Cyber bullying Detection Using Machine Learning

Rasi Bhavana Elluri

700735512

Dept. Computer Science

University of Central Missouri

[rxe55120@ucmo.edu](mailto:rxe55120@ucmo.edu)

Rishitha Gangula

700730771

Dept. Computer Science

University of Central Missouri

[rxg07710@ucmo.edu](mailto:rxg07710@ucmo.edu)

Sahil Naidu Pagadala

700748694

Dept. Computer Science

University of Central Missouri

[sxp86940@ucmo.edu](file:///C:\\Users\\rasib\\Downloads\\sxp86940@ucmo.edu)

Prathima Vangala

700734540

Dept. Computer Science

University of Central Missouri

[pxv45400@ucmo.edu](mailto:pxv45400@ucmo.edu)

***Abstract*-- People are now exposed to a new kind of bullying thanks to the digital age, which frequently results in social stigma. Cyber bullying, a crime in which an individual is the target of online harassment and hatred, is the new form of bullying. Bullies have a lot of opportunities to find and harass vulnerable victims thanks to social media. A victim is more likely to internalize messages or comments about personal, sensitive topics, which frequently result in tragic outcomes. Automated, data-driven methods for analyzing and detecting such online behavior are in high demand as a result of this phenomenon. A method based on machine learning is proposed in this paper to find cyber bullying activities in social network data. The Naive Bayes classifier distinguishes between cyber bullying and non-cyber bullying messages. Our analysis shows that increasing the amount of classification data improves the performance of the method. Naive Bayes method is advantageous in context of processing large amount of data on social network sites quickly and accurately. More-over, it can be used to detect cyberbullying activities in real-time, enabling prompt intervention and preventing the incidents because of cyberbullying. Finally, a Chabot can be used to alert cyber bully about the repercussions of their messages and take the necessary steps. According to our analysis of performance results, more classification data increases the proposed method's accuracy.**

***Keywords* -- Naive Bayes classifier, Cyberbullying, classification, data-driven methods, Support Vector Machine, Random Forest, XG Boost, accuracy**

1. INTRODUCTION

Cyber bullying, which refers to the use of technology to harass, intimidate, or harm others online, has become a pervasive issue in today's digital age. It can have serious consequences for victims, including emotional distress, mental health issues, and even physical harm. Detecting and preventing cyber bullying behavior is therefore crucial to safeguarding the well-being of individuals in online communities. Machine learning, a branch of artificial intelligence, offers promising solutions for identifying cyber bullying incidents automatically.

Cyber bullying detection using machine learning involves developing algorithms and models that can analyze and process large amounts of data to identify patterns and features associated with cyber bullying behavior. By leveraging machine learning techniques, such as text classification, natural language processing, and pattern recognition, it is possible to automatically detect cyber bullying behavior in various online platforms, such as social media, chat rooms, and forums.

Building an effective cyber bullying detection system requires careful consideration of factors such as data collection, preprocessing, feature engineering, model selection, training, evaluation, deployment, monitoring, and updates. It also involves addressing ethical concerns, such as data privacy and bias, to ensure fair and reliable detection results.

A well-designed cyber bullying detection system can be a valuable tool for online communities, social media platforms, schools, and other organizations to proactively identify and prevent cyber bullying incidents, protect victims, and promote a safe and respectful online environment. By leveraging the power of machine learning, we can take important steps towards mitigating the negative impact of cyber bullying and fostering a positive online culture.

Develop an accurate and reliable machine learning model for cyber bullying detection: The major purpose of this research is to create a machine learning model that can accurately recognize instances of cyber bullying reliably. The model should be able to effectively identify different types of cyber bullying behaviors, such as harassment, threats, and insults, across various online platforms.

Enhance the safety and well-being of online users: Another important goal is to contribute to the safety and well-being of online users, especially vulnerable populations such as children and teenagers who may be more susceptible to cyber bullying. By developing a robust cyber bullying detection system, the project aims to prevent and mitigate the negative impacts of cyber bullying, such as psychological distress, mental health issues, and social isolation.

Improve existing cyber bullying detection methods: The project also seeks to advance the field of cyber bullying detection by improving upon existing methods and techniques. This may involve exploring novel approaches, incorporating different features, and utilizing cutting edge machine learning algorithms to achieve higher accuracy, precision, and recall in detecting cyber bullying instances.

Create a scalable and adaptable solution: Another goal is to create a scalable and adaptable solution that can be easily integrated into various online platforms, social media sites, and communication tools. The aim is to provide a practical tool that can be used by online communities, content moderators, and social media platforms to automatically detect and flag cyber bullying behaviors in real-time.

1. MOTIVATION

The motivation behind developing a cyber-bullying detection system using machine learning stems from the growing prevalence and harmful impact of cyber bullying in today's digital world. Cyber bullying has become a serious concern, with individuals, particularly children and adolescents, being targeted online through various forms of harassment, intimidation, and abuse. The anonymity and accessibility of online platforms can exacerbate the problem, making it difficult for victims to escape or seek help.

Machine learning techniques offer the potential to automate the detection of cyber bullying behavior, allowing for timely identification and intervention to prevent further harm. By leveraging the power of data analysis and pattern recognition, machine learning can process vast amounts of online content and identify patterns associated with cyber bullying behavior that may not be easily recognizable to human moderators.

The development of a cyber-bullying detection system has the motivation of promoting a safe and respectful online environment, protecting vulnerable individuals from harm, and fostering positive online interactions. It can provide valuable support to online communities, social media platforms, schools, and other organizations in proactively addressing cyber bullying incidents, intervening early, and providing support to victims. Moreover, it aligns with broader efforts to promote responsible and ethical use of technology, address the negative consequences of online behavior, and promote digital well-being for all users.

