“Cyberbullying Detection Using Machine Learning Techniques”

A Report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNVERSITY ANANTAPUR.

In Partial Fulfillment of the Requirements for the Award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
BY

V. NAGESH	**19121A05R0**
V.GANGA PRADEEP	**19121A05R3**
V. HEMANTH KUMAR REDDY	**19121A05R7**
V. SINDHURA	**19125A05R9**
T. CHIRUDEEP	**19121A05P3**

Under the Guidance of

Dr. M. Sakthivel
Professor
Dept of CSE, SVEC

Department of Computer Science and Engineering
SREE VIDYANIKETHAN ENGINEERING COLLEGE
(Affiliated to JNTUA, Anantapuramu)
Sree Sainath Nagar, Tirupathi – 517 102
2019-2023

SREE VIDYANIKETHANENGINEERINGCOLLEGE

(Affiliated to Jawaharlal Nehru Technological University Anantapur)
Sree Sainath Nagar, A. Rangampet, Tirupati – 517 102, Chittoor Dist., A.P.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project Work entitled

"Cyberbullying Detection Using Machine Learning Techniques "

is the bonafide work done by

V. NAGESH	**19121A05R0**
V.GANGA PRADEEP	**19121A05R3**
V. HEMANTH KUMAR REDDY	**19121A05R7**
V. SINDHURA	**19125A05R9**
T. CHIRUDEEP	**19121A05P3**

In the Department of Computer Science and Engineering, Sree Vidyanikethan Engineering College, A.Rangampet. is affiliated to JNTUA, Anantapuramu in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2019-2023.

This is work has been carried out under my guidance and supervision.

The results embodied in this report have not been submitted in any University or Organization for the award of any degree or diploma.

Internal Guide

Dr. M. Sakthivel
Professor
Dept of CSE
Sree Vidyanikethan Engineering College
Tirupathi

Head

Dr. B. Narendra Kumar Rao
Prof & Head
Dept of CSE
Sree Vidyanikethan Engineering College
Tirupathi

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

ABSTRACT

Research on cyberbullying detection is gaining increasing attention in recent years as both individual victims and societies are greatly affected by it. Access to social media is very easy and the mistreatment of people through informal words and comments bullying, sexism, racism, aggressive content, harassment, toxic comment, etc has increased. Hence there needs to be a monitoring system established to control and reduce these types of misbehaviour and cyberbullying that spread around social networks which has inspired to development of this research to automate the detection process of cyberbullying. Our major motivation is to build a model that categorizes offensive and non-offensive. For this process, we have developed a machine-learning algorithm that uses Random Forest and J48 for feature extraction and detection analysis of Twitter data.

Keywords: Cyber Bullying, Sentiment Analysis, Behavioural Analysis.

INTRODUCTION

Cyberbullying is known as an unnoticed crime and there are many victims who have been bullied online through toxic comments. Some statistics show that as many as 50% of children have experienced cyberbullying. Victims of cyberbullying can experience wide-ranging effects, including mental health issues, poor academic performance, a desire to drop out of school, and even suicidal ideation. Even though many victims face this, many of them are not being registered because of social status and other reasons and are being unnoticed. Cyberbullying also tarnishes the image of a person. It hampers their reputation with the false rumors spread about them. Nearly 8 out of 10 children are being subjected to these various cyberbullying. Cyberbullying detection using machine learning involves using algorithms and statistical models to identify and classify instances of online harassment, abuse, and bullying. This process typicallyinvolves training a model on a large dataset of examples of cyberbullying, along with examples of non- cyberbullying text. The model can then be used to make predictions about whether new instances of online text are likely to be cyberbullying. The features that the model considers when making these predictions can include word usage, sentiment, and other linguistic patterns. The goal of this approach is to provide a fast, scalable, and automated way to detect and address cyberbullying, so that online communities can be safer and more supportive for everyone.

Some common machine learning techniques used in cyberbullying detection include:

Supervised learning: This is the most commonly used approach, where the model is trained on a labeled dataset of examples of cyberbullying and non-cyberbullying text. The model uses this training data to learn patterns in the language that are associated with cyberbullying, and then uses these patterns to make predictions on new, unseen text.

Unsupervised learning: This approach involves training the model on a large dataset of text without explicit labels, and then using clustering or other techniques to identify patterns and group similar instances of text together. This can be used to identify clusters of text that are likely to be cyberbullying, or to uncover other patterns in the data that may be useful for detecting cyberbullying.

Deep learning: Deep learning models, such as neural networks, can be used to automatically learn patterns in the text data, and to make predictions about whether a given instance of text is cyberbullying ornot. This approach can be especially effective when combined with large amounts of training data and powerful computational resources.

The performance of machine learning models for cyberbullying detection can be evaluated in a number of ways, such as accuracy, precision, recall, and F1 score. The choice of evaluation metric will depend on the specific goals and requirements of the task, and may require careful consideration of trade-offs between false positive and false negative rates. In general, the use of machine learning for cyberbullying detection can be a powerful tool for making online communities safer and more supportive for everyone. However, it is important to keep in mind that machine learning models are not perfect, and that there may be biases and limitations in the data and algorithms that can affect the performance and fairness of these systems.

Additionally, it is important to note that the use of machine learning for cyberbullying detection should be part of a larger, comprehensive approach to addressing online harassment and abuse. Machine learning models can help to identify instances of cyberbullying and bring them to the attention of moderators, but they are not a complete solution in and of themselves.

Effective strategies for addressing cyberbullying also need to consider the social and emotional impact ofonline harassment on individuals and communities, as well as the broader cultural and legal context in which cyberbullying occurs. They may also need to involve a range of stakeholders, including educators, parents, law enforcement, and technology companies, to create a safe and supportive online environment for everyone.

Machine learning can be a valuable tool for detecting and addressing cyberbullying, but it should be used in a responsible and thoughtful way, as part of a larger and more comprehensive approach to making online communities safer and more supportive.

It is also important to consider ethical and privacy implications when using machine learning for cyberbullying detection. For example, personal data of users may be collected and processed in order to train and evaluate the models, and it is crucial to ensure that this data is handled in a secure and confidential manner, in accordance with relevant privacy regulations and ethical principles.

Furthermore, there is also a risk that machine learning models may be used to unfairly target and penalize certain individuals or groups, for example, by misidentifying innocent speech as cyberbullying or by disproportionately affecting marginalized or minority communities. To address these concerns, it is important to develop and implement robust fairness and accountability measures, such as regular audits and evaluations of the models, to ensure that they are operating in afair and ethical manner.

Finally, it is also important to recognize that cyberbullying is a complex and evolving problem that is influenced by a wide range of social, cultural, and technological factors. Machine learning models for cyberbullying detection are just one aspect of a larger and more comprehensive approach to addressing this issue, and they need to be used in a way that is aligned with broader goals and strategies for creating a safe and supportive online environment. Previous methods for detecting cyberbullying mainly relied onhuman moderators or rules-based systems. For example, moderators would manually review flagged posts or comments and determine whether they constituted cyberbullying, based on predefined criteria. Rules-based systems, on the other hand, would use a set of predefined rules or keywords to automaticallyidentify instances of cyberbullying.

However, these methods had several limitations, including high levels of subjectivity, the need for significant human effort and resources, and the inability to effectively keep up with the large volume and speed of online content.

In recent years, machine learning has emerged as a more effective and scalable approach to detecting cyberbullying. Machine learning models can automatically learn patterns in the text data that are indicative of cyberbullying, and they can process vast amounts of data quickly and efficiently, allowing for near real-time detection of cyberbullying incidents.

Current methods for using machine learning for cyberbullying detection typically involve training supervised machine learning models on large datasets of labeled examples of cyberbullying and non-cyberbullying text, and then using these models to make predictions about new, unseen instances of online text.

These models can be trained using a variety of techniques, such as logistic regression, decision trees, and neural networks.

Additionally, some current methods also incorporate additional information, such as user demographics and network structure, to enhance the performance of the models and to provide a more comprehensive understanding of cyberbullying incidents. For example, some studies have shown that incorporating network information, such as the relationships between users, can improve the accuracy of cyberbullying detection models.

The relevance of the project to the field of machine learning lies in the use of advanced techniques and models to analyze and classify online text data, in order to detect instances of cyberbullying. Machine learning algorithms have the ability to learn from large amounts of data and make predictions based onpatterns and relationships in that data, making them well-suited for the task of detecting cyberbullying. In recent years, there has been a growing interest in the use of machine learning for the detection ofonline harassment, and cyberbullying in particular. This is due to the increasing prevalence ofcyberbullying and the need for more effective methods to address this issue. The use of machinelearning can help to overcome some of the challenges associated with traditional methods ofcyberbullying detection, such as the lack of scalability and the difficulty of accurately identifyinginstances of abuse in unstructured text data.

The project is also relevant to the field of online harassment because it seeks to develop and evaluate systems for detecting and preventing cyberbullying. The results of the project have the potential to contribute to a better understanding of the dynamics and patterns of online harassment and abuse, and to provide a framework for developing more effective and scalable methods for addressing these issues. Theuse of machine learning has the potential to play a key role in addressing the problem of cyberbullying, andthe project has the potential to make a valuable contribution to the field.

STATEMENT OF PROBLEM

Cyber bullying, defined as the use of digital technology to harass, humiliate, or threaten someone, isa growing problem in today's digital age. With the increasing popularity of social media, instant messaging, and other online platforms, cyber bullying has become more prevalent and can have severe consequences for the victims, including low self-esteem, depression, and even suicide. The traditional methods of detecting and addressing cyber bullying, such as manual moderation by platform administrators or reporting by users, are time-consuming and often ineffective. They also rely heavily on human judgment, which can be prone to biases and errors. The problem of cyber bullying detection using machine learning is to develop a system that can accurately and efficiently identify instances of cyber bullying in online content, such as social media posts and comments, and provide relevant information to support the prevention and intervention of such incidents. The challenge is to develop a system that can distinguish between instances of cyber bullying and other forms of online expression, such as humor, sarcasm, and disagreement, while also considering the cultural, linguistic, and contextual differences of the content. The system must also be able to handle the large volume and velocity of online content, operate in real-time, and maintain privacy and ethical standards.

