JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

UNCOVERING OFFENSIVE LANGUAGE IN TWITTER DATA VIA AUTOMATED MACHINE LEARNING AND NLP

Mohammed Arshad Ali*[1], Kallem Vinayak Adarsh[2], D Bhanu Teja[3], G. Lava Kumar[4]
*[1] UG Student, Department of Information Technology, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India
[2] UG Student, Department of Information Technology, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India
[3] UG Student, Department of Information Technology, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India.
[4] Assistant Professor, Department of Information Technology, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India.

***Abstract**:* Social networks' ubiquity has only grown in recent years. The idea of social media was to allow us to communicate with loved ones, share positive life experiences, and express our opinions to the world. But the real world isn't flawless; there are people who spread hate speech-related messages, use it to harass particular people, or even build robots whose sole purpose is to attack particular circumstances or individuals. It's difficult to identify what kind of text it is, but there are a few approaches you might take: machine learning algorithms or natural language processing, which can look at and make predictions based on the metadata attached to the text. In this piece of work. We show the results of our preliminary research on the most effective machine learning methods for identifying foul language in tweets. Following an examination of the current literature trend on contemporary text classification techniques, they previously chose Linear SVM, from which we received 92% and 95% of accuracy, and Naive Bayes algorithms, from which we acquired 90% and 93% of accuracy for our initial testing. We have employed several attribute selection strategies for the preparation of the data, which will be supported in the literature section. Following our tests, we were able to identify offensive language using NLP with XGBoost (eXtreme Gradient Boosting) with 94% accuracy and 95% recall.

***Index Terms* - Hate speech, NLP, Text Classification, Linear SVM, Naive Bayes, XGBoost, Accuracy.**

I. INTRODUCTION

Social media like Facebook, Twitter, Instagram, and TikTok have made talking to each other faster and easier. People share all sorts of stuff there, like their thoughts, feelings, and what they've done. But sometimes, mean stuff gets said on these sites. People might use hurtful words about someone's religion, race, gender, or other personal things. This can be really harmful to them and to groups of people they belong to [1], [2]. The problem is, because social media is so casual, it's hard to spot the mean stuff automatically. To fix this, we need computer programs that can learn to find and deal with mean stuff on social media. These programs use fancy math to understand the words people use and figure out when they're being mean. But it's not easy because words can mean different things depending on the situation, and new ways of being mean are always popping up. We also have to be careful with these programs so they don't unfairly punish people or stop them from saying what they want. It's a tricky problem, but if we work together, we can make social media a safer and happier place for everyone. Think of social media like a big party where everyone chats with each other online. It's super convenient, but sometimes people say mean things. They might insult someone's race, religion, or other personal stuff, which can really hurt. The tricky part is, social media language is pretty informal, so it's tough to catch all the mean stuff automatically. To tackle this, we're developing computer programs that learn to spot and deal with mean comments. These programs use smart math to understand words and figure out when they're being nasty. But it's not a simple task because words can mean different things in different situations, and new ways of being mean keep popping up.

We also have to make sure these programs are fair and don't wrongly punish people or stop them from saying what they want. It's a challenging puzzle, but if we team up, we can make social media a nicer place for everyone to hang out.

II. EXISTING SYSTEMS

The Naive Bayes is an efficient text classification algorithm because it makes use of probabilistic computations and feature independence. In contrast, linear SVM determines the best hyperplanes for class separation and performs well in high-dimensional fields. By utilizing the advantages of both algorithms—Naive Bayes for feature selection or preprocessing and Linear SVM for classification—we can perhaps boost the model's overall performance.[3]

EXISTING SYSTEM DISADVANTAGES

Furthermore, It's prediction accuracy is lower. Training a machine is a time-consuming process. The effectiveness of SVM may decrease while handling really large datasets.

PROPOSED SYSTEM

A machine learning algorithm called NLP using XGBoost is a member of the gradient boosting technique family. It is renowned for its effectiveness, quickness, and capacity to manage complicated data.[4]

Decision trees are a type of weak predictive model that XGBoost combines and trains consecutively to produce accurate predictions. The purpose of each new tree is to fix the mistakes in the preceding ones, therefore enhancing the performance of the model bit by bit.

PROPOSED SYSTEM ADVANTAGES

- Its excellent prediction precision.
- Fit for big datasets.
- NLP enables computers to comprehend and analyze spoken language.
- NLP approaches play a crucial role in machine translation by making it easier to extract structured information from unstructured text.

