**Toxic Message classifier**

**Members:**

Syeda Khadija K181135

Haiqa Azeem K181105

Nabia K180144

**Section:** H

**Introduction**

Discussing things you care about can be difficult. The threat of abuse and harassment online means that many people stop expressing themselves and give up on seeking different opinions. Cyberbullying is a major societal problem that can lead to teen suicides

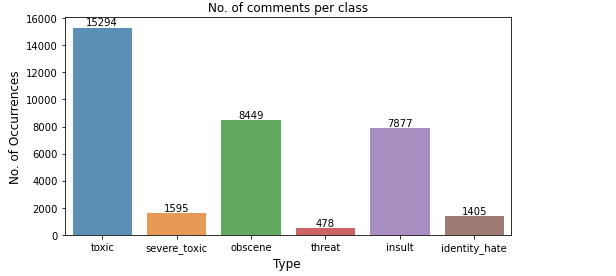
Platforms struggle to effectively facilitate conversations, leading many communities to limit or completely shut down user messages.

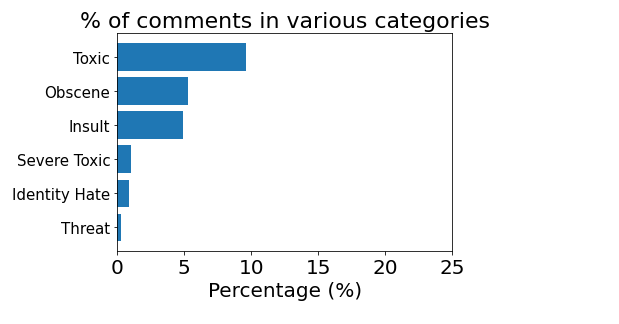
The Conversation AI team, a research initiative founded by Jigsaw and Google (both a part of Alphabet) are working on tools to help improve online conversation. One area of focus is the study of negative online behaviors, like toxic messages (i.e. messages that are rude, disrespectful or otherwise likely to make someone leave a discussion). So far they’ve built a range of publicly available models served through the Perspective API, including toxicity. But the current models still make errors, and they don’t allow users to select which types of toxicity they’re interested in finding (e.g. some platforms may be fine with profanity, but not with other types of toxic content).

In this project a multi-headed model is built that’s capable of detecting different types of toxicity like:

1. Threats,
2. Obscenity
3. Insults
4. Identity-based hate
5. Toxic
6. Severe toxic

Improvements to the current model will hopefully help online discussion become more productive and respectful.

* **These were the results on the dataset that we had from kaggle.**



**Methodology**

The aim of this work is to identify the toxic messages and to achieve this objective, we have developed a machine learning based solution. The Dataset to train the model was downloaded from Kaggle.

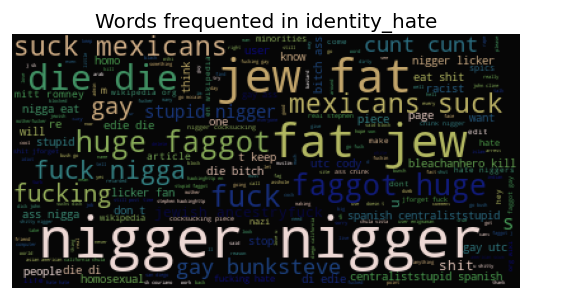
Toxic Comment Classifier is implemented using Natural Language Processing (NLP).The datasets are read from kaggle. Toxic comment classifier helps to filter out toxic words to make online world a safer and more harmonious place.

**Preprocessing**

IT is removing capital letters, numbers, punctuation and '\n'. It will load the CSV and take a peek at 1st 5 rows. Entries with all 0 under the 6 categories are neutral and considered as non-toxic. In cleaning, all special characters, the numbers, and the single letter words are removed. We got the clean dataset after these steps having no special characters, multi-spaces, capital letters or single letter word. And the dataset is split into tokens with the help of these functions ***CountVectorizer***, ***TfidfVectorizer***.

Separate our dataset into 6 sections. Each section is comment + 1 category. Creating WordCloud Useful to show the words which occur most frequently for each category.

**Wordcloud output:**



**Stop words removal**

The stop words such as and, the, was, etc. frequently appear in the text and are not helpful for prediction process, hence they are removed.

**Feature extraction**

On the preprocessed dataset, the features are extracted using the Tf-Idf. We have to provide a fixed size numerical vector as input to the machine learning based classification model or learning algorithms. For modelling and calculation of tf-idf vector we used scikit learn library function.