OBJECTIVES

- Accurately identifying instances of cyberbullying in online communication.
- Classifying the type of cyberbullying being used (e.g., harassment, threats, spreading rumors).
- Determining the severity of cyberbullying to prioritize interventions and responses.
- Predicting the likelihood that an individual will become a victim of cyberbullying tointervene and prevent bullying before it occurs.
- Identifying the individuals involved in cyberbullying incidents, in order to hold themaccountable and provide support to victims.

SCOPE

The scope of the project includes the following key areas:

- Data Collection and Pre-processing: Gathering a dataset of online content and preparing it for analysis, including cleaning, tokenizing, and normalizing the text.
- Feature Engineering: Extracting relevant features from the text data, such as sentiment, keywords, and entities.
- Model Training and Evaluation: Selecting and training machine learning algorithms, such as Convolutional Neural Networks, Naive Bayes, and Support Vector Machines, to detect cyber bullying and evaluating their performance using metrics such as accuracy, precision, recall, and F1 score.
- Deployment and Integration: Implementing the solution in a real-world environment and integrating it into existing systems, such as social media platforms, to detect cyber bullying in real-time.
- User Interaction: Designing an interface for users, such as administrators and moderators, tointeract with the system, view outputs, and make decisions based on the results.

SOFTWARE AND HARDWARE REQUIREMENTS

Software Requirements	:	Keras, TensorFlow, Jupyter Notebook, Visualstudio
Technology Implemented	:	JUPYTER NOTEBOOK
Language Used	:	PYTHON 3.9.3
User Interface Design	:	Html, css, javascriptWeb
Browser	:	Google Chrome
Hardware Requirements	:	Intel i3 processor and above 4GB RAM and 256GBHDD

EXISTING SYSTEM

Even though many systems have been developed till now using different machine learning algorithms like Random Forest, Naive Bayes, and Artificial Neural network, the accuracy of those models is low and the work using those classification techniques is done with the mindset of detecting Cyber Bullying. In the existing System Support Vector Machine (SVM) is used in which accuracy decreases as the amount of data increases. SVM's are not ideal for dealing with frequent language ambiguities typical for cyberbullying. Only keywords or bag words are used for detection. Personality and Emotions are not considered. Limited data availability, false positives and false negatives, contextual ambiguity, bias in training data, scalability, and computational complexity are some of the limitations in the existing system. Cyber bullying detection using machine learning is a crucial task in the modern digital world, as it helps to identify and prevent bullying behaviour in online communication platforms. The following is an overview of the existing systems that use machine learning to detect cyber bullying.

1. **Supervised Machine Learning**: In this type of machine learning, a model is trained on a labelleddataset to identify the type of speech that constitutes bullying. The model then predicts the type of speech in new, unseen data. This approach is often used in sentiment analysis and text classification tasks. Some of the commonly used algorithms for this approach include Naive Bayes,Support Vector Machines (SVM), and Logistic Regression.
2. **Unsupervised Machine Learning**: This approach does not use labelled data to train the model, but instead, the model is trained on the patterns and relationships within the data. Some of the commonly used algorithms for this approach include clustering algorithms such as K-Means and hierarchical clustering, and anomaly detection algorithms such as One-Class SVM.
3. **Deep Learning**: Deep learning algorithms, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), are used to detect cyber bullying based on the semantic meaning of the text. These algorithms are designed to handle large amounts of data and complex relationships between the data. They can learn features from the text that are important for detecting cyber bullying.
4. **Transfer Learning**: Transfer learning is a deep learning technique in which a model trained on a large dataset is fine-tuned for a specific task. In the context of cyber bullying detection, a pre-trained language model, such as BERT, can be fine-tuned for the task of cyber bullying detection. This approachcan improve the performance of the model, as it leverages the knowledge learned from the large pre-trained dataset.
5. **Hybrid Approaches**: Some existing systems use a combination of supervised and unsupervised machine learning, or deep learning and transfer learning, to detect cyber bullying. This approach can improve the performance of the model, as it leverages the strengths of multiple machine learning techniques.

In cyberbullying detection, there are several existing systems for cyber bullying detection using machine learning. Each approach has its strengths and weaknesses, and the choice of which approach to use depends on the specific requirements of the task. However, regardless of the approach used, it is important to note that the performance of these systems can be improved through the use of appropriate feature engineering, data pre-processing, and model fine-tuning techniques.

Techniques like unsupervised labelling methods which use N-gram, TF-IDF methods to detect cyberbullying are used which use the youtube dataset to detect attacks. Another key aspect of existing cyber bullying detection systems is the dataset used for training and evaluating the models. The quality and diversity of the dataset play a crucial role in determining the performance of the models. Some existing systems use annotated datasets, which contain examples of bullying and non-bullying behaviour, to train and evaluate the models. Other systems use unannotated datasets, such as social media posts, and apply unsupervised machine learning techniques to detect bullying behaviour.

One of the challenges faced by existing systems is the changing nature of cyber bullying behaviour. Cyber bullies often use creative and evolving methods to harass their victims, making it difficult for machine learning models to keep up. Additionally, the context in which the bullying behaviour occurs is important in determining whether it is bullying or not. For example, a joke between friends may be misinterpreted as bullying by a machine learning model. To address these challenges, it is important to regularly update the dataset used to train and evaluate the models, as well as to fine-tune the models basedon the changing nature of cyber bullying behaviour.

Another challenge faced by existing cyber bullying detection systems is the need for privacy and security. Many cyber bullying detection systems use social media posts as their source of data, which raises privacyand security concerns. To address these concerns, it is important to use appropriate data anonymization techniques and secure data storage methods.

Finally, it is important to note that cyber bullying detection using machine learning is only one aspect of preventing cyber bullying. The most effective approach is to take a multi-disciplinary approach that combines technological solutions with education, awareness, and policy initiatives.

In conclusion, existing systems for cyber bullying detection using machine learning have made significant advances in identifying and preventing cyber bullying. However, there are still challenges thatneed to be addressed, such as the changing nature of cyber bullying behaviour, privacy and security concerns, and the need fora multi-disciplinary approach. Nevertheless, the use of machine learning to detect cyber bullying is a promising and effective approach, and further research and development in this area is expected to lead to even better results.

It is also important to note the ethical considerations when using machine learning for cyber bullying detection. One of the main concerns is the potential for false positives, where a machine learning model may incorrectly label a statement as bullying. This can have serious consequences for the person wrongly accused of bullying, including social and professional consequences. To minimize the risk of false positives, it is important to use robust evaluation methods, such as cross- validation and test-time augmentation, and to have human annotators validate the results of the models.

Another ethical concern is the potential for bias in the models. Machine learning models are only as good as the data they are trained on, and if the training data contains biases, these biases will be reflected in the models. This can result in unfair treatment of certain groups, such as minority groups or those with different political views. To address this, it is important to use diverse and representative datasets and to monitor the models for potential biases.

Finally, the use of machine learning for cyber bullying detection raises privacy concerns. Social media platforms, in particular, collect a vast amount of personal information, which can be used to detect bullying behaviour. However, this raises concerns about data privacy and data protection. To address these concerns, it is important to use secure data storage and transfer methods, and to use data anonymization techniques where appropriate.

In conclusion, the use of machine learning for cyber bullying detection is a complex issue that requires a thorough understanding of the technical, ethical, and legal considerations. While machine learning has the potential to greatly improve the ability to detect and prevent cyber bullying, it is important to approach this issue with caution and to carefully consider the potential consequences of using these technologies.

SOFTWARE REQUIREMENTS SPECIFICATION

Introduction:

1. Purpose: Explain the purpose and scope of the project.
2. Objectives: Describe the main objectives and goals of the project.
3. Overview: Give a brief overview of the system and its functionalities.

FunctionalRequirements:

1. Input: Specify the types of inputs that the system should accept.
2. Processing: Define the machine learning algorithms to be used for detection of cyber bullying.
3. Output: Outline the types of outputs that the system should provide.

Non-FunctionalRequirements:

1. Performance: Specify the performance requirements such as response time and accuracy.
2. Security: Discuss the security measures to be implemented to protect sensitive data.
3. Usability: Describe the user-friendliness of the system and ease of use.

TechnicalRequirements:

1. Hardware: List the hardware requirements for the system, including server specifications andstorage requirements.
2. Software: Specify the software requirements, including programming languages andlibraries.
3. Infrastructure: Describe the network infrastructure and connectivity requirements.

User Requirements:

1. User roles: Define the different user roles and their access levels.
2. User interface: Describe the user interface, including the layout and design.
3. User interactions: Specify the ways in which users will interact with the system.Acceptance Criteria:

1. Define the criteria that must be met for the system to be considered complete and ready for release.
2. Include functional, performance, and security requirements.

Deployment:

1. Explain the steps to deploy the system in a production environment.
2. Include any necessary training or support for users.
3. Maintenance and Support:
4. Discuss the maintenance and support requirements, including bug fixing and updating.
5. Include a plan for future development and enhancements

PROPOSED SYSTEM

Working Methodology:

Building a machine learning model to identify cyber bullying requires several steps:

Data Collection: Collect a large and diverse dataset of text (such as social media posts or online comments) that contain instances of cyber bullying and non- cyber bullying examples.