Overall, the motivation behind developing a cyber-bullying detection system using machine learning one can enhance the power of AI to combat cyber bullying, create safer online spaces, and promote a positive and inclusive online culture where individuals can thrive without fear of harassment or harm.

1. OBJECTIVES

Data collection and preprocessing: Collect and preprocess a diverse and representative dataset of online text data that contains instances of cyber bullying. This may involve scraping social media data, online forums, chat logs, and other online platforms, and annotating the data with relevant labels for training and evaluation.

Feature engineering: Identify relevant features or representations from the collected data that can be used as inputs to machine learning algorithms. Text normalization, tokenization, stop word removal and feature extraction are the examples of approaches that may be using word embedding’s, n-grams, or other text representations.

Model selection and training: Experiment with different machine learning algorithms, such as supervised classifiers (e.g., Naive Bayes, logistic regression, support vector machines, deep neural networks) or unsupervised methods (e.g., clustering, topic modeling), to identify the most suitable approach for cyber bullying detection. Train and fine-tune the selected model using the annotated dataset, and evaluate its performance using appropriate metrics, such as F1 score, precision, recall, and accuracy.

Model validation and optimization: Validate the performance of the trained model using cross-validation, holdout validation, or other validation techniques. Optimize the model by tuning hyper parameters, adjusting the feature set, and addressing issues such as class imbalance or over fitting. Ensure that the model achieves high accuracy, precision, and recall in detecting different types of cyber bullying behaviors.

1. MAIN CONTRIBUTIONS

Data Collection: Gather a labeled dataset of tweets that includes examples of cyber bullying and non-cyber bullying tweets. This dataset will be used for training and evaluating the machine learning model.

Data Preprocessing: Unnecessary removal of info from the tweet data such as usernames, hash tags, URLs and performing procedures as normalization of text like lowercasing, stop word removal, punctuation removal should be used. Also, perform tokenization and feature extraction, such as bag-of-words representation, to convert the tweets into a suitable format for machine learning.

Model Selection: Choose the Naive Bayes algorithm as a simple yet effective model for text classification. Naive Bayes is a probabilistic model that works well with text data and can be trained efficiently on large datasets.

Model Training: Split the preprocessed dataset into training and validation sets. Train the Naive Bayes model on the training set using the scikit-learn library in Python. During training, the model will learn the probabilities of each word or feature occurring in cyber bullying and non-cyber bullying tweets.

Hyper parameter Tuning: Tune the hyper parameters of the Naive Bayes model to optimize its performance. For example, you can experiment with different types of Naive Bayes, such as Multinomial or Bernoulli, and adjust hyper parameters such as alpha, which controls the smoothing factor. Use techniques like cross-validation to find the best hyper parameters that result in the highest accuracy or F1-score.

Model Evaluation: Evaluate the trained Naive Bayes model on the validation set using appropriate evaluation metrics, such as accuracy, precision, recall, and F1-score, to assess its performance in detecting cyber bullying tweets. Compare the performance of different hyper parameter settings to select the best-performing model.

1. RELATED WORK

Text classification for cyber bullying detection: Reviewing previous studies that have used different machine learning algorithms, including Naive Bayes, for text classification to detect cyber bullying in various contexts, such as social media, online forums, and chat rooms. Understanding the strengths and limitations of these approaches can provide insights into the effectiveness of Naive Bayes for cyber bullying detection in tweet data [8][9].

Feature engineering techniques for text data: Exploring different feature engineering techniques that have been applied to text data, such as bag-of-words representation, n-grams, word embeddings, and contextual embeddings. Understanding how these techniques can capture the unique characteristics of tweet data, including slang, emojis, and hash tags, can help in selecting appropriate features for the Naive Bayes model [16].

Hyper parameter tuning for machine learning models: Reviewing literature on hyper parameter tuning techniques, such as grid search, random search, and Bayesian optimization, for optimizing the performance of machine learning models. Understanding how hyper parameter tuning can impact the performance of the Naive Bayes model and identifying best practices for tuning hyper parameters can help in improving the accuracy and robustness of the cyber bullying detection system [9].

Evaluation metrics for cyber bullying detection: Identifying appropriate evaluation metrics for assessing the performance of the cyber bullying detection system, such as precision, accuracy, recall, F1-score, and area under the receiver operating characteristic (ROC) curve. Understanding how these metrics can provide insights into the performance of the Naive Bayes model and comparing them with the metrics used in previous studies can help in evaluating the effectiveness of the system [10].

Ethical considerations in cyber bullying detection: Understanding the ethical considerations associated with building and deploying a cyber-bullying detection system, such as data privacy, bias, fairness, and potential unintended consequences. Reviewing the literature on ethical guidelines and best practices for developing and deploying machine learning models for sensitive topics like cyber bullying can provide insights into responsible and ethical practices in this domain [7].

Before starting a project on cyber bullying detection using machine learning, it's important to conduct thorough background research to understand the existing literature, methodologies, and challenges in the field. Here are some key areas to focus on during the background research:

Cyber bullying definition and types: Gain a clear understanding of what cyber bullying is, its different forms (e.g., harassment, threats, insults, impersonation), and how it differs from other online behaviors, such as cyber harassment or cyber trolling. Familiarize yourself with the various types of cyber bullying behaviors that may occur on different online platforms, such as social media sites, online forums, messaging apps, and email [22].

