



# Data Science Intern at Data Glacier

**Project:** Hate Speech Detection using Transformers (Deep Learning)

**Week 9:** Deliverables

**Name:** Pranav Boyapati

**University:** University of Texas at Dallas

**Email:** [pranav13b@gmail.com](mailto:pranav13b@gmail.com)

**Country:** United States of America

**Specialization:** Data Science

**Batch Code:** LISUM34

**Date:** July 30, 2024

**Submitted to:** Data Glacier

## Table of Contents:

1. Project Plan .....	3
2. Problem Statement .....	3
3. Data Collection .....	3
4. Data Preprocessing .....	4
4.1.Text Cleaning .....	4
4.1.1. Lower Case .....	4
4.1.2. Remove Punctuation .....	4
4.1.3. Remove URL.....	4
4.1.4. Remove @tags .....	4
4.1.5. Remove Special Characters .....	4
4.2.Preprocessing Operations .....	4
4.2.1. Tokenization .....	4
4.2.2. Remove Stop Words .....	5
4.2.3. Lemmatization .....	5
4.2.4. WorldCloud .....	5
4.3.Feature Extraction .....	6
4.3.1. TF-IDF Model .....	6
4.4.Split the Dataset into Train and Test .....	6

## 1. Project Plan

Weeks	Date	plan
Weeks 07	July 19, 2024	Problem Statement, Data Collection, Data Report
Weeks 08	July 26, 2024	Data Preprocessing (Text Cleaning)
Weeks 09	August 2, 2024	Data Preprocessing (Preprocessing Operation + Feature Extraction)
Weeks 10	August 9, 2024	Building the Model
Weeks 11	August 16, 2024	Model Result Evaluation
Weeks 12	August 23, 2024	Flask Development + Heroku
Weeks 13	August 30, 2024	Final Submission (Report + Code + Presentation)

## 2. Problem Statement

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, color, ancestry, sex or another identity factor. In this problem, we will take you through a hate speech detection model with Machine Learning and Python.

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.

## 3. Data Collection

The Data is about Twitter hate Speech taken from Kaggle [1] which contains the 3 number of features and 31962 number of observations. Dataset using Twitter data, it was used to research hate-speech detection. The text is classified as: hate-speech, offensive language, and neither. Due to the nature of the study, it is important to note that this dataset contains text that can be considered racist, sexist, homophobic, or offensive.

Table 1: Data Information

<b>Total number of observations</b>	31962
<b>Total number of files</b>	1
<b>Total number of features</b>	3
<b>Base format of the file</b>	csv
<b>Size of the data</b>	3.12 MB

## **4. Data Preprocessing**

In part, we explain the data preprocessing approach that we apply in the text data.

### **4.1 Text Cleaning**

First, we clean our text because it was so messy data.

#### **4.1.1 Lowercase**

Converting a word to lower case (NLP -> nlp). Words like Racism and racism mean the same but when not converted to the lower case those two are represented as two different words in the vector space model (resulting in more dimensions). Therefore, we convert all text word into lower case letter.

#### **4.1.2 Remove Punctuation**

It is important to remove the Punctuation because is not important. Therefore, we remove that Punctuation to do that we use regular expression.

#### **4.1.3 Remove URLs**

In this part, we remove URLs because we are working on hate speech application which detect the hate and free speech and to get the output, we need to give only text not URLs therefore, we remove the URLs because we need only clean text input.

#### **4.1.4 Remove @tags**

In this part, we remove @tags which used when we mentioned someone So, it's doesn't concern to our application therefore, we remove @tags by using regular expressions.

#### **4.1.5 Remove Special Characters**

Remove Special Characters is essentially the following set of symbols [!"#\$%&'()\*+,-./:;<=>?@[^\_`{|}~] which basically don't have meaning. Therefore, we remove that kind of symbols because we don't need that. To remove we use python isalnum method.

## **4.2 Preprocessing Operation**

In this part, we implement the preprocessing operation

### **4.2.1 Tokenization**

Tokenization is breaking the raw text into small chunks. Our text data is into paragraph so to convert into work tokenize we use nltk work\_tokenize library. These tokens help in

understanding the context or developing the model for the NLP. The tokenization helps in interpreting the meaning of the text by analyzing the sequence of the words.