Our data set consists of text in both English and Hinglish. tweets in total, which were manually labeled as toxic or non-toxic by human annotators. For the Hinglish dataset, we collected real-time tweets and Whatsapp messages. The Hinglish dataset has approximately 9,482 entries in total, which were also manually labeled as toxic or non-toxic. networking platforms. It has approximately 15,307 We extracted tweets from various social media platforms like Twitter chats, and YouTube comments for Hinglish dataset. The dataset has approximately 3000 entries. We combined them to create a large dataset. Tweets and Label are the two columns of the dataset. The label contains the numbers 0, 1 , 2, which represent bulllying, non bullying and offensive respectively. The collection contains real samples of messages extracted from various social media platforms and networking websites. It has a large variety of negative words that are generally used by people in their messages. This would help us in detecting every offensive context of tweet. The next step is to preprocess the data after it has been extracted. It is performed because real-time data contains a large number of unwanted characters, necessitating cleaning of data in order to prepare the data for the detection . This is a time-consuming yet critical task.

Data Preparation

Before running several ML models, the data must be cleaned, Cleaning the data and removing unnecessary characters, symbols, and stopwords is an essential step in preparing the data for ML models. It helps to improve the accuracy of the model and prevent overfitting. It's great to see that you have taken the necessary steps to clean the data and remove unwanted words and symbols. It's also important to note that using a manual list of stopwords can be useful, but there are also pre-built libraries and packages available that can automatically remove stopwords based on the language. These packages can save time and improve accuracy in the cleaning process.

Data Preprocessing: Clean the data by removing any irrelevant information, such as URLs, and perform any necessary text normalization, such as converting all text to lowercase.

Preprocessing Methodologies

We used natural language processing techniques after cleaning the data because the algorithm cannot perform directly with unprepared text, that is, it cannot understand the sentences given to it, so we transform these sentences into understandable format using some preprocessing techniques. We use the following

• Tokenization - Tokenization is the process of dividing a text sequence into smaller parts, which include phrases, words, concepts, and symbols known as tokens. Tokenization is a fundamental step in natural language processing and is often performed as a pre-processing step before text data is fed into machine learning models. The goal of tokenization is to break down text into smaller units that can be analyzed and processed more easily, allowing for more accurate natural language processing tasks such as sentiment analysis, part-of-speech tagging, and text classification. Tokenization can be performed at different levels, including word-level, sentence-level, and document-level.

• Lemmatization - Lemmatization is the process of converting a word to its base or dictionary form, known as the lemma. It is an essential step in natural language processing that helps reduce the number of distinct words and variations in a text document. For example, the lemma of the word "running" is "run," and the lemma of the word "better" is "good."

• Vectorization - Vectorization is the process of converting text into numerical vectors that can be processed by machine learning algorithms. In NLP, vectorization is used to represent words or documents as numerical vectors, where each dimension of the vector corresponds to a specific feature. The values in the vector indicate the importance or weight of each feature.

. **Data segmentation:**

The dataset is divided into two types of data that is training data and testing data. For the model to be used in real time, the testing data must be retrieved from the various platforms using text mining. Both datasets are preprocessed and subjected to multiple ML models.

Feature Engineering: Decide on the features that the model will use to make its predictions. In the case of text classification, this usually involves creating numerical representations of the text, such as using term frequency-inverse document frequency (TF-IDF) or word embeddings.

Feature choice:

We prepare the relevant text features after separating the data. This technique aids in determining the quality of the produced vector representations. This works with related words that tend to close with terms with varying degrees of resemblance. Vectorization is done before running the training and testing data sets through the ML models.

1) **Count Vectorization**: Count Vectorization is a technique in Natural Language Processing (NLP) that converts a collection of text documents into a matrix of token counts. In this technique, a count of each word occurrence in the text is calculated, and then the word occurrences are transformed into numerical values.

The process of Count Vectorization involves the following steps:

- Tokenization: The text is first split into individual words or tokens.
- Counting: The number of times each token appears in the document is counted.
- Vectorizing: The counts are transformed into a sparse matrix of integers, where each row represents a document and each column represents a unique token in the corpus.

Count Vectorization is a common pre-processing step used in various NLP tasks such as text classification, topic modeling, and information retrieval. It allows text data to be analyzed using machine learning algorithms by representing the text as a numeric matrix.

2) **TF-IDF**: TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a numerical statistic that reflects how important a word is to a document in a collection or corpus. It is commonly used in natural language processing and information retrieval for text analysis and search engine ranking.

TF-IDF takes into account both the frequency of a word in a document and the frequency of the word in the entire corpus. The term frequency (TF) of a word is the number of times the word appears in a document, divided by the total number of words in the document. The inverse document frequency (IDF) of a word is calculated as the logarithm of the total number of documents in the corpus, divided by the number of documents that contain the word.

The TF-IDF score for a word in a document is calculated by multiplying the term frequency of the word in the document by the inverse document frequency of the word. Words with higher TF-IDF scores are considered to be more important to the document, and are therefore given higher weight in analysis and ranking.

Model Selection: Choose an appropriate machine learning algorithm for the task of text classification., such as a support vector machine (SVM), a neural network, or a Logistic Regression.

Models Used:

We have used following models initially then we used F1 score to check which model have high accuracy.

1) SVM (Support Vector Machine):

Support Vector Machines (SVMs) are a type of supervised learning algorithm used for classification and regression analysis. SVMs are particularly useful for classification tasks where the data points are not easily separable by a linear boundary.

In SVM, a hyperplane is used to separate the data points of different classes. The goal is to find the hyperplane that maximizes the margin between the closest data points of different classes. These closest points are known as support vectors. SVMs use a kernel function to transform the data into a higher-dimensional space, which can help to find a better separating hyperplane.

SVMs are known for their ability to handle high-dimensional data and perform well on small to medium-sized datasets. They have been successfully applied to a wide range of applications, including image classification, text classification, and bioinformatics.

2) KNN (K Neighbors): KNN (K Nearest Neighbors) is a non-parametric classification algorithm that works by finding the k-nearest data points to a given data point in the feature space. It then classifies the data point based on the majority class of its k-nearest neighbors. The choice of k is an important parameter, as a lower value of k may lead to overfitting while a higher value may result in underfitting. KNN is a simple yet effective algorithm for classification and can be used for both binary and multi-class classification problems**.**

.**3) Logistic Regression:** Logistic regression is a popular machine learning algorithm used for binary classification problems, where the target variable has only two possible outcomes. In logistic regression, the goal is to find the best fitting S-shaped curve, also known as the sigmoid function, that maps the input features to the target variable. The output of the sigmoid function is interpreted as the probability of the input belonging to the positive class.

The logistic regression model uses the maximum likelihood estimation technique to estimate the parameters of the sigmoid function. The model tries to find the values of the coefficients that maximize the likelihood of the observed data given the input features. The optimization problem is solved using gradient descent or other optimization algorithms.

Logistic regression is a simple and efficient algorithm that is easy to interpret and has low computational overhead. It can handle non-linear relationships between the input features and the target variable by adding polynomial or interaction terms to the model. However, logistic regression assumes that the input features are independent and linearly related to the target variable, which may not always be the case in practice.

4) Decision Tree Learning: In research, decision trees can be a useful approach for detecting cyberbullying. A decision tree is a form of supervised learning technique that may be used to categorize data into several groups. Decision trees can be used in cyber bullying detection to examine aspects of online content (such as text, photos, or videos) and classify them as instances of cyber bullying or non-cyber bullying. If the model's performance is inadequate, you may need to refine it by adjusting its settings or redefining its characteristics. This iterative procedure will assist you in developing a decision tree model that is successful at detecting cyberbullying.Overall, decision trees can be a valuable technique for detecting cyberbullying in studies.

Decision trees can help identify instances of cyberbullying and aid in the creation of effective intervention techniques by examining aspects of online content.

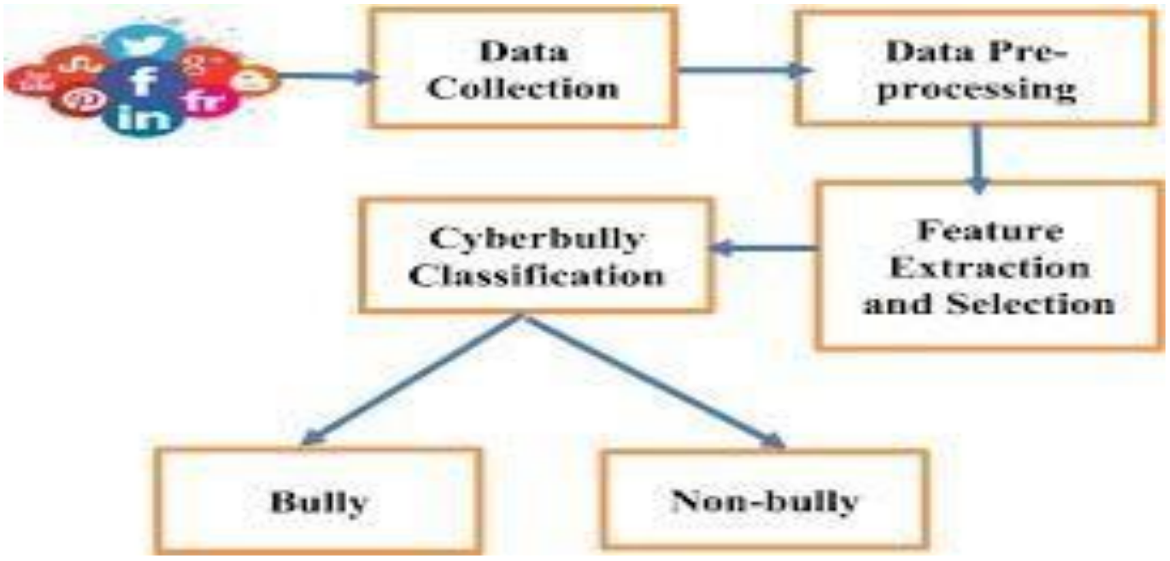


Fig.Detection model

Model Training: Train the model on the preprocessed and feature-engineered data. This involves providing the model with the features and corresponding labels (e.g., cyber bullying or non-cyber bullying) for each instance in the training data.