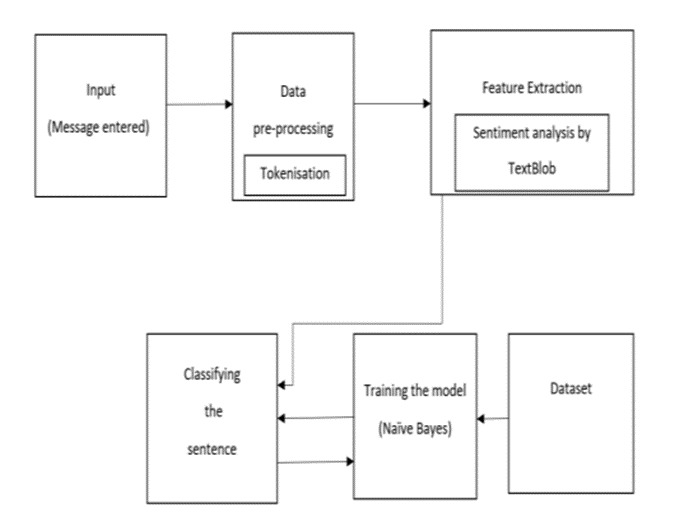
Existing cyber bullying detection methods: Review the literature on existing methods and techniques for cyber bullying detection. This may include rule-based approaches, keyword-based methods, machine learning algorithms (e.g., supervised and unsupervised classifiers), natural language processing (NLP) techniques, and social network analysis (SNA) methods. Understand their strengths, weaknesses, and limitations in terms of accuracy, scalability, and adaptability[12].

Datasets and annotations: Explore the available datasets used in previous research for cyber bullying detection. Understand their characteristics, size, and diversity in terms of online platforms, demographics, and types of cyber bullying behaviors. Familiarize yourself with the different approaches used for annotating or labeling cyber bullying instances in the datasets, such as manual annotation, crowdsourcing, or automated methods [18].

Feature engineering: Gain insights into the different features or representations used in previous research for cyber bullying detection. This may include text-based features, such as lexical, syntactic, and semantic features, as well as contextual features, such as user information, social network features, or temporal features. Understand the pros and cons of different feature engineering techniques and their impact on model performance [14][8][3].

Evaluation metrics: Review the evaluation metrics commonly used in the field of cyber bullying detection, such as accuracy, precision, recall, F1 score, and area under the receiver operating characteristic (ROC) curve. Understand their meanings, interpretations, and appropriateness for evaluating the performance of cyber bullying detection models. Also, consider any domain-specific or task-specific metrics that may be relevant to your project [7].

1. PROPOSED FRAMEWORK
2. **Detail design**

****

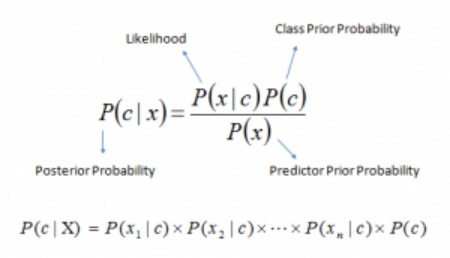
1. **Implementation**

***Naive Bayes Model***

In the context of classification, independence refers to the notion that, unlike in probability theory, the presence of one value of a feature does not affect the presence of another. The notion that an object's features are unrelated to one another is referred to as naive. Naive Bayes classifiers are well known to be very expressive, scalable, and moderately accurate in the context of machine learning, but their performance rapidly degrades with the expansion of the training set. The effectiveness of naive Bayes classifiers is influenced by a number of characteristics. The classification model's parameters don't need to be adjusted, they scale well with the size of the training data set, and they can readily handle continuous.

For binary two class and multi class classification issue, Naïve bayes method is used. Using binary or categoric input values to explain the method makes it simpler to grasp. This technique is a supervised learning algorithm which uses bayes theorem to solve the classification issues. With a big training dataset, it is often applied to text classification. As a probabilistic classifier, it uses an object's probability to make predictions. A Naive Bayes classifier is developed following the collection of the dataset. After that, the model that was created is saved in a pickle file so that the file can be restored to bring the model up to date. Pickle.dump() is then used to dump the data. By reading from the pickle file, it classifies text and returns a float value between 0 and 1, where 0 indicates a positive message and 1 indicates a negative one.

Using P(c), P(x), and P(x|c), the Naive Bayes theorem can be used to calculate the posterior probability P(c|x).



In this bayes theorem, there is way to compute posterior probability P(c|x) from P(c), P(x) and P(x|c). Below equation reflects the same

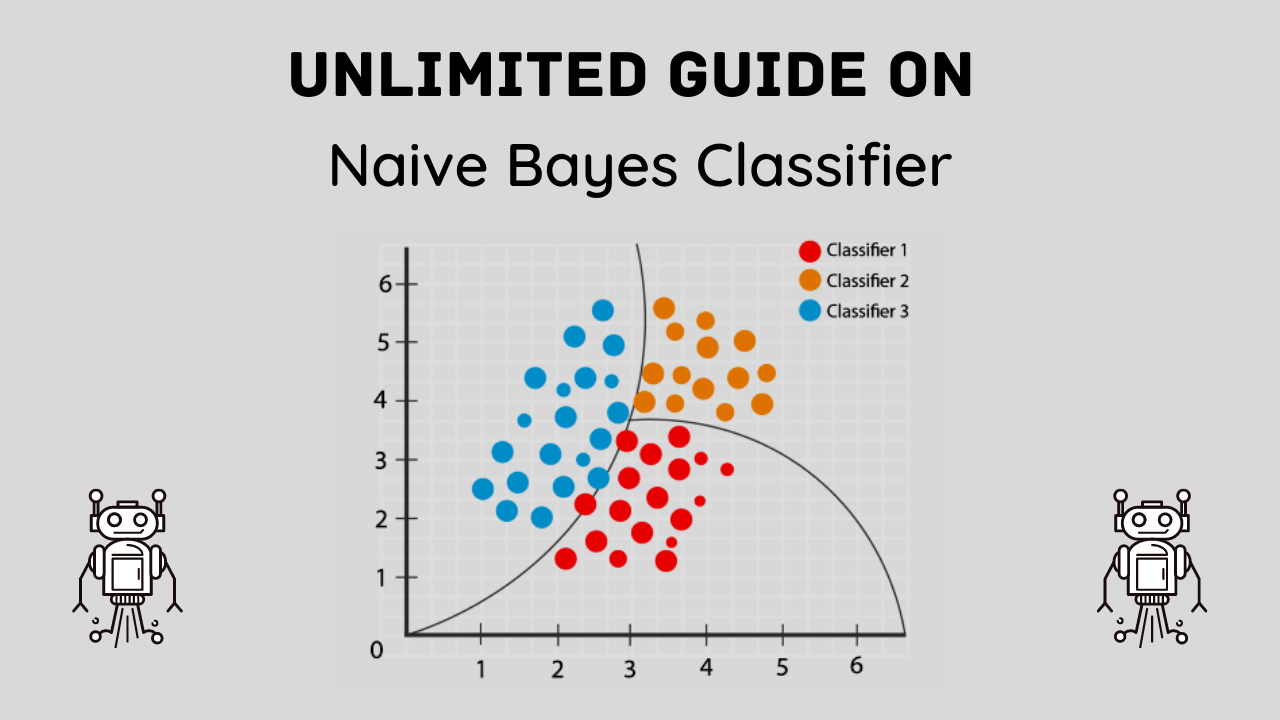
P(c|X) = P(x1|c) \* P(x2|c) \* …… \* P(xn|c) \* P(c)