III. METHODOLOGIES

We have discovered numerous methods for identifying and categorizing offensive language after reviewing the literature. Neural networks and deep learning have been utilized for this, and the performance outcomes were varied in each publication. The authors of the dataset utilized achieved good results [5], [6], even if the SVM based solution has not yet surfaced. Although it is not included in the most current literature, NLP with XGBoost has shown promising outcomes with issues of a similar nature. Particularly the XGBoost classifier, these are low-cost and simple to apply. For automatic language categorization, these classifiers can offer a more affordable, quicker, and superior option compared to Deep Learning and Neural Network models. We have therefore made the decision to put these strategies into practice and assess their outcomes.

1.Date Collection:

Gathering data is the first concrete step in creating a machine learning model. This is an important phase that will have a cascading effect on the model's quality; the more and better data we collect, the more effective the model will be.

Data can be gathered using a variety of methods, including physical interventions, web scraping, and more.

A Twitter dataset is used in an NLP-inspired data augmentation method for adverse event prediction.[7]

2.Dataset:

The 24783 rows × 7 columns that make up the dataset are explained in the following sections.

1. Unnamed: This column is typically used as a reference or identifier for each row of data and has no specified name.
2. Count: The number of times a certain property or category occurs in the dataset is displayed in this column.
3. Offensive language: Whether a tweet or text contains offensive, vulgar, or unsuitable words or phrases is indicated by this column.
4. Neither: This label designates a category that is impartial and holds information that does not fit into any of the previous labels.[8]
5. Classification: Depending on the substance of the tweet, it may fall under categories like "hate speech," "offensive language," or "neither."
6. Tweet: This contains the content that is being evaluated for neutrality, offensive language, or hate speech.

3.Data Preparation:

Sort and organize data so that it is ready for training. Clean up everything that could need it (get rid of duplicates, fix mistakes, handle missing values, normalize data, convert data types, etc.). Data should be randomized to eliminate the impact of the specific sequence in which we gathered and/or otherwise processed the data.[9]

Cleaning, tokenizing, and lemmatizing the dataset's column "tweet" To obtain the similarity score and matching score, the fuzzywuzzy tool and the cosine similarity technique are also used. Divided into sets for evaluation and training.

4.Model Selection:

We applied the XGBoost method to NLP and obtained a 94% accuracy on the train set. As a result, we put this algorithm into practice.

5.Analyze and Prediction:

We selected just two features from the real dataset:

1. Class: a thorough explanation of the augmented data.
2. tweet - forecasts whether or not the unfavorable occurrences will transpire.

6.Accuracy on Test set:

Accuracy on test set: On the test set, we achieved 94% accuracy.

7.Saving the Trained Model:

Saving the Trained Model: Using a library like pickle, save your trained and tested model into a.h5 or.pkl file as soon as you feel comfortable introducing it into a production-ready setting. Verify that pickle is installed in your setting. Importing the module and dumping the model into the.pkl file come next.[10], [11]

REQUIREMENTS ENGINEERING

[12]. These are the requirements for doing the project. Without using these tools & software's we can't do the project. So we have two requirements to do the project. They are

- Hardware Requirements.
- Software Requirements

1.HARDWARE REQUIREMENTS

The hardware requirements should be a comprehensive and uniform definition of the entire system since they may form the foundation of a contract for the system's implementation. Software engineers use them as the foundation for their system designs. It should be what the system does, not how it ought to be put into practice.

- PROCESSOR : Dual Core 2 Duos.

- RAM : 4GB DD RAM
- HARD DISK : 250 GB

2. SOFTWARE REQUIREMENTS

[13]. The system specification is found in the software requirements paper. It ought to have a definition in addition to a list of specifications. Rather than focusing on how the system should operate, it is a list of what it should perform. The software requirements specification is created based on the software needs. It is helpful for tracking teams and their progress during development activities, as well as for calculating costs and organizing and carrying out team activities.

- Operating System : Windows 7/8/10
- Platform : Spyder3
- Programming Language : Python
- Front End : Spyder3

3. FUNCTIONAL REQUIREMENTS

A software system's or one of its components' functions are defined by a functional requirement. A set of inputs, the behavior, and a function are described as First off, by using a hybrid cloud architecture, the system is the first to accomplish the common understanding of semantic security for data confidentiality in attribute-based reduplication systems.

4. NON-FUNCTIONAL REQUIREMENTS

The system's primary non-functional requirements are as follows:

Utilization

Because the system is fully automated, there is either no user intervention at all or very little.