Ensemble Method :

Ensemble learning is a general meta approach to machine learning that seeks better predictive performance by combining the predictions from multiple models.

Although there are a seemingly unlimited number of ensembles that you can develop for your predictive modeling problem, there are three methods that dominate the field of ensemble learning. So much so, that rather than algorithms per se, each is a field of study that has spawned many more specialized methods.The three main classes of ensemble learning methods are **bagging**, **stacking**, and **boosting**, and it is important to both have a detailed understanding of each method and to consider them on your predictive modeling project.

We have trained the model using above cllasifiers,As we are using ensemble algorithms we are going to train the model using each and every algorithm.After training them we are going to use the ensemble method called Max Voting.

Max Voting :

It is mainly used for classification problems. The method consists of building multiple models independently and getting their individual output called 'vote'. The class with maximum votes is returned as output.

In the below example, three classification models (logistic regression, xgboost, and random forest) are combined using sklearn VotingClassifier, that model is trained and the class with maximum votes is returned as output. The final prediction output is pred_final. Please note it’s a classification, not regression, so the loss may be different from other types of ensemble methods.

```
from sklearn.ensemble import VotingClassifier
VotingClassifier(estimators=[('lr', model_1), ('xgb', model_2), ('rf', model_3)], voting='hard')
```

Model Evaluation: Evaluate the performance of the model by testing it on a held- out test set and computing metrics such as accuracy, precision, recall, and F1 score.

To evaluate the accuracy for every model in the second phase. We used F1 score for evaluation and improved the accuracy of the model by recurring the various stages. We thought of finding the best suitable combination of dimension selection methods like and count vectorizers , TF-IDF and machine learning models. For completing this, we have compared both TF-IDF and count vectorizer; based on the comparison we got, we finalized the best pair with high accuracy and low prediction time and created its pickle file. After that, we fed the test data to the models inorder to compare the accuracy of various algorithms. using these our model will predict whether the text read belongs to the context of bullying or no in English.

Model Fine-tuning:

If necessary, fine-tune the model by adjusting its parametersor trying a differentalgorithm, and then repeat the evaluation step.As there is still improvement for the accuracy we have changed the classifiers and used various combinations of them to improve the accuracy of the model.we got 2 % increase in the accuracy of the model by using random forests decision tree and linear regression.

ALGORITHM USED

The Algorithm that will be implemented for working of this proposed system isas follows:

Step 1: Import all the modules required such as numpy, pandas, matplotlib, sklearn, Tensorflow.

The following modules have to be imported form the skikitlearn, If those modules are not installed then we have to install them first then we have to import them otherwise we might get errors.These modules are imported as follows,

```
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 xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import VotingClassifier
from sklearn.ensemble import RandomForestClassifier
import re
import nltk
```

Here we are using nltk lilbrary available in skikit for Natural language processing.

Step 2: Load the cyberbullying dataset and extract the attributes.

We have pulled real-time tweets from Twitter, Whatsapp chats, and YouTube comments in both English and Hinglish. Pre-processing text data is an important step in natural language processing to clean up and standardize the data.

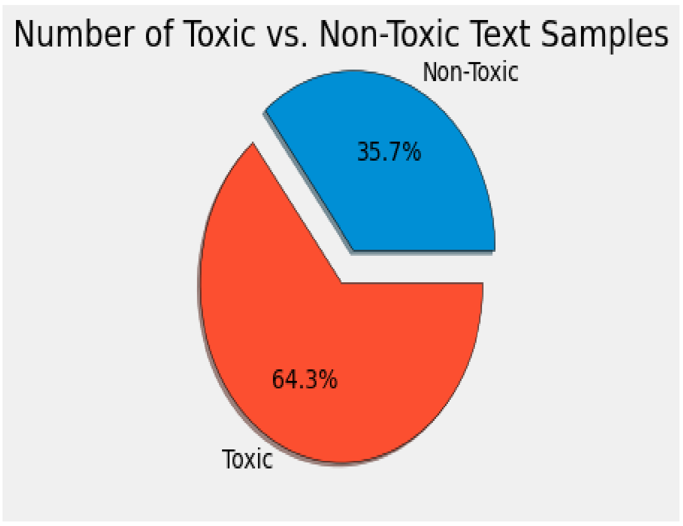
Dataset:

The dataset is the snapshot of the twitter information during a particular period of time .

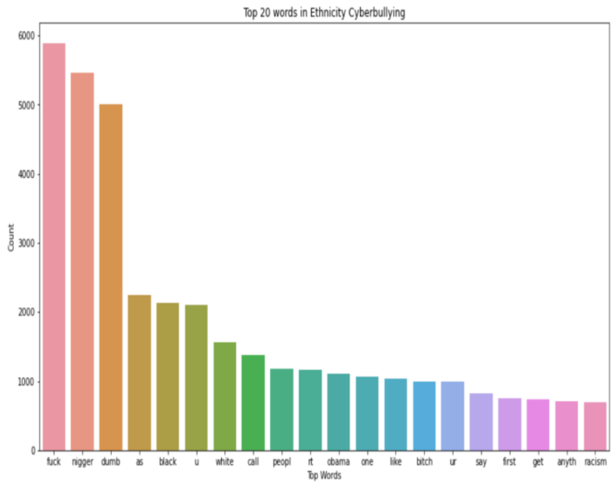
It consists of tweets from all kinds of users .It has seven columns na--mely 'Unnamed,'count', 'hate_speech' , 'offensive_language ', 'neither', 'class', 'tweet '.

The class is our dependent variable and remaining

are the independent varibles .The class consists of numeric values so we would be converting them into discrete values by labelling them.The dataset have nearly 24000 rows .we would be partitioning it for testing and training .It does not have any missing values or inconsistent values .The tweets is the most important variable as we are going to use it for classifying the comments.The class is our target variable one which will be predicted.



Top 20 words used in cyberbullying tweets:



We are going to download the dataset using read_csv() method.

Step 3: Perform the necessary preprocessing techniques and analyze the data.

We have pulled real-time tweets from Twitter, Whatsapp chats, and YouTube comments in both English and Hinglish. Pre-processing text data is an important step in natural language processing to clean up and standardize the data. It's common to remove superfluous characters, such as emojis, special characters, and punctuation, as they can interfere with the analysis and modelling process., we must clean and prepare the data for detection before applying machine learning algorithms to it. During the preprocessing stage we have removed various unwanted patterns like hexadecimal patterns, stopwords, numerical Characters, Hashtags and other special symbols. It is accomplished by utilizing numpy and vectorize routines. We have manually come up with a list of stopwords in those languages that are taken and used it to delete them.
These terms should be removed from the clean data because their existence reduces the model's accuracy and predictions. We then used NLP techniques such as tokenization to break raw text into words known as tokens, lemmatization to reduce a given word to its core word, and vectorization to convert raw text into vectors or numbers.

Step 4: Split the dataset into training and testing data.

After completion of pre-processing, we have divided the dataset into two parts such as testing data and training data. After that, we have used the two most significant text selection functions, which are:

1. Inverse frequency
2. Vectorizer of Counts

Step 5: Perform training the machine learning model using Ensemble learning, Naïve Bayes, SVM, K Nearest Neighbors, Logistic Regression.

We have trained the model using above classifiers, As we are using ensemble algorithms we are going to train the model using each and every algorithm. After training them we are going to use the ensemble method called Max Voting.

Max Voting :

It is mainly used for classification problems. The method consists of building multiple models independently and getting their individual output called 'vote'. The class with maximum votes is returned as output.

In the below example, three classification models (logistic regression, xgboost, and random forest) are combined using sklearn VotingClassifier, that model is trained and the class with maximum votes is returned as output. The final prediction output is pred_final. Please note it's a classification, not regression, so the loss may be different from other types of ensemble methods.

```
from sklearn.ensemble import VotingClassifier
VotingClassifier(estimators=[('lr', model_1), ('xgb', model_2), ('rf', model_3)], voting='hard').
```

Step 6: Compare the accuracies of the algorithms and use the best algorithm for prediction.

To evaluate the accuracy for every model in the second phase. We used F1 score for evaluation and improved the accuracy of the model by recurring the various stages. We thought of finding the best suitable combination of dimension selection methods like and count vectorizers , TF-IDF and machine learning models. For completing this, we have compared both TF-IDF and count vectorizer; based on the comparison we got, we finalized the best pair with high accuracy and low prediction time and created its pickle file. After that, we fed the test data to the models inorder to compare the accuracy of various algorithms. using these our model will predict whether the text read belongs to the context of bullying or no in English.

Step 7: Predict the new data using the algorithm which obtained better accuracy.

PROJECT DESIGN

Use case diagram:

A use case diagram is a type of Unified Modeling Language (UML) diagram that represents the functionality offered by a system to its users. It is used to mode the relationships between the system and its actors (users or external systems) and to describe the interactions between the actors and the system. A use case diagram provides a high-level view of the functional requirements of a system and helps to communicate the system's capabilities to stakeholders.

A use case diagram consists of:

Actors: Represented as stick figures, actors are the entities that interact with the system, such as a human user or another system.

Use cases: Represented as oval shapes, use cases describe the functional requirements of the system and what the system can do for its actors.

Associations: Represented as lines connecting actors and use cases, associations indicate the relationships between the actors and the use cases they participate in.

System boundary: Represented as a rectangle, the system boundary encloses the use cases and actors and defines the scope of the system.

The use case diagram helps to identify the functional requirements of a system and to ensure that the system meets the needs of its users. It is used during the requirements gathering and analysis phase of the software development process to provide a visual representation of the system's capabilities and to facilitate communication between stakeholders.