#### 4.2.2 Removing StopWords

StopWords is basically 'a,' 'is,' 'the,' 'are' etc. If we see our dictionary, then these words do not have meaning and don't need that to build Hate speech detection application. To remove stop words from a sentence, we divide text into words which we did above in tokenization and then remove the word if it exists in the list of stop words provided by NLTK. To do that, we first import the StopWords collection from the nltk.

#### 4.2.3 Lemmatization

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. Lemmatization is like stemming but it brings context to the words. So, it links words with similar meanings to one word. Like the word Intelligently, intelligence, convert into root form intelligent.

#### 4.2.4 WordCloud

A Wordcloud is a visual representation of text data, which is often used to depict keyword metadata on websites, or to visualize free form text. Tags are usually single words, and the importance of each tag is shown with font size or color.

Below you see the hate speech and free speech WordCloud:

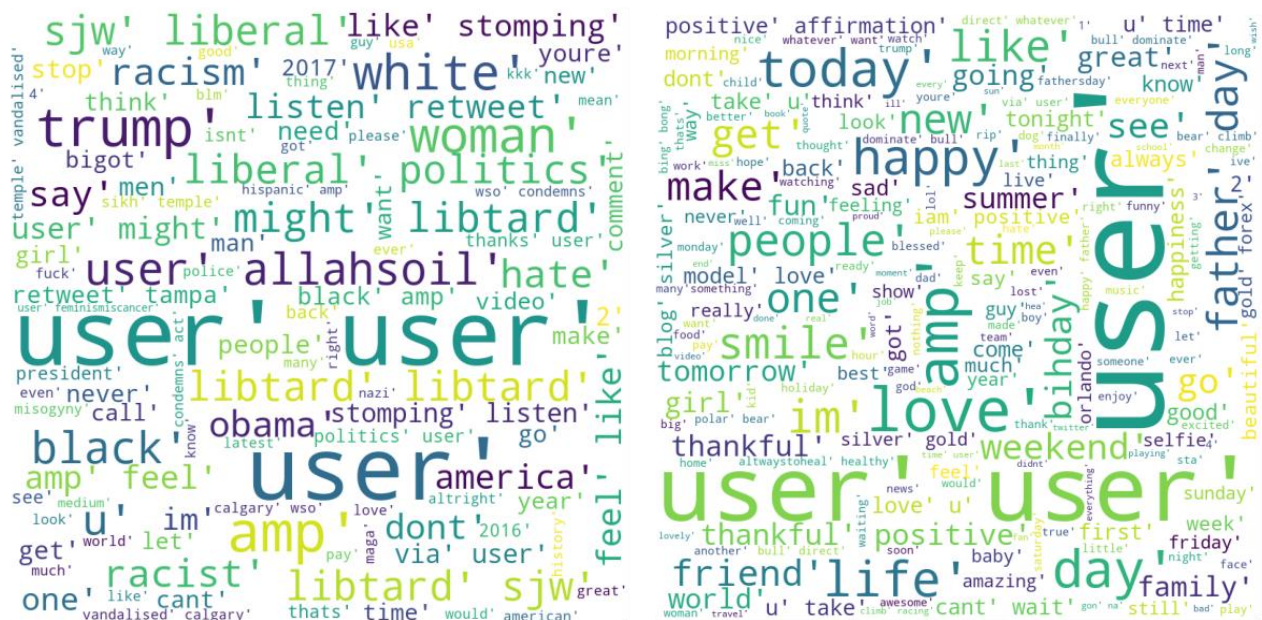


Figure 1: Hate Speech vs FREE Speech WordCloud

## **4.3 Feature Extraction**

### **4.3.1 TF-IDF Model**

Once the dictionary is ready, we apply Term Frequency-Inverse Document Frequency (TF-IDF) model, and we take 2000 most frequent words from dictionaries for each Hate/Free Speech of the whole dataset. Each word count vector contains the frequency of 2000 words in the whole dataset file.

## **4.4 Split the Data into Train into Test**

In this part, we split the data into Train. And we split 80% for training and 20% for test. Data splitting is when data is divided into two or more subsets. Typically, with a two-part split, one part is used to evaluate or test the data and the other to train the model. Data splitting is an important aspect of data science, particularly for creating models based on data.

## Reference

- [1] [https://www.kaggle.com/datasets/vkrahul/twitter-hate-speech?select=train\\_E6oV3lV.csv](https://www.kaggle.com/datasets/vkrahul/twitter-hate-speech?select=train_E6oV3lV.csv)