Dependability

The traits carried over from the selected Python platform make the system more dependable. Python code is more dependable when it is constructed.

Achievement

This system will respond to the user on the client system in a very short amount of time since it is being developed in high level languages and employs cutting edge back-end technologies.

Sustainability

The system is made to accommodate multiple platforms. The system is designed to work with a broad spectrum of hardware and any software platform.

Execution

Software from Jupyter notebooks is used to implement the system in a web context. The platform is Windows 10 Professional, and the server serves as the intelligence server. Interface: The Jupyter notebook serves as the basis for the user interface.[14]

IV. SYSTEM ARCHITECTURE

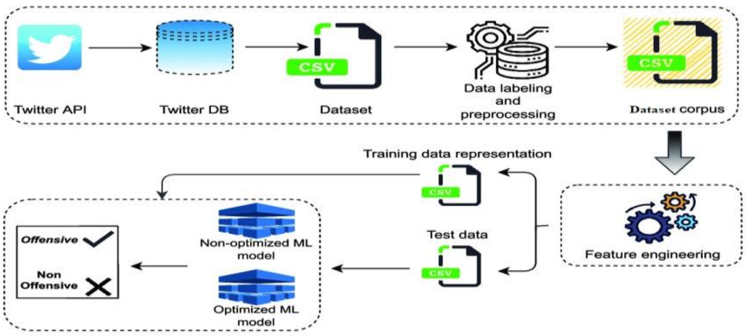

FIG: System Architecture Model

V. LITERATURE SURVEY

The task of multi-class sentiment analysis entails classifying online posts into different emotion classes. Bouazizi and Ohtsuki address this task in their 2019 work, "Multi-class sentiment analysis on Twitter: Classification performance and challenges." Although binary or ternary classification is the usual focus of sentiment analysis, multi-class classification is difficult due to the intricacy of natural languages and the subtleties of human expression. With an accuracy of 60.2%, the writers examine the categorization of Twitter postings into seven sentiment classes. This accuracy highlights the effect of numerous classes on performance, as compared to 81.3% for binary classification. They present a brand-new paradigm to express emotions and examine how they relate to one another, pointing out problems and offering solutions for the future.

In the paper they published in 2019, "Cyber social media analytics and issues: A pragmatic approach for Twitter sentiment analysis," In their exploration of social media analytics, Sharma, Jain, Bhatia, Tiwari, Mishra, and Trivedi highlight the importance of social media in the digital age. They draw attention to the dual nature of social media data, which has the potential to be abused for defamation and scamming in addition to being used for branding and image development. The study discusses the difficulties in evaluating social media data and suggests a workable strategy for thorough analysis. Their research focuses on analyzing Twitter data in real-time to identify feelings and emotions in user messages using lexicon-based and machine learning techniques. The findings enable a more thorough comprehension of user viewpoints by providing an effective classification of opinions into positive, negative, and neutral categories.