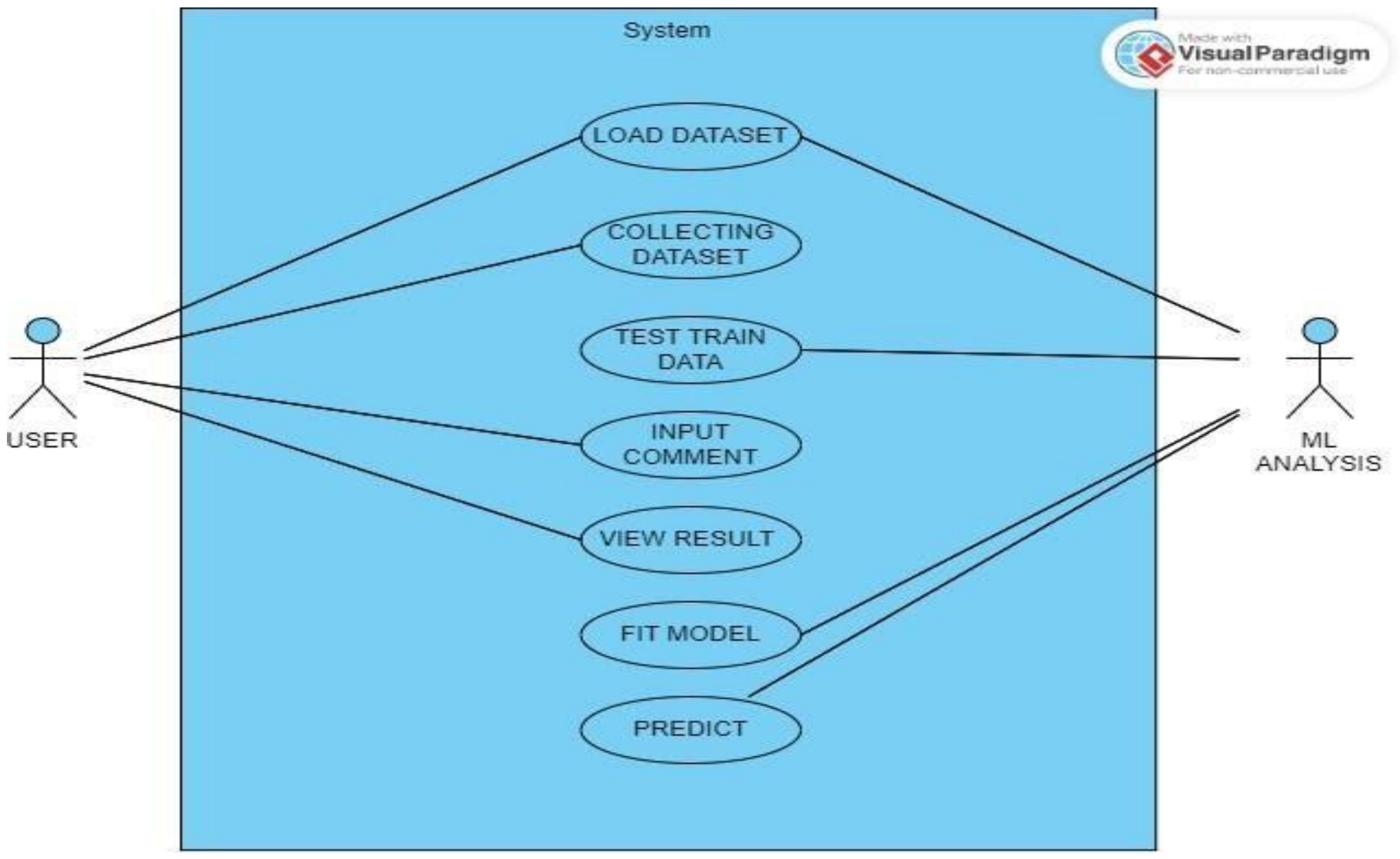


Fig. Use Case Diagram

Sequence diagram:

A sequence diagram is a type of Unified Modeling Language (UML) diagram that represents the interactions between objects or components in a system over time. It provides a visual representation of the interactions between objects and the order in which they occur, helping to describe the behavior of a system and to communicate it to stakeholders.

A sequence diagram consists of:

Objects: Represented as rectangles, objects represent instances of classes and represent the entities that interact with each other in the system.

Lifelines: Represented as vertical lines, lifelines represent the existence of objects over time and the interactions they participate in.

Messages: Represented as arrows, messages represent the interactions between objects and describe the behavior of the system.

Activations: Represented as rectangles on lifelines, activations represent the processing time of an object during an interaction.

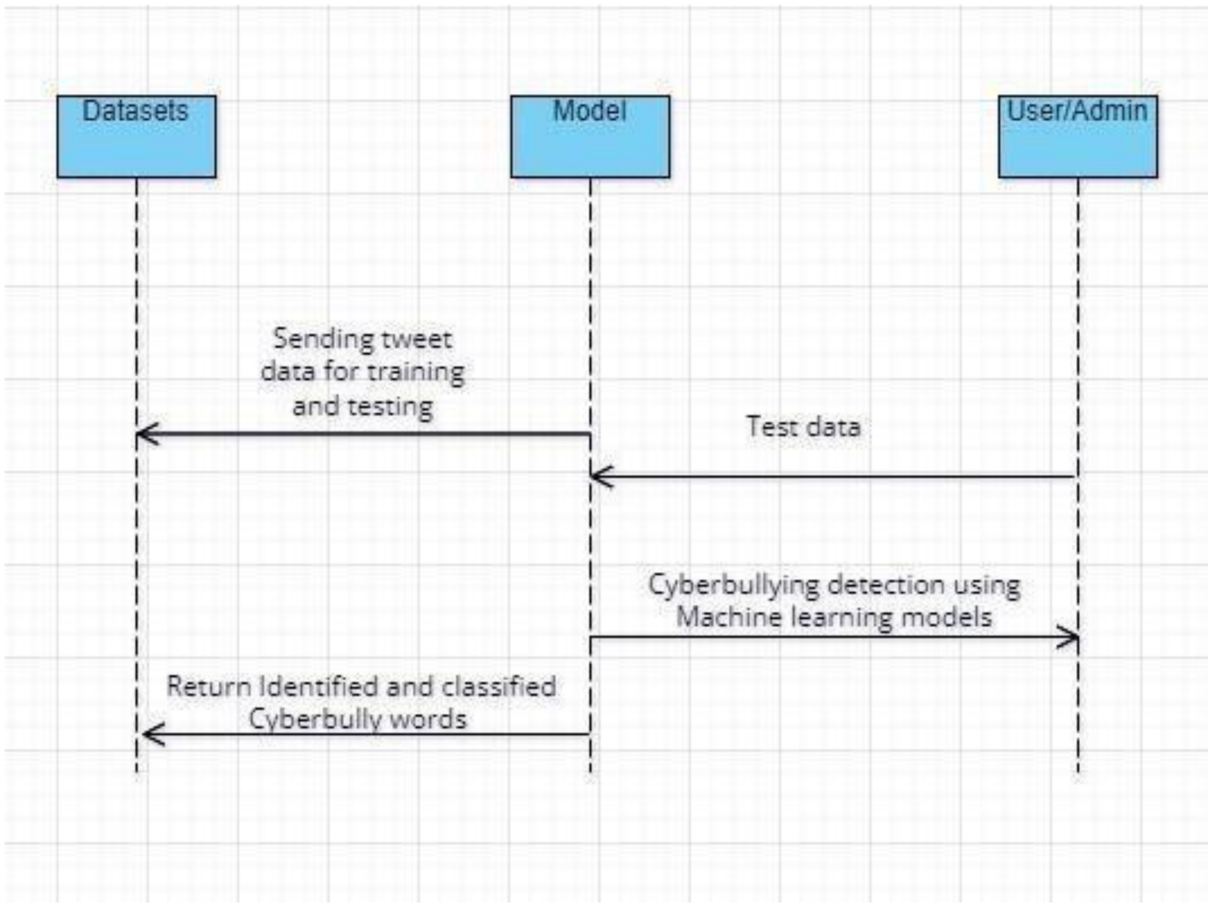

Datasets
Model
User/Admin
Sending tweet data for training and testing
Test data
Cyberbullying detection using Machine learning models
Return Identified and classified Cyberbully words

Fig. Sequence Diagram

State Chart Diagram:

A state chart diagram, also known as a state machine diagram or state diagram, is a type of Unified Modeling Language (UML) diagram that represents the behavior of a system as a series of states and the events that cause the system to transition from one state to another. It provides a visual representation of thebehavior of a system and helps to communicate the behavior to stakeholders.

