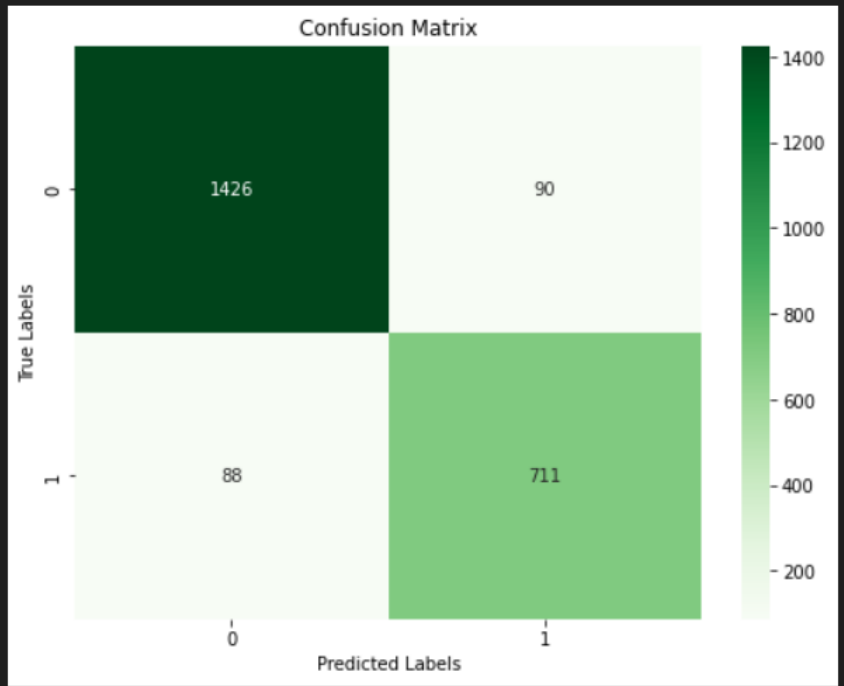
```
Accuracy: 0.923110151187905
---------------------------
              precision    recall  f1-score   support

        Free       0.94      0.94      0.94      1516
        Hate       0.89      0.89      0.89       799

    accuracy                           0.92      2315
   macro avg       0.91      0.92      0.92      2315
weighted avg       0.92      0.92      0.92      2315
```

We can say that results are good to use this model to predict. Free speech results are better than Hate speech results, basically because of size differences.

```
0 - 0 -> Predicted Trues that they are actual Trues
0 - 1 -> Predicted Trues that they are actual Falses
1 - 0 -> Predicted Falses that they are actual Trues
1 - 1 -> Predicted Falses that they are actual Falses
```



Confusion matrix of True – Predicted Labels.

Using Model with Other Inputs

I defined one last function that doing every step of our project to tweets and getting the result. I used this function to predict with new inputs. Here are the examples:

```
#input that isn't placed in our dataset.

input_text = "i love my family. They are so supportive #family"
predicted_label = classify_text(input_text)

if predicted_label == 1:
    print("The input contains hate speech.")
else:
    print("The input does not contain hate speech.")
```

✓ 0.2s

```
The input does not contain hate speech.
```

Free Speech Example

```
#input from another dataset which is not in out trained dataset.

input_text = "@user another extremist suppoing #violence  #discrimination blindly suppoing #apaheidisrael. is it a must for #cpc 2 b nuts"
predicted_label = classify_text(input_text)

if predicted_label == 1:
    print("The input contains hate speech.")
else:
    print("The input does not contain hate speech.")
```

✓ 0.0s

```
The input contains hate speech.
```

Hate Speech Example

Thank You