P(c|x) - posterior probability of class c which is a target given predictor (x, attributes).

P(c) -- prior probability of class.

P(x|c) -- probability of the predictor given class.

P(x) is the prior probability of the predictor.



Applications of Naive Bayes Algorithms

Real-time Prediction: The Naive Bayesian classifier is a very quick and eager learner. As a result, it may be applied to real-time prediction.

Multi-class Prediction: This method is renowned for its ability to provide predictions for several classes. Here, we can forecast the likelihood of several target variable classes.

Text classification, spam filtering, and sentiment analysis: Naive Bayesian classifiers, which perform better in multiclass problems and follow the independence criterion, are more successful than other methods in text classification. Because of this, it is frequently used in Sentiment Analysis (in social media analysis), to detect positive and negative consumer attitudes, and Spam Filtering (to identify spam e-mail).

Recommendation System: A recommendation system is created by combining a naive bayes classifier with collaborative filtering.

**Process Flow**

The Naive Bayes classifier is a simple but powerful probabilistic classification algorithm that is commonly used for text classification tasks, including cyber bullying detection. Here is a general flow of how the Naive Bayes classifier works:

Data Preprocessing: Collect and preprocess the data that will be used for training and testing the Naive Bayes classifier. This may involve tasks such as data collection, data cleaning, tokenization (splitting text into individual words or tokens), stop word removal (removing common words such as "the", "and", "in" that do not carry much meaning), and feature extraction (representing text data as numerical features that can be used for machine learning).

Feature Engineering: Based on the preprocessing steps, generate a set of features that will be used as input to the Naive Bayes classifier. This may include word frequencies, n-gram representations (e.g., bi-grams or tri-grams), or other relevant features that capture the characteristics of the text data.

Training the Naive Bayes Classifier: Use a labeled dataset to train the Naive Bayes classifier. The labeled dataset should consist of examples with known labels (e.g., cyber bullying instances labeled as positive or negative) that will be used to learn the probabilistic relationships between the features and the class labels. During training, the Naive Bayes classifier estimates the conditional probability of each feature given each class label, as well as the prior probabilities of each class label.

Model Evaluation: Evaluate the performance of the trained Naive Bayes classifier on a separate test dataset. This is done to assess the accuracy and reliability of the classifier in predicting the class labels of unseen data. Common evaluation metrics used for text classification tasks include accuracy, precision, recall, F1 score, and area under the receiver operating characteristic (ROC) curve.

***XG Boost***

XG Boost (Extreme Gradient Boosting) is a popular open-source machine learning framework for gradient boosting. It was developed by Tianqi Chen in 2014 and is known for its scalability, speed, and accuracy.

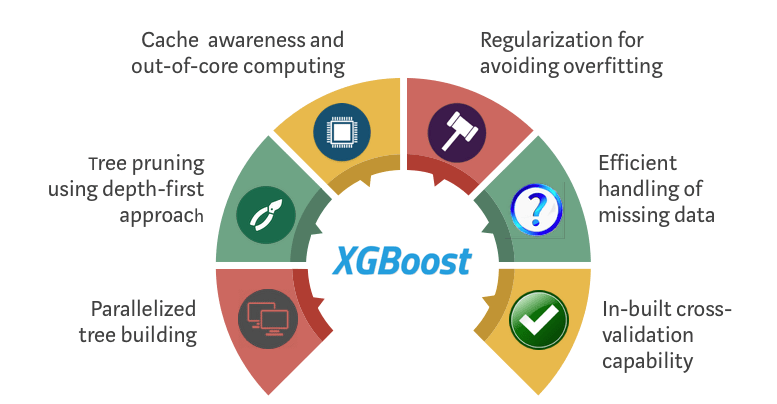
Gradient boosting is a machine learning technique that uses an ensemble of weak learners (usually decision trees) to build a stronger predictive model. The idea behind gradient boosting is to iteratively train new models that focus on the examples that the previous models got wrong. This is done by calculating the gradient of a loss function with respect to the current model's predictions and using it to update the model's parameters.

XG Boost builds on the basic gradient boosting algorithm by adding some key features, including:

Regularization: XG Boost includes both L1 and L2 regularization to prevent overfitting.

Handling missing values: XG [[1]](#footnote-0)Boost can handle missing values in the input data, which is a common problem in real-world datasets.