A state chart diagram consists of:

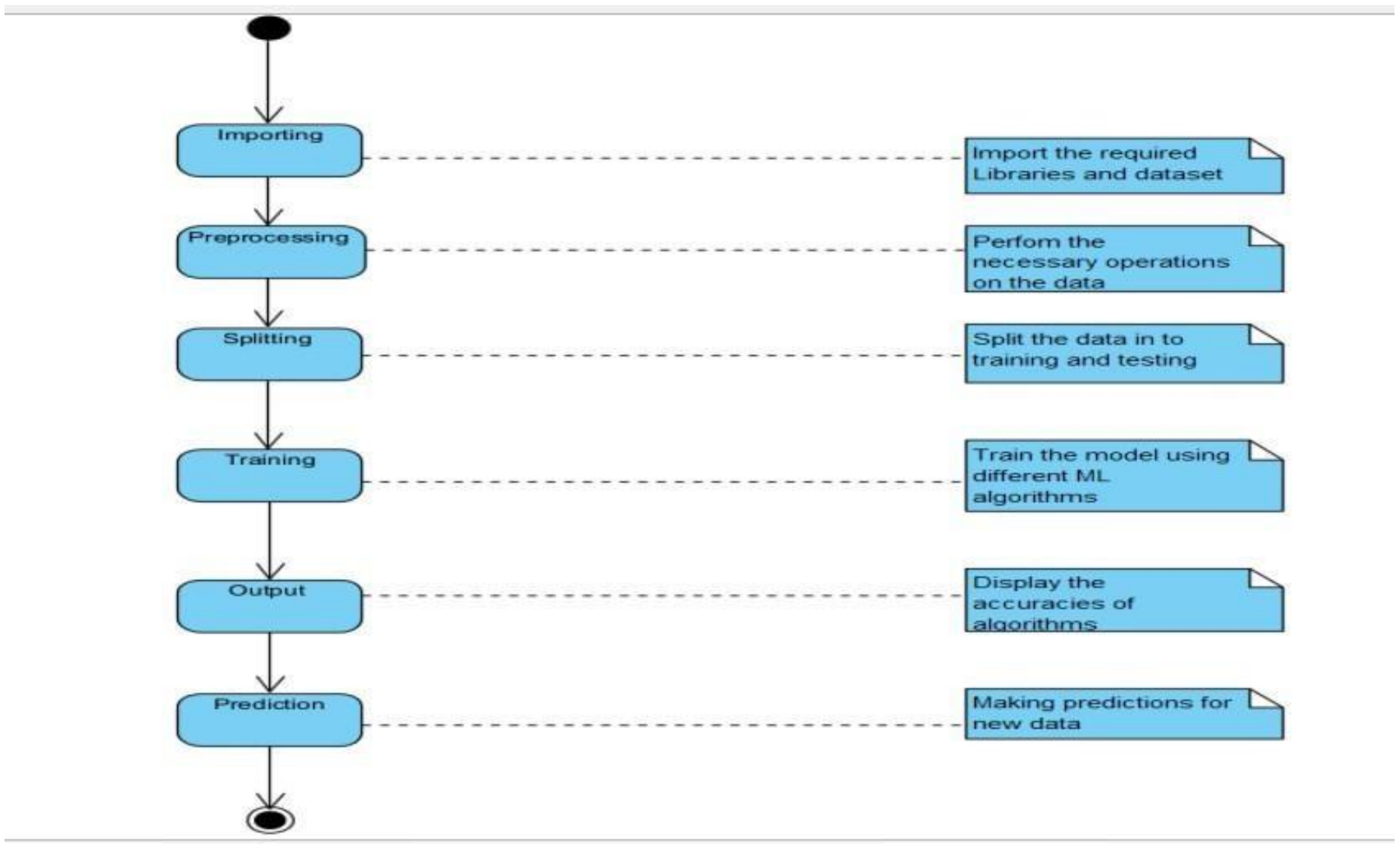
States: Represented as rectangles, states represent the different conditions or phases that a system can be in.

Transitions: Represented as arrows, transitions represent the events that cause a system to change from one state to another.

Events: Represented as ovals, events represent the external or internal triggers that cause transitions to occur.

Actions: Represented as short horizontal lines with text, actions describe the activities that take place during a transition.

A state chart diagram helps to describe the behavior of a system and to understand how a system reacts to events. It is used to model the dynamic behavior of a system and to communicate the behavior to stakeholders. The diagram is particularly useful in describing complex behavior and can be used to model the behavior of real-time systems, embedded systems, and user interfaces.


Importing
Import the required Libraries and dataset
Preprocessing
Perfom the necessary operations on the data
Splitting
Split the data in to training and testing
Training
Train the model using different ML algorithms
Output
Display the accuracies of algorithms
Prediction
Making predictions for new data

Data Flow Diagram:

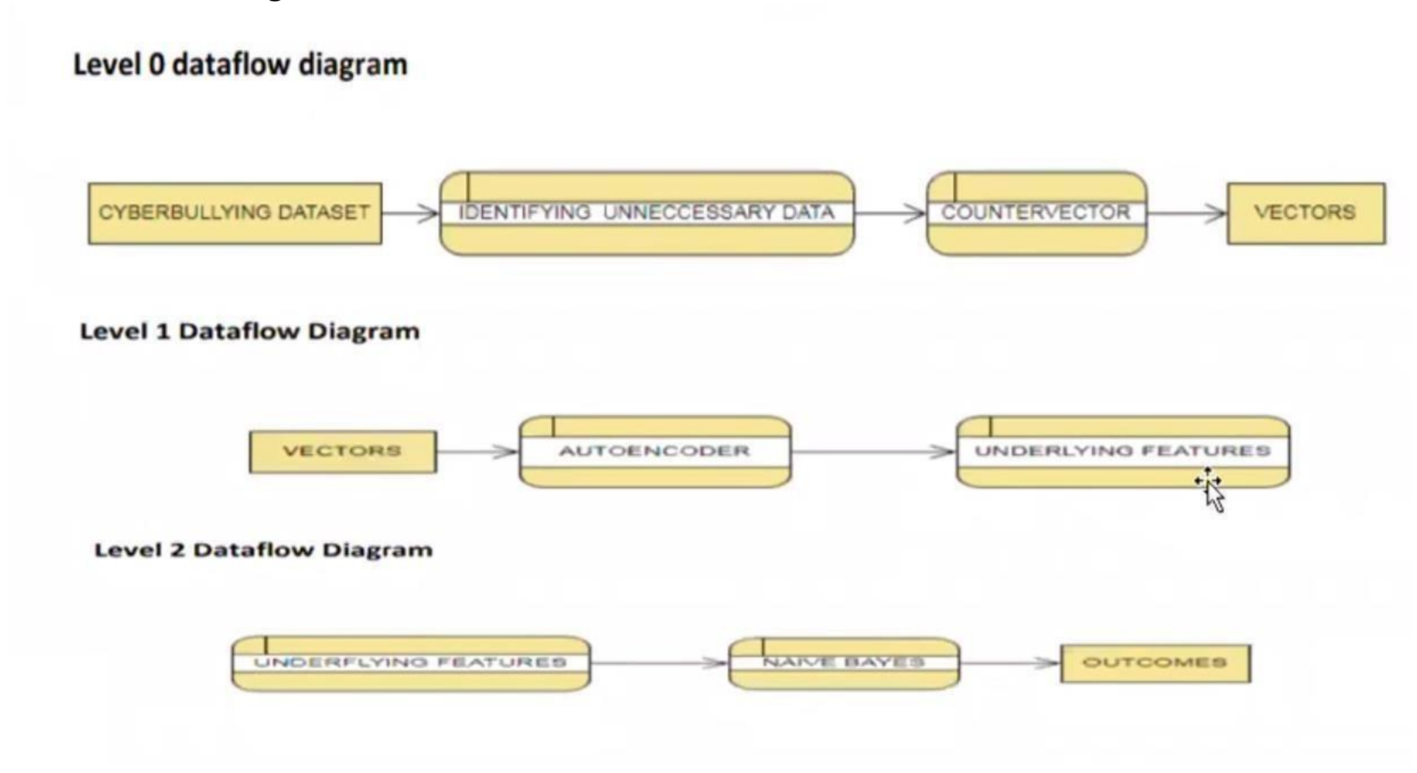
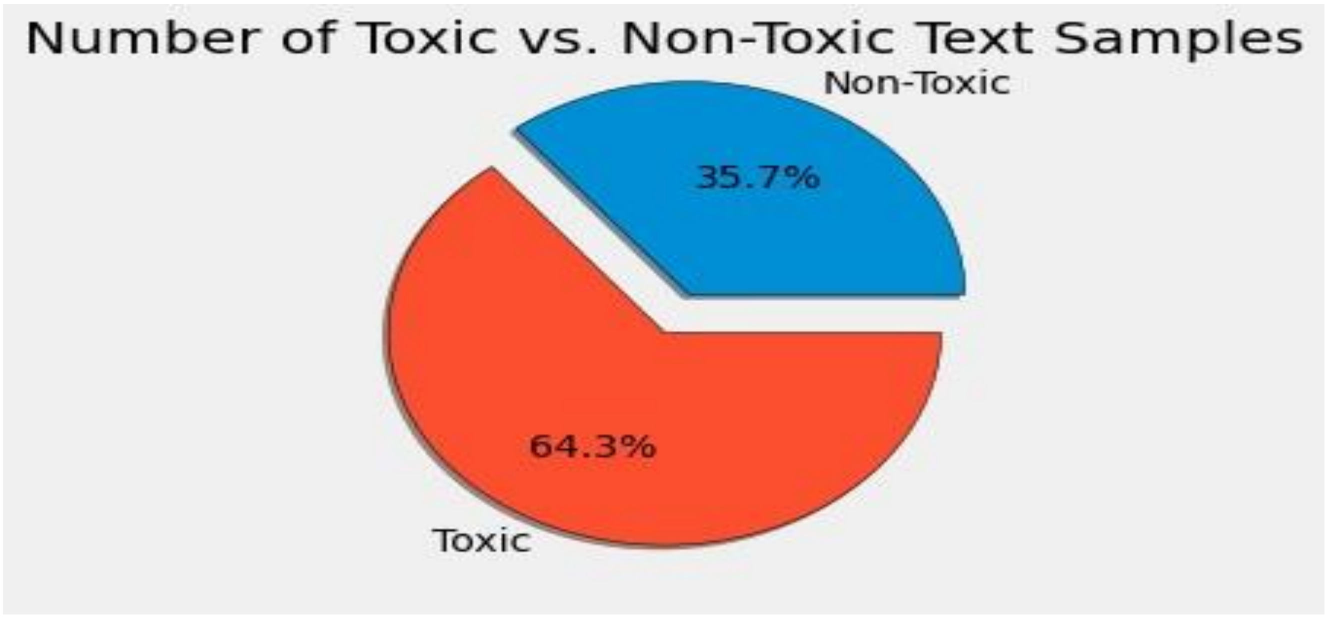