SNAPSHOTS

FIG : Interface Page

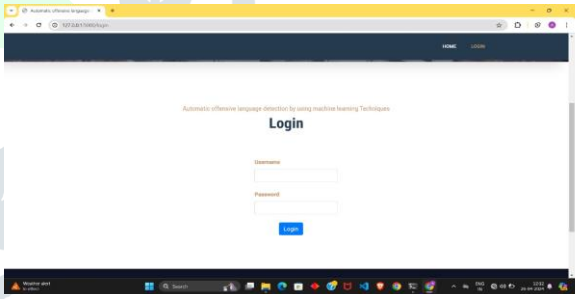

FIG: User Information

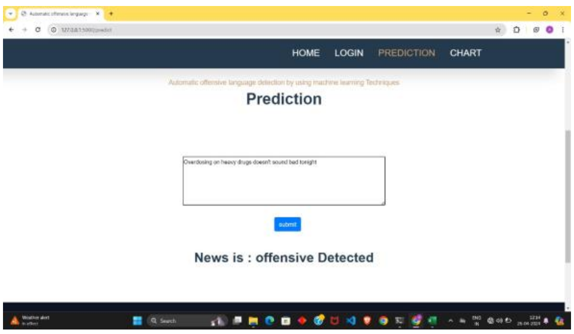

FIG: Input

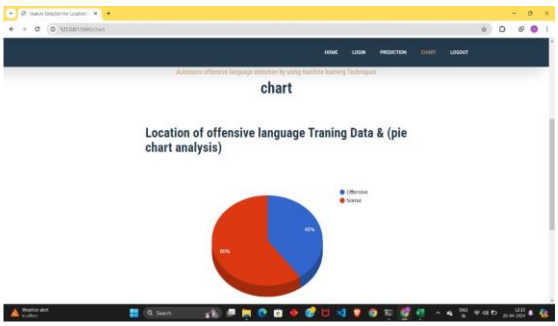

FIG: Result

VI. CONCLUSION

We have investigated the usage of Naive Bayes and Linear SVM classifiers in this work to identify abusive language in tweets. We have discovered that the Linear SVM is highly sensitive to the sort of data utilized during the training phase. It was also found that the procedure of parameter regulation was hampered by the data normalization with tags. Due of the high standard derivation for the tests with varied seeds, the tests also demonstrated that the evaluation sequence of messages has a significant impact on the classifier's final outcome. This is a typical procedure because, for instance, if lengthy strings of messages bearing the same label are provided as input. [15]

For instance, the learning is ordered by the other inputs resulting in an imbalance of the weights due to the weight regulation and the learning coefficient (alpha). For the Linear SVM to produce decent results, a balanced input was therefore required. It turned out to be a somewhat difficult effort to establish the parameters for this method.

The Naive Bayes classifier, on the other hand, turned out to be an effective text classifier. One of your advantages is that this method is incredibly quick due to its simplicity and ease of implementation. This algorithm proved to be superior to numerous methods presented in the literature.

.

VII. REFERENCES

[1] F. Del-Vigna, A. Cimino, F. Dell-Orletta, M. Petrocchi, and M. Tesconi, “Hate me, hate menot: Hate speech detection on facebook,” in First Italian Conference on Cybersecurity, 2017.

[2] J. Jacobs and K. Potter, Hate crimes: Criminal law & identity politics. Oxford UniversityPress on Demand, 1998.

[3] M. Bouazizi and T. Ohtsuki, “Multi-class sentiment analysis on twitter: Classificationpe formance and challenges,” Big Data Mining and Analytics, vol. 2, no. 3, pp. 181–194, Sep.2019.

[4] G. Jalaja and C. Kavitha, Sentiment Analysis for Text Extracted from Twitter. Singapore:Springer Singapore, 2019, pp. 693–700.

[5] S. Sharma and A. Jain, “Cyber social media analytics and issues: A pragmatic approach fortwitter sentiment analysis,” in Advances in Computer Communication and ComputationalSciences, S. K. Bhatia, S. Tiwari, K. K. Mishra, and M. C. Trivedi, Eds. Singapore: SpringerSingapore, 2019, pp. 473–484.

[6] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar, “Identifying andcategorizing offensive language in social media (offenseval),” arXiv preprint arXiv:1903.08983,2019.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectionaltransformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.

[8] P. Liu, W. Li, and L. Zou, “Transfer learning for offensive language detection usingbidirectional transformers,” in Proceedings of the 13th International Workshop on SemanticEvaluation, 2019, pp. 87–91.

[9] J. Han, S. Wu, and X. Liu, “Identifying and categorizing offensive language in social media,”in Proceedings of the 13th International Workshop on Semantic Evaluation, 2019, pp. 652–656.

[10] A. Nikolov and V. Radivchev, “Offensive tweet classification with bert and ensembles,” inProceedings of the 13th International Workshop on Semantic Evaluation, 2019, pp. 691–695.

[11] J. Risch, A. Stoll, M. Ziegele, and R. Krestel, “hpidedis at germeval 2019: Offensivelanguage identification using a german bert model,” in Preliminary proceedings of the 15thConference on Natural Language Processing (KONVENS 2019). Erlangen, Germany: GermanSociety for Computational Linguistics & Language Technology, 2019, pp. 403–408.

[12] T. Davidson, D. Warmsley, M. Macy, and I. Weber, “Automated hate speech detection andthe problem of offensive language,” in Proceedings of the 11th International AAAI Conferenceon Weblogs and Social Media, ser. ICWSM ’17, 2017.

[13] G. Pitsilis, H. Ramampiaro, and H. Langseth, “Detecting offensive language in tweets usingdeep learning,” arXiv preprint arXiv:1801.04433, 2018.

[14] Z. Mossie and J.-H. Wang, “Vulnerable community identification using hate speechdetection on social media,” Information Processing & Management, p. 102087, 2019.

[15] P. Mathur, R. Shah, R. Sawhney, and D. Mahata, “Detecting offensive tweets in hindi-english code-switched language,” in Proceedings of the Sixth International Workshop on NaturalLanguage Processing for Social Media, 2018, pp. 18–26.