***sklearn.feature\_selection.***

RDF trained model and TF-IDF vectorizer object is created for each category and it calculates the F1 scores (f1 score is btter than accuracy as it factors into account false positive and false negative rates that needs to be minimized) across all models which include:

1. Log Regression
2. KNN
3. BernoulliNB
4. MultinomialNB
5. SVM
6. Random Forest

**TEXT CLASSIFICATION**

It is easy for human to classify images or text, but it is difficult for computers, which deal only with numbers, and to be more accurate, they process numbers in the form of electrical impulses. Therefore, any data must be converted into that form so computer can process it and give us back the result. The text classification algorithms use Natural Language Processing (NLP), Data Mining, and Machine Learning techniques to classify online comments [7-9]. Thus, be

Thus, before classifying, we have to analyze and vectorize the input data, and extract features from the text. Looking at the data of Kaggle, we can see two csv files: "train.csv" and" test.csv". We use Python 3 software and its packages such as Pandas to upload these files, manipulate and examine the data.

The classification was done using six different models and their accuracy score was recorded.

1. Log Regression :

The logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more independent variables.

1. KNN:

K-nearest neighbors (KNN) algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems. However, it is mainly used for classification predictive problems in industry.

1. BernoulliNB

The Bernoulli variation, generates a Boolean indicator about each term of the vocabulary equal to 1 if the term belongs to the examining document and 0 if it does not.

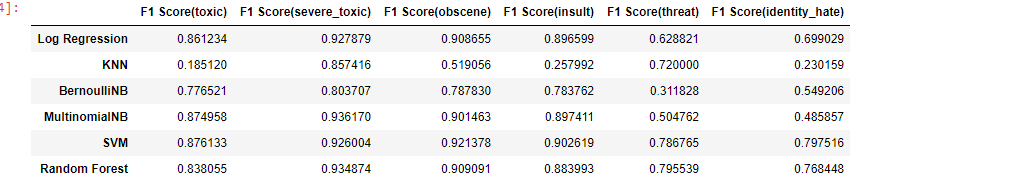
1. MultinomialNB

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

1. SVM

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they’re able to categorize new text.

1. Random Forest:

It is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

**Risk and dependencies**

The main issue is the linguistic diversity of hate, which is not fully captured by a dictionary. For example, keywords cannot detect sarcasm and forms of humor. Moreover, the dictionaries of hateful words and insults require constant updates, as new terminology and slang quickly develop in social media.

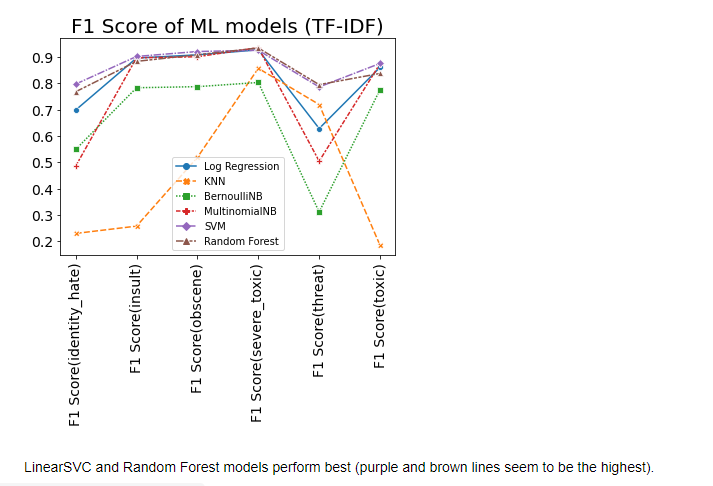
Furthermore, standards (i.e., what is interpreted as hate) differ by the online community, so that an expression that is hateful in one community may be considered neutral, humor, or typical discourse in another community

**Results**

Flask app that outputs probability that a message falls under various categories of toxicity. Random Forest is used to train our model instead of LinearSVC although the latter performs well, as RDF has predict\_proba function and LinearSVC does not. We need to output a probability score for each message. The results generated according to random forest will give the probability of the following:

1. Threats,
2. Obscenity
3. Insults
4. Identity-based hate
5. Toxic
6. Severe toxic

After reviewing all the results of F1 score on all the models applied on all 6 different categories, it could easily be interpreted by the following graph that random forest and SVM had the highest score that means good accuracy.



So we used Random forest to train our model in which 70% was training and 30% testing.

**User Interface:**

User was asked to input a message according to these six categories, the probabilities were shown:



For example in this input,the word jew is used so the probability shown by our model for identity hate is 0.8 which is too high.

**Conclusion**

Threat and abuse on online platforms is getting very common as the technology is progressing. Threats and abuse online discourages people from expressing themselves. To cater the issue we developed a system in which the messages falling in any of these categories will be shown along with the severity of it. We concluded that Random forest was the most accurate to use out of all the six models so we trained it accordingly.

1. Threats
2. Obscenity
3. Insults
4. Identity-based hate
5. Toxic
6. Severe toxic

**References**

* <https://scikit-learn.org/stable/user_guide.html>
* <https://stackoverflow.com/>
* <https://www.datacamp.com/>
* <https://www.kaggle.com/>