Fig. Data Flow Diagram

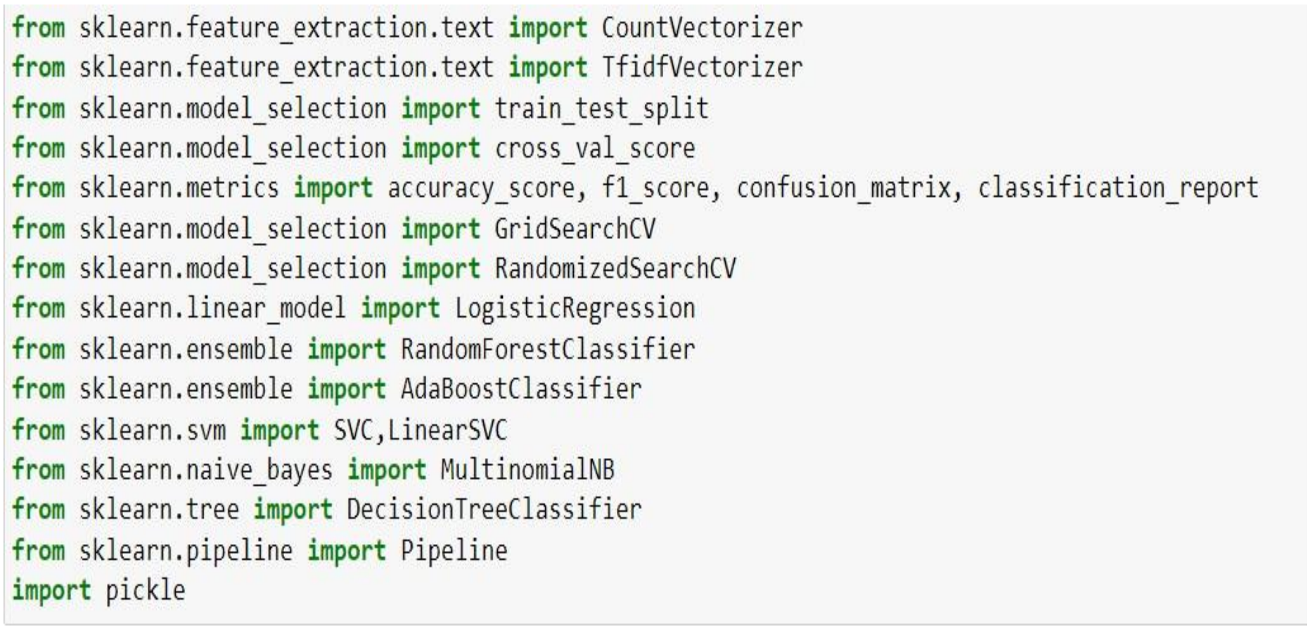
PROJECT CODE

DATASET DESCRIPTION:

The dataset is the snapshot of the twitter information during a particular period of time .
It consists of tweets from all kinds of users .It has seven columns na--mely 'Unnamed,'count', 'hate_speech' , 'offensive_language ', 'neither', 'class', 'tweet '.
The class is our dependent variable and remaining
are the independent varibles .The class consists of numeric values so we would be converting them into discrete values by labelling them.The dataset have nearly 24000 rows .we would be partitioning it for testing and training .
It does not have any missing values or inconsistent
values .
The tweets is the most important variable as we are going to use it for classifying the comments .The class is our target variable one which will be predicted.

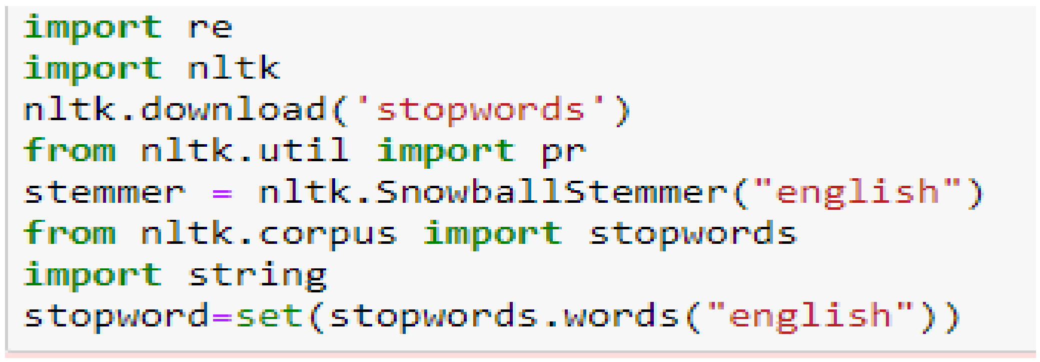


IMPORTING LIBRARIES:



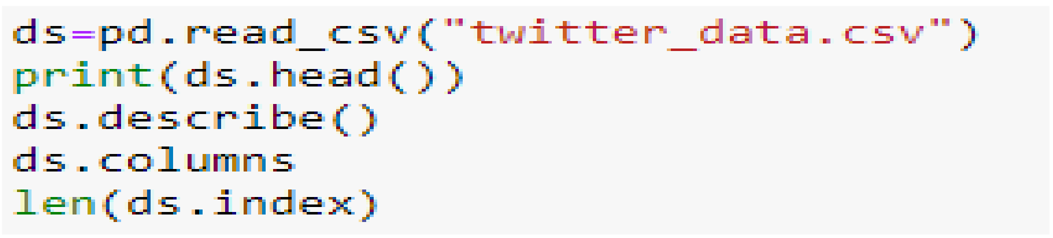
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classification_report
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.svm import SVC,LinearSVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.pipeline import Pipeline
import pickle
```

IMPORTING AND DOWNLOADING NLP TOOLS:

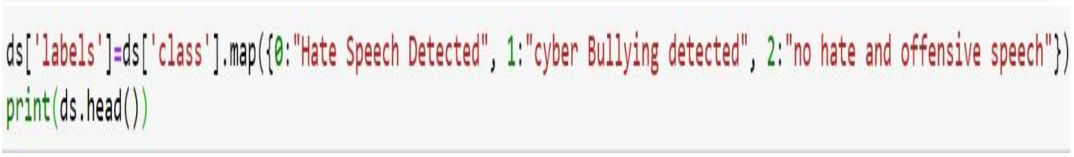


```
import re
import nltk
nltk.download('stopwords')
from nltk.util import pr
stemmer = nltk.SnowballStemmer("english")
from nltk.corpus import stopwords
import string
stopword=set(stopwords.words("english"))
```

LOADING DATASET:

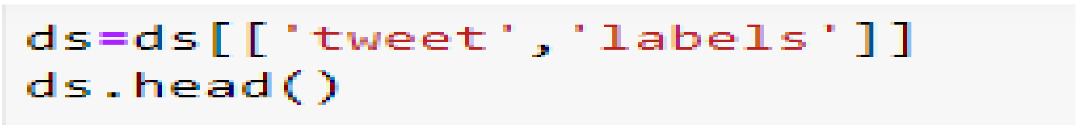


```
ds=pd.read_csv("twitter_data.csv")
print(ds.head())
ds.describe()
ds.columns
len(ds.index)
```



```
ds['labels']=ds['class'].map({0:"Hate Speech Detected", 1:"cyber Bullying detected", 2:"no hate and offensive speech"})
print(ds.head())
```

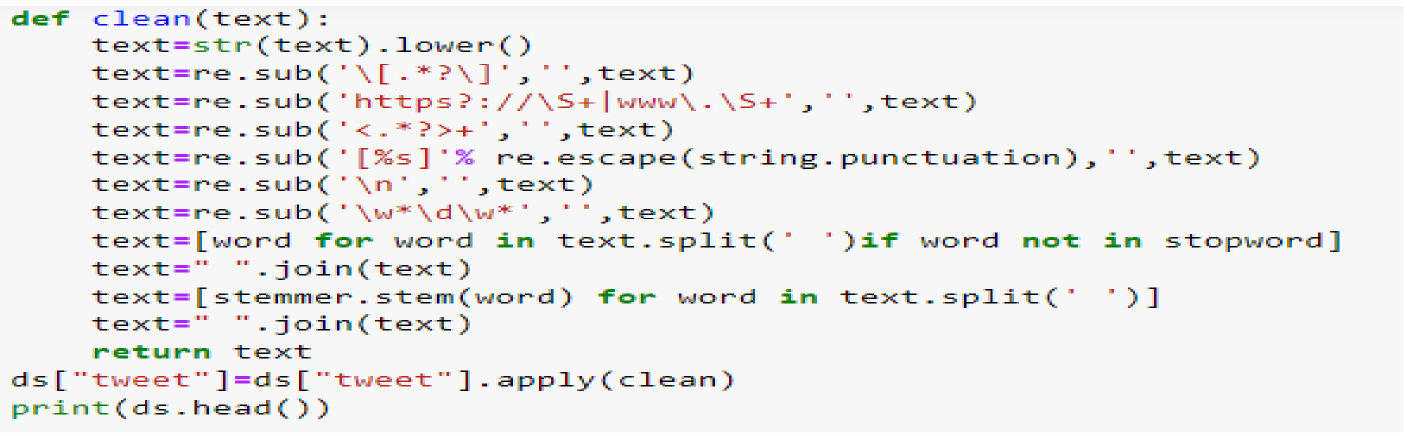
Here, we would be converting the target variable into a discrete variable as we would be working with decision tree classifier.