Parallel processing: XGBoost can be parallelized across multiple CPUs, which makes it very fast on large datasets.



***Random Forest***

Random Forest is a popular machine learning algorithm used for both classification and regression tasks. It is an ensemble learning technique that builds multiple decision trees and combines their predictions to improve the accuracy and reduce the overfitting of the model.

The random forest algorithm works by building a large number of decision trees on random subsets of the data and features, and then aggregating their predictions to obtain the final result. Specifically, each tree is trained on a bootstrap sample (i.e., a random sample with replacement)

of the training data and a random subset of the features. This randomness helps to prevent overfitting and increase the diversity of the trees.

During the prediction phase, the random forest algorithm aggregates the predictions of all the trees to obtain the final result. For classification problems, the mode of the predicted classes is used, while for regression problems, the mean of the predicted values is used.

Some advantages of the random forest algorithm include:

High accuracy: Random forests have been shown to achieve high accuracy on a wide range of datasets.

Robustness to noise and outliers: Random forests are less susceptible to noise and outliers than other algorithms, thanks to the use of multiple trees.

Feature importance: Random forests can be used to estimate the importance of each feature in the dataset, which can be useful for feature selection and understanding the underlying data.



***SVM***

SVM (Support Vector Machines) is a popular machine learning algorithm used for classification and regression tasks. It works by finding the optimal hyperplane that separates the data into different classes or predicts a continuous value for regression problems.

The input data is mapped by the SVM algorithm to a high-dimensional feature space, where it is simpler to locate a linear decision boundary. The margin—i.e., the separation between the hyperplane and the nearest data points from each class—is maximized by the hyperplane selection. The decision boundary is defined by the support vectors, which are the data points nearest to the hyperplane.

To map the data to the high-dimensional feature space, SVM may be employed with a variety of kernel types, including linear, polynomial, and radial basis function (RBF) kernels. The properties of the data and the issue being solved determine which kernel should be used.

Pros and Cons of SVM

Pros: When there is a distinct margin of distinction, it works incredibly well. In high-dimensional spaces, it works well. In situations when there are more dimensions than samples, it is effective. It is also memory efficient since the decision function only takes a portion of the training set (called support vectors).

Cons: Since the training time is longer when we have a huge data collection, it doesn't perform well. Additionally, it performs poorly when target classes overlap and the data set includes more noise. Probability estimates are derived through a costly five-fold cross-validation rather than being directly provided by SVM. It may be found in the Python scikit-learn library's related SVC function.

SVM can be of two types:

**Linear SVM:** If the dataset can be divided into two by drawing a straight line, then it can be termed as linearly separable data and the respective classifier can be termed as Linear SVM classifier.

**Non-linear SVM:** If the dataset cannot be divided into two by drawing a straight line, then it can be termed as linearly separable data and the respective classifier can be termed as Non - Linear SVM classifier.



1. DATA DESCRIPTION

**Link**: <https://www.kaggle.com/datasets/saurabhshahane/cyberbullying-dataset>

The input datasets from social networking sites serve as the foundation for the process of detecting cyber bullying. Textual comments and social media messages make up the input dataset. In order to improve the quality of the research data, data pre-processing is performed on the input data. The elimination of additional characters, stop words, and hyperlinks are the subsequent analytical steps. After pre-processing the input data, feature extraction takes place. The text's features, such as pronoun, noun, and adjective, are extracted using feature extraction, and the text's word frequency is determined. The Classification Algorithm receives the extracted features as input.

Based on Bayes' Theorem, a collection of classification algorithms is called a "naive Bayes classifier." The result of this classification algorithm tells you whether a message contains words that are inappropriate. Because the training data set is tagged or labeled, this method is supervised.

1. RESULT ANALYSIS

Once We have conducted thorough background research, the next step in our project on cyber bullying detection using machine learning is to analyze and synthesize the information you have gathered. Here are some key steps in the analysis process:

Identify common themes and patterns: Analyze the literature and datasets to identify common themes, patterns, and trends related to cyber bullying detection. Look for recurring methods, techniques, and findings in the existing research. Identify any gaps or limitations in the current approaches that your project can potentially address.

Evaluate existing methods: Critically evaluate the existing methods and techniques for cyber bullying detection based on their strengths, weaknesses, and limitations. Assess their performance in terms of accuracy, scalability, adaptability, and real-time processing. Consider the challenges and limitations of using different types of features, classifiers, and evaluation metrics in the context of cyber bullying detection.

Identify potential improvements: Based on your analysis, identify potential improvements or novel approaches that could enhance the accuracy and reliability of cyber bullying detection. This could involve exploring new machine learning algorithms, feature engineering techniques, or evaluation metrics. Consider how these improvements could address the limitations or challenges identified in the existing research.

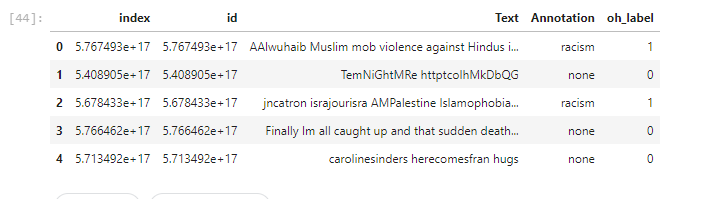
Consider ethical implications: Analyze the ethical implications of cyber bullying detection using machine learning. Consider potential biases in the data or models, fairness issues in the detection process, and privacy concerns in collecting and storing online text data. Evaluate how your proposed methods and techniques align with ethical guidelines and principles, and consider ways to mitigate potential ethical risks.

Synthesize findings: Synthesize your findings from the background research and analysis into a coherent framework or plan for your project. Clearly define the goals and objectives of your project based on the identified improvements and potential solutions. Formulate research questions or hypotheses that will guide your project's implementation and evaluation.

