Data Science Intern at Data Glacier

Project: Hate Speech Detection using Transformers (Deep Learning)

Team Member: Manhui Zhu

Email: zmanhui09@outlook.com

Country: China

College: University of Southern California

Specialization: Data Science

Table of Contents

1. Project Plan

Weeks	Date	Deliverables
Week 7	June 19, 2024	Problem Statement, Data Intake Report, Project Plan
Week 8	June 26, 2024	Data Preprocessing
Week 9	July 2, 2024	EDA (Exploratory Data Analysis)
Week 10	July 9, 2024	Feature Extraction
Week 11	July 16, 2024	Model Building and Training
Week 12	July 23, 2024	Model Performance Evaluation
Week 13	July 30, 2024	Final Submission (Slides + Report + Code)

2. Problem Statement

The term hate speech is understood as any type of verbal, written or behavioral 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 generally a task of sentiment classification. A model that can classify hate speech from a certain piece of text can be achieved by training it on a data that is generally used to classify sentiments. For the task of hate speech detection model, we will use the Twitter tweets to identify tweets containing Hate speech.

3. Data Intake Report

Name: Twitter Hate Speech
Report date: 06/19/2024
Internship Batch: LISUM33
Version: 1.0
Data intake by: Manhui Zhu
Data Intake reviewer: Data Glacier
Data Storage location: https://github.com/Manhui-z/Data-Glacier-Internship/tree/0083a551094656a2b96e6b2b64fd353394d34756/Week%207

Tabular data details:

Name of data	hate_speech.csv
Total number of observations	31962
Total number of features	3
Base format of the file	.csv
Size of the data	2.95 MB

Proposed Approach:

- The full dataset is consisting of 3 features: \`id\` with data type int64, \`label\` with data type int 64, and \`tweet\` with data type object.
- There is no missing values in the dataset.

4. Data Preprocessing

There are mainly 4 approaches to transform raw text into a structured format, making it easier for models to analyze and learn from the data. They are text cleaning, removing stop words, tokenization, and lemmatization.

4.1 Text Cleaning

The usual tweets have many causal colloquial expressions, special characters, and emojis. These messy texts make it difficult for the model to learn the underlying pattern and classify the hate speech and non-hate speech. Therefore, we need to clean the text first before we fit data into the model for training. Here are detailed steps.

- **Lowercasing:** For the same words apple and Apple, the computer will recognize them as different words. To avoid this from happening, we need to all words in lowercase.
- **Removing User Mentions:** @Users is used when we mentioned someone in our tweets. It usually doesn't have any special meanings, so we remove @Users by using `re` (regular expression) package.
- **Removing URLs:** Since the model cannot directly interpret whether the content represented by the URLs is problematic, it is not helpful for the model training, so we remove it.
- **Removing Special Characters:** For better text understanding, we remove special characters in tweets by using `string.punctuation` module which containing characters like `!"#$%&'()*+,-./:;<=>?@[\]^_{|}~`.
- **Removing leading and trailing whitespace:** we remove meaningless whitespace before and after each tweet.

4.2 Removing Stop Words

Stop Words are some high-frequency common words in English language expression like 'and', 'the', 'is', but they may not contribute to the meaning and context of the sentence. We use `nltk` library to help remove the stop words. This reduces number of words the model needs to handle so that models can focus on more meaningful words, which improves efficiency and reduce noise.

4.3 Tokenization

Tokenization splits text into smaller units called token, it is usually in units of words. It is the foundation for other steps in NLP task, like stemming, lemmatization, and vectorization, which take tokens as their input.

4.4 Lemmatization

Lemmatization reduces words to their base or root form by considering the context and meaning of the word. It leads to better consistency in features, reduces redundancy, and helps in normalizing text, which is useful for classification task that requiring semantic understanding.