Hate Speech Detection Using Decision Tree Algorithm

J. Lavanya a, M. Ramesh a, J. Sravan Kumar a, G. Rajaramesh a* and Subhani Shaik a

a Department of IT, Sreenidhi Institute of Science and Technology (Autonomous), Hyderabad, India.

Authors’ contributions
This work was carried out in collaboration among all authors. All authors read and approved the final manuscript.

Article Information
DOI: 10.9734/JAMCS/2023/v38i81791

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here:
https://www.sdiarticle5.com/review-history/101251

Received: 01/04/2023
Accepted: 05/06/2023
Published: 19/06/2023

Abstract

The advancement of the internet and social media, people has access to various platforms to freely share their thoughts and opinions on various topics. However, this freedom of expression is abused to incite hatred against individuals or groups of people based on race, religion, gender, etc. question. Therefore, to address this emerging problem on social media sites, recent studies have used various feature engineering techniques and machine learning algorithms to automatically detect hate speech posts on different datasets. Advances in machine learning have intrigued researchers seeking and implementing solutions to the problem of hate speech. Currently, we are using decision tree algorithm technique to detect hate speech using text data.

Keywords: Groups of people; social media; feature engineering; decision tree.

*Corresponding author: Email: rajarameshbg@sreenidhi.edu.in;

1 Introduction

Hate speech can be defined as any speech directed at a group of people on the basis of race, religion, ethnicity, nationality, sexual orientation or gender identity. Hate speech is often used to spread hatred and discrimination. It can also be used to intimidate and threaten people. Identifying hate speech is often a job of classifying feelings. Thus, for training, a model capable of classifying hate speech from a specific text snippet can be obtained by training on data typically used to classify sentiment. Therefore, for the task of a hate speech detection model, in this project we used data from Twitter. Hate content on social media continues to rise. While Face book, Twitter, and Google have tried various measures to combat this hateful content, most of them risk violating free speech. Hate speech, on the other hand, offers an effective way to combat online hate without losing freedom of expression [1].

Therefore, another tactic for these platforms may be to promote hate speech as a means of repelling hateful content. However, to successfully promote this type of hate speech requires a solid understanding of its dynamics in the online world. This understanding is greatly hampered by the lack of well-preserved data. Hate Speech Detection is a model for identifying and tracking hateful and offensive speech on the Internet. Social media is where many people make hateful and offensive comments about others [2,3]. Therefore, hate speech detection has become an important method for solving problems in today's online world. Global Internet access has fundamentally changed our view of the world. Social media (SM) is a byproduct of the World Wide Web that comes in many forms: online gaming platforms, dating apps, forums, online news services, and social networks. Various social networks target different purposes: dissemination of ideas (Twitter or Facebook), professional contacts (LinkedIn), sharing of images (Instagram), dissemination of videos (YouTube), meetings (Tinder), etc. They all have one thing in common: they try to connect people [4,5,6].

The power of social media is such that the number of active users worldwide will reach 3.02 billion per month by 2021. This would represent about a third of the world's population. Among the many social networks in existence, Twitter is currently one of the major platforms and one of the most important data sources for researchers. Twitter is a well-known public web of real-time micro blogs, where news often predates official media. Due to its short message limit (currently 280 characters) and unfiltered feed, its usage has grown rapidly, especially at events, averaging 500 million tweets per day [7].

2 Literature Review

Hate speech detection using machine learning is an area of research that has received increasing attention in recent years due to the rise of hate speech and its negative impact on society. In this literature review, we look at some of the most important research on using machine learning to detect hate speech.

"Automated Hate Speech Detection and the Offensive Language Problem" by Davidson et al. (2017) This study proposes a deep learning approach to detect hate speech in Twitter posts [8]. The authors trained a convolution neural network (CNN) model on a dataset of tweets labeled as hate speech or not. The results show that the model achieves high accuracy and precision in detecting hate speech [9].