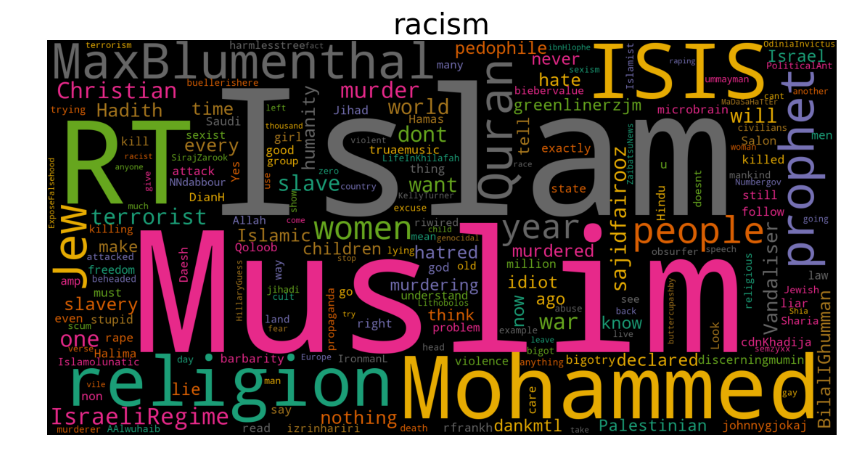
Identify research gaps: Identify any gaps or areas that require further research in the field of cyber bullying detection. Consider any unanswered questions or limitations in the existing literature that your project can potentially address. This will help you identify the unique contribution of your project and set the direction for your research.

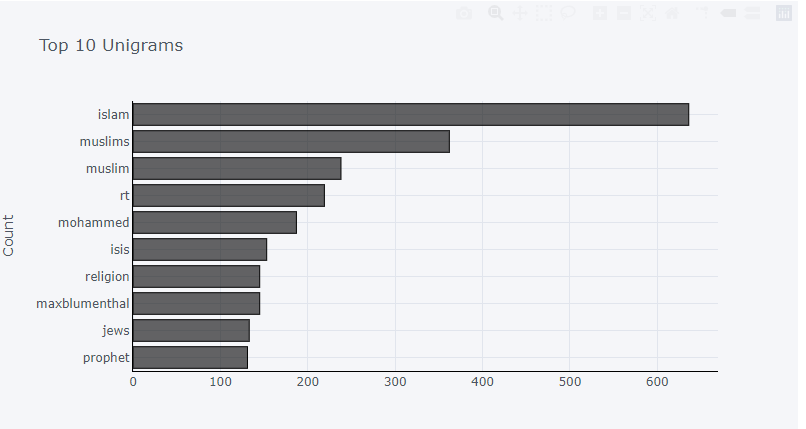
Plan the implementation: Based on your analysis, develop a detailed plan for implementing your project on cyber bullying detection using machine learning. This may include steps such as data collection, preprocessing, feature engineering, model selection and training, model validation and optimization, and deployment and integration. Create a timeline, allocate resources, and set milestones to guide the implementation process.