"Hate Speech Detection on Twitter: A Comparative Study" by Bhardwaj et al. (2019) In this study, the authors compared the performance of different machine learning algorithms, including Naive Bayes, Support Vector Machines (SVM) and Random Forests, in detecting hate speech on Twitter [10]. Waseem and Hovy, "Deep Learning for Hate Speech Detection in Tweets" (2016) [11] This study proposes a deep learning approach to detect hate speech in Twitter posts using a neural network to long- and short-term memory (LSTM) [12]. Deshpande and Patel, “Detecting Hate Speech in Social Media Using Ensemble Learning” (2018) In this study, the authors propose an ensemble learning approach for detecting hate speech on Twitter. They combined several machine-learning algorithms, including support vector machines, logistic regression and naive Bayes, to improve model performance. "Hate Speech Detection: Problem Solved? by Davidson et al. (2017) [13] This article presents a dataset on hate speech and offensive language called the Hate Speech and Offensive Language (HSOL) dataset, which is Neither the collection of tweets is unlabeled. The authors score an F1 score of 0.93 on their data set, showed the effectiveness of their method. "A Hybrid Approach to Hate Speech Detection in Twitter", by Samghabadi and Bagheri (2020) This article proposes a hybrid approach that combines machine learning and rule-based approaches to detect hate speech [12].
3 Methodology

A decision tree topic is a representation technique for generating tree structures where each node represents a test on an attribute value and each branch represents the result of the test. The leaves speak to the class. It shows the connections found in the training dataset. This method is quick unless the preparation is very useful. He does not speculate as to the distribution of probabilities for these particular data. The process of constructing a shaft is called inducement. 

Building decision trees: The decision tree algorithm is a top-down greedy algorithm, which means producing a tree whose leaves are as uniform as can reasonably be expected. The actual step of the algorithm is to isolate non-uniform sheets into sheets as uniform as expected in this case, until no further splitting is possible.

Proposition Systems Interpretability: Decision trees provide a transparent and intuitive way to understand how hate speech detection models make decisions. The tree structure allows the user to walk through the reasoning process and identify the most important features that contribute to the decision.

Speed and efficiency: Decision trees are fast and efficient algorithms for classification tasks, making them suitable for real-time applications. They can handle large datasets with high dimensions and are computationally efficient. Robustness: Decision trees are relatively insensitive to noise and outliers in the data, making them less likely to overfit or underfit the model. This reduces errors and improves model accuracy. Flexibility: Decision trees can be easily modified and adapted to different types of data, making it a versatile tool for hate speech detection.

4 Implementation

Steps in building Hate Speech detection using Decision Tree

Before jumping straight to the implementation part, let's dive into the steps involved in building a hate speech detection project using Python [14].

- Set up the development environment
- Understand the data
- Import the required libraries
- Preprocess the data
- Splitting the data
- Construct the model
- Measure the results

1. Setting up the development environment

The first step is to set up a development environment to create a hate speech detection project using Python. To develop a hate speech detection project, you must have jupyter notebook software installed on your system. Otherwise, you can also use Google Collab https://colab.research.google.com/ for the development of the project [14].

2. Understanding the data

The dataset used to build our hate speech detection model is available at www.kaggle.com. This dataset consists of Twitter hate speech detection data and is used to study hate speech detection. The text in the data was categorized as hate speech, offensive language, and neither.

Given the nature of the research, it is important to note that this dataset includes texts that can be considered racist, sexist, homophobic or generally offensive. The Hate Speech Detection dataset is comprised of 7 columns that include the index, count, hate speech, offensive language, none, class and tweet [14].

- **Index** - This column has an index number.
- **Count** - Indicates how many users coded each tweet.
- **Hate Speech** - This column lists the number of users who have tagged a tweet as hateful.
- **Offensive Language** – There is the number of users who rated the tweet as offensive.
- **Neither** – This indicates the number of users who rated the tweet as neither. Offensive nor non-offensive
**Class** – It has a class label for the majority of users, where 0 signifies hate speech, 1 signifies offensive language and 2 does not designate either.