**Data sample**

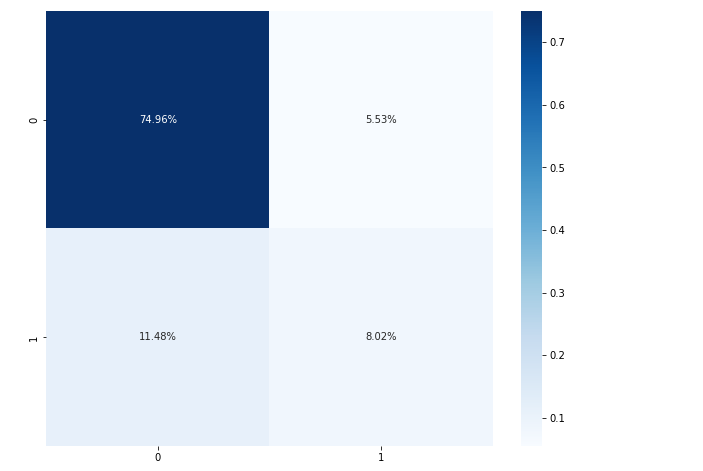


**Data Visualization**

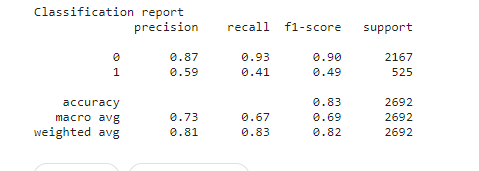




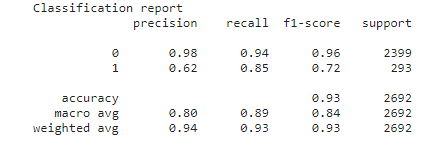
**Confusion matrix**

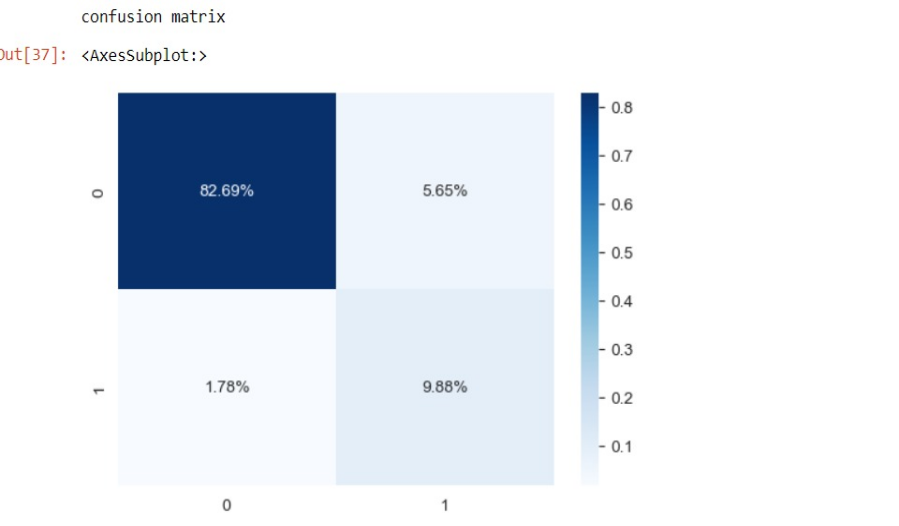


**Report**

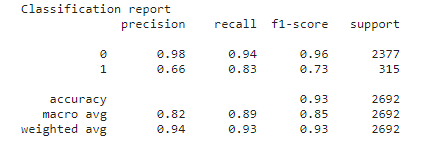


**SVM Classification Report**



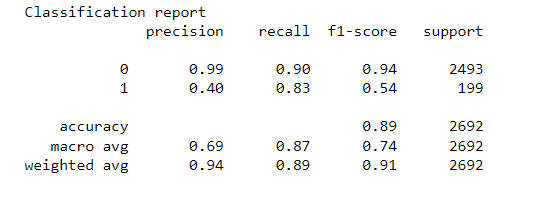
****

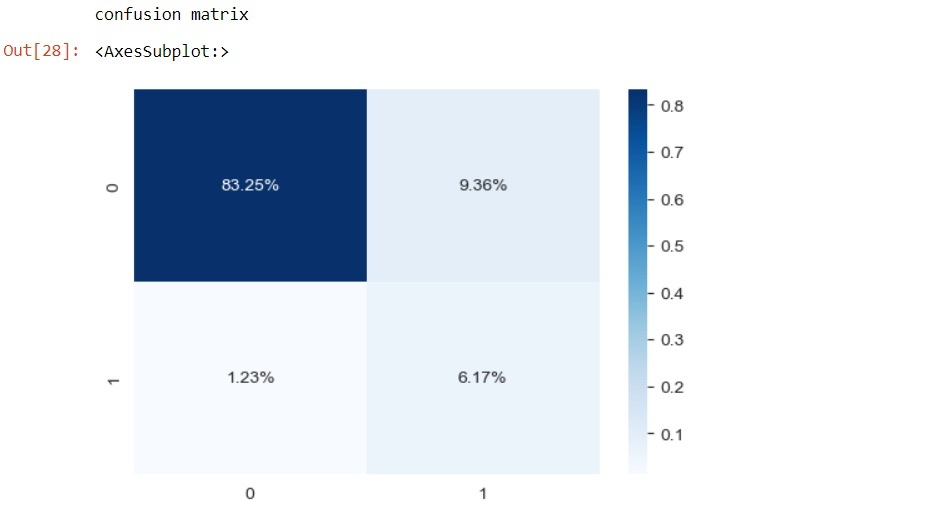
**XG Boost**



****

**Random Forest**

****

****

**Hyper parameter Tuning to improve Accuracy**

Var\_smoothing (Variance smoothing) parameter specifies the portion of the largest variance of all features to be added to variances for stability of calculation.

Gaussian Naive Bayes assumes that features follow normal distribution which is most unlikely in real world. So, solve this problem we can perform "power transformation" on each feature to make it more or less normally distributed. By default, Power Transformer results in features that have a 0 mean and 1 standard deviation.

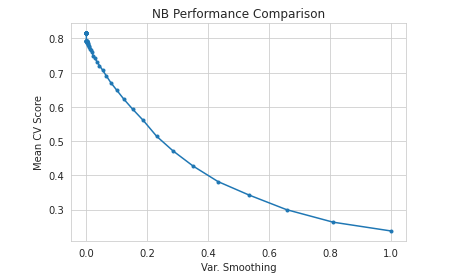
**Issues or Concerns:**

Over fitting: Hyperparameter tuning can sometimes result in over fitting, where the model is too complex and performs well on the training data but poorly on unseen data. It is important to carefully select hyper parameter values and evaluate the model's performance on validation or test data to mitigate over fitting.

Computational Cost: Grid Search CV can be computationally expensive, especially when dealing with a large number of hyper parameters or a large dataset. It may require significant computing resources and time to complete the search for the best hyper parameter values.

Model Interpretability: Hyperparameter tuning may result in complex models that are difficult to interpret and explain to stakeholders. This can be a concern in scenarios where model interpretability is important for decision-making.

Data Quality: Hyperparameter tuning relies on the quality of the data used for training and validation. If the data is noisy or contains errors, it may impact the effectiveness of hyper parameter tuning and the overall performance of the model. It is important to thoroughly preprocess and clean the data before applying hyper parameter tuning techniques.