**Tweet** – The following column contains the text of the tweet.

3. **Importing the required libraries**

After analyzing the data, our next step is to import the libraries needed for the project. A few libraries we use in this project are pandas, numpy, scikit-learn, and nltk.

```python
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
```

![Fig. 1. Importing libraries](image1.png)

Let's import the NLTK (The Natural Language Toolkit) library, used for the symbolic and statistical processing of natural language for English written in Python programming language.

```python
import nltk
nltk.download("stopwords")
```

![Fig. 2. Natural language Processing](image2.png)

Once the necessary libraries have been imported, it is time to upload the data to our project.

```python
from google.colab import files
uploaded = files.upload()
```

![Fig. 3. Data upload](image3.png)
The distribution of text length seems almost the same in all three classes.

The number of tweets seems to be a lot higher in the direction of class 1.

4. Box-plot visualization

From the box-plot, it seems that category 1 tweets have a much longer text. There are also outliers present such that the length of the text will not be a useful feature to consider.
5. Preprocessing the data

As part of the initial data processing, we prepare the raw data and compare it with an automated learning model. This is a crucial first stage in the development of a ML model. When you create an ML project.

We may not find data that is clear and formatted. Before doing anything with the data, it should be cleansed and formatted. To do that, we use data preprocessing tasks.
Fig. 8. Data preprocessing tasks

We use two important terms for natural language processing, stop words and stems. Stop words are words (data) that are not useful to deal with natural language. We can avoid typing these words. Stemming is the process of creating morphological variations of stem words. We need to find better and more predictable roots for each text.

```python
def clean(text):
    text = str(text).lower()
    text = re.sub('[\d\W\s]+', '', text)
    text = re.sub('https:\/\/[\x\W\s]+', '', text)
    text = re.sub('<[^>]+>', '', text)
    text = re.sub('[\d\W\s]+', re.escape(string.punctuation), '', text)
    text = re.sub('\\n', '\n', text)
    text = re.sub('\\x\x\x\x', '', text)
    text = [word for word in text.split(' ') if word not in stopwords]
    text = '.join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text = '.join(text)
    return text

df[['tweet']] = df['tweet'].apply(clean)
print(df.head())
```

Fig. 9. Predictable roots for each text

6. Splitting the data

The next important step is to explore the dataset and split it into training and test data.
7. Construct the model

After data separation, our next task is to find the right algorithm for our model. A decision tree classifier can be used to construct a project to detect hate speech. Decision trees are a type of supervised machine learning primarily used for classification problems [15].

8. Measure the results

The final step in the development of a ML model is prediction. In this step, we can measure the performance of our model on the test items.

Output:

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hate Speech Detected</td>
<td>0.36</td>
<td>0.33</td>
<td>0.35</td>
<td>465</td>
</tr>
<tr>
<td>No hate and No offensive speech</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>1379</td>
</tr>
<tr>
<td>Offensive language detected</td>
<td>0.92</td>
<td>0.93</td>
<td>0.93</td>
<td>6335</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.88</td>
<td>0.88</td>
<td>0.88</td>
<td>8179</td>
</tr>
<tr>
<td>macro avg</td>
<td>0.70</td>
<td>0.69</td>
<td>0.70</td>
<td>8179</td>
</tr>
<tr>
<td>weighted avg</td>
<td>0.87</td>
<td>0.88</td>
<td>0.88</td>
<td>8179</td>
</tr>
</tbody>
</table>

Decision Tree, Accuracy Score: 0.8770028784937834

We can conclude that our hate speech detection model is 87% accurate.
5 Conclusion

Researchers have a long history of ignoring hate and offensive language. In this project, we are applying a decision tree algorithm to identify hate language on social media platforms. We find that our model performs remarkably well compared to previous models. We built a hate speech detection project using decision trees. Hate speech is a major issue that we see on social media platforms like Facebook and Twitter, and we can detect it with the most specific models.

**Competing Interests**

Authors have declared that no competing interests exist.
References


© 2023 Lavanya et al.; This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
https://www.sdiarticle5.com/review-history/101251