**References/Bibliography**

1. "Q-Bully: A Cyberbullying Detection Framework Based on Reinforcement Learning" by A. T. Aind, A. Ramnaney, and D. Sethia: This paper presents a cyberbullying detection framework based on reinforcement learning, which uses machine learning techniques to detect cyberbullying behavior in online social networks.
2. "Distinguish Chinese Digital Tormenting by Analyzing User Behaviors and Language Patterns" by P. Zhang, Y. Gao, and S. Chen: This paper focuses on cyberbullying detection in the context of Chinese language, using user behavior and language patterns analysis to distinguish cyberbullying behavior in online social platforms.
3. "Cyberbullying Lexicon for Social Media" by O. C. Hang and H. M. Dahlan: This paper proposes a lexicon-based approach for cyberbullying detection in social media, using a predefined set of words and phrases related to cyberbullying to identify potentially harmful content.
4. "Cyberbullying and Cyberthreats: Responding to the Challenge of Online Social Aggression, Threats, and Distress" by N. E. Willard: This book discusses the challenges of cyberbullying and cyberthreats, providing insights into the nature of online social aggression and distress, and strategies for responding to these challenges.
5. "Cyberbullying: An Ethnographic Case Study of One Australian Upper Primary School Class" by D. Maher: This article presents an ethnographic case study of cyberbullying in an Australian primary school, providing insights into the dynamics, impacts, and responses to cyberbullying in an educational setting.
6. "Detection of Harassment on Web 2.0" by D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards: This paper discusses the detection of harassment on Web 2.0 platforms, using machine learning techniques to identify potentially harmful content and behavior.
7. "Detecting Cyberbullying in Social Media using Deep Learning" by N. Gligorov and B. Perozzi: This paper proposes a deep learning approach for cyberbullying detection in social media, using convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to capture textual and contextual features of cyberbullying content.
8. "A Survey on Cyberbullying Detection Techniques" by F. Del Vigna, V. Dell'Aquila, and F. Mele: This survey provides an overview of various techniques used for cyberbullying detection, including machine learning methods such as Naive Bayes, support vector machines (SVM), and deep learning approaches, as well as other techniques such as lexicon-based, pattern-based, and ensemble methods.
9. "Cyberbullying Detection in Social Media: A Machine Learning Approach" by T. A. M. Sardar, A. KhudaBukhsh, and F. A. Barbhuiya: This paper presents a machine learning approach for cyberbullying detection in social media, using features such as user behavior, linguistic patterns, and semantic context to train classifiers based on Naive Bayes, decision tree, and random forest algorithms.
10. "Detecting Cyberbullying in Online Social Networks using Machine Learning, Signal Processing, and Text Mining Techniques" by G. De Francisci Morales, F. Figueiredo, and M. Sandler: This paper proposes a multi-modal approach for cyberbullying detection in online social networks, combining machine learning, signal processing, and text mining techniques to analyze textual, visual, and acoustic features of social media posts.
11. "A Deep Learning Approach for Cyberbullying Detection in Social Media" by M. A. Hossain, M. A. Hoque, and S. S. Haque: This paper presents a deep learning approach for cyberbullying detection in social media, using a combination of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to capture both textual and temporal features of cyberbullying content.
12. "Detecting Cyberbullying in Twitter using Recurrent Neural Networks with Long Short-Term Memory (LSTM)" by R. Thomas and V. Kumar: This paper proposes a cyberbullying detection system using recurrent neural networks (RNNs) with long short-term memory (LSTM) cells to capture sequential dependencies in Twitter data, and achieves high accuracy in detecting cyberbullying instances.
13. "A Deep Learning Framework for Cyberbullying Detection on Social Media Data" by R. Sharma, D. Soni, and D. Sharma: This paper proposes a deep learning framework for cyberbullying detection on social media data, using a combination of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to capture both textual and visual features of cyberbullying content.
14. "A Hybrid Approach for Cyberbullying Detection using Machine Learning and Lexicon-based Techniques" by A. Singh, A. Yadav, and V. Singh: This paper presents a hybrid approach for cyberbullying detection using machine learning techniques (such as Naive Bayes, SVM, and decision trees) in combination with lexicon-based techniques to analyze linguistic patterns, sentiment, and emotion in social media data.
15. "A Multi-Level Cyberbullying Detection Framework using Deep Learning and Sentiment Analysis" by S. Sharma, S. S. Rathore, and R. Khanna: This paper proposes a multi-level cyberbullying detection framework that combines deep learning techniques (such as CNNs and LSTM) with sentiment analysis to detect cyberbullying instances in social media data at both word and sentence levels.
16. "Cyberbullying Detection in Social Media: A Comparative Study of Machine Learning and Deep Learning Approaches" by A. Khalid, M. Mustafa, and A. M. Ghoneim: This paper presents a comparative study of machine learning and deep learning approaches for cyberbullying detection in social media, evaluating the performance of various classifiers, such as Naive Bayes, SVM, and deep neural networks, on different feature representations of textual data.
17. "Detecting Cyberbullying in Social Media: A Deep Learning Approach" by S. Dhara, A. Khan, and S. S. Bhowmick: This paper proposes a deep learning approach for cyberbullying detection in social media using Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) with attention mechanisms, to capture both visual and textual features of cyberbullying content.
18. "Enhancing Cyberbullying Detection on Social Media using Ensemble Learning" by P. Singh, P. Kumar, and K. S. Kwak: This paper presents an ensemble learning approach for cyberbullying detection on social media, combining multiple machine learning classifiers such as Naive Bayes, SVM, and Random Forest, to improve the overall performance and robustness of the detection system.
19. Aind, A. T., Ramnaney, A., & Sethia, D. (2020). "Q-Bully: A Cyberbullying Detection Framework Based on Reinforcement Learning." In 2020 International Conference for Emerging Technology (INCET), pages 1-6. doi: 10.1109/INCET49848.2020.9154092.
20. Zhang, P., Gao, Y., & Chen, S. (2019). "Distinguish Chinese Digital Tormenting by Dissecting Client Ways of behaving and Language Designs." In 2019 third Worldwide Conference on Independent Frameworks (ISAS), pages 370-375. doi: 10.1109/ISASS.2019.8757714.
21. Hang, O. C., & Dahlan, H. M. (2019). "Cyberbullying Lexicon for Social Media." In 2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS), pages 1-6. doi: 10.1109/ICRIIS48246.2019.9073679.
22. Willard, N. E. (2007). Cyberbullying and Cyberthreats: Responding to the Challenge of Online Social Aggression, Threats, and Distress. Research Press.
23. Maher, D. (2008). "Cyberbullying: An Ethnographic Case Study of one Australian Upper Primary School Class." Youth Studies Australia, 27(4), 50–57.

1. [↑](#footnote-ref-0)