```
ds=ds[['tweet','labels']]
ds.head()
```

Here, we would be visualizing the changes we have made.

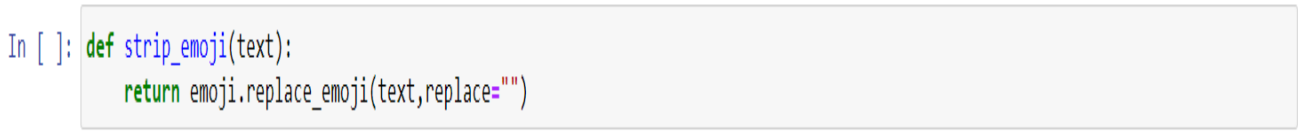
USER DEFINED FUNCTIONS:



```
def clean(text):
    text=str(text).lower()
    text=re.sub('\[.*?\]','',text)
    text=re.sub('https?://\S+|www\.\S+','',text)
    text=re.sub('<.*?>+','',text)
    text=re.sub('[%s]'% re.escape(string.punctuation),'',text)
    text=re.sub('\n','',text)
    text=re.sub('\w*\d\w*','',text)
    text=[word for word in text.split(' ')if word not in stopword]
    text=" ".join(text)
    text=[stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text
ds["tweet"]=ds["tweet"].apply(clean)
print(ds.head())
```

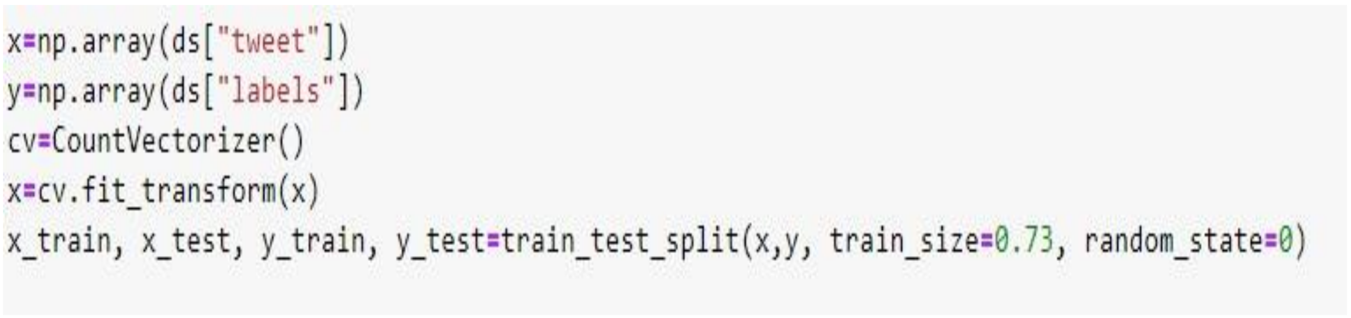
The inbuilt function clean is used to remove the unwanted gibberish or symbols from the tweets that are in the dataset.

Function to remove emoji's in text:



```
In [ ]: def strip_emoji(text):
            return emoji.replace_emoji(text,replace="")
```

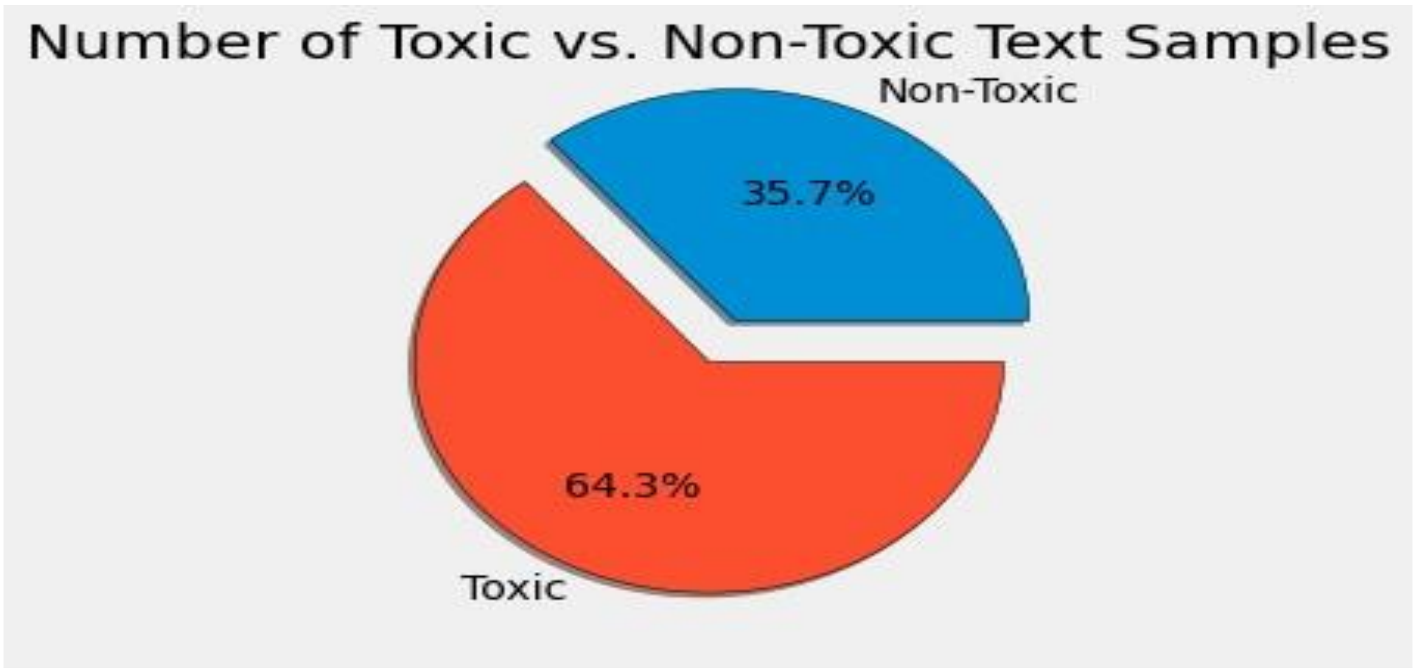
SPLITTING THE DATASET:



```
x=np.array(ds["tweet"])
y=np.array(ds["labels"])
cv=CountVectorizer()
x=cv.fit_transform(x)
x_train, x_test, y_train, y_test=train_test_split(x,y, train_size=0.73, random_state=0)
```

RESULTS

Number of Toxic vs Non toxic samples in our dataset:

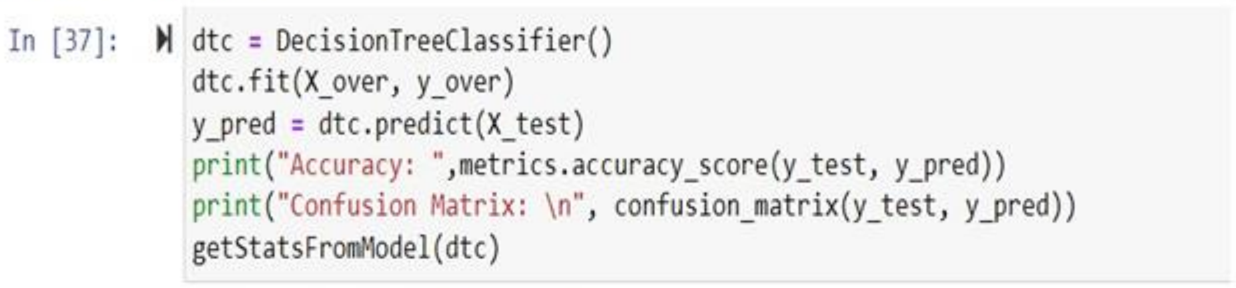


CREATION AND TRAINING OF DIFFERENT MODELS:

Decision Tree Classifier:

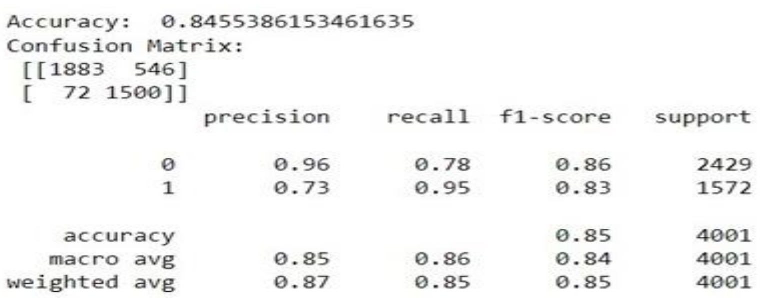
Input:

Decision Tree Classifier:



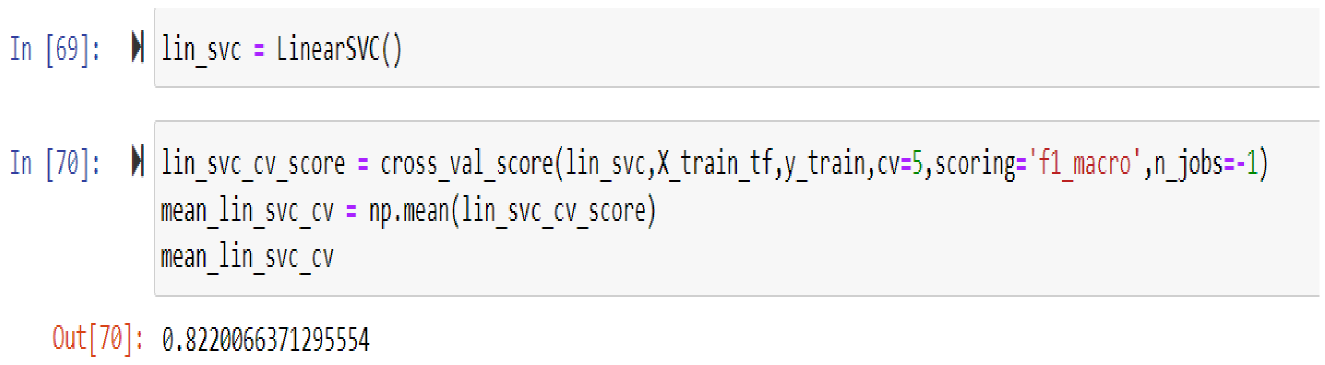
```
In [37]:  dtc = DecisionTreeClassifier()
           dtc.fit(X_over, y_over)
           y_pred = dtc.predict(X_test)
           print("Accuracy: ",metrics.accuracy_score(y_test, y_pred))
           print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
           getStatsFromModel(dtc)
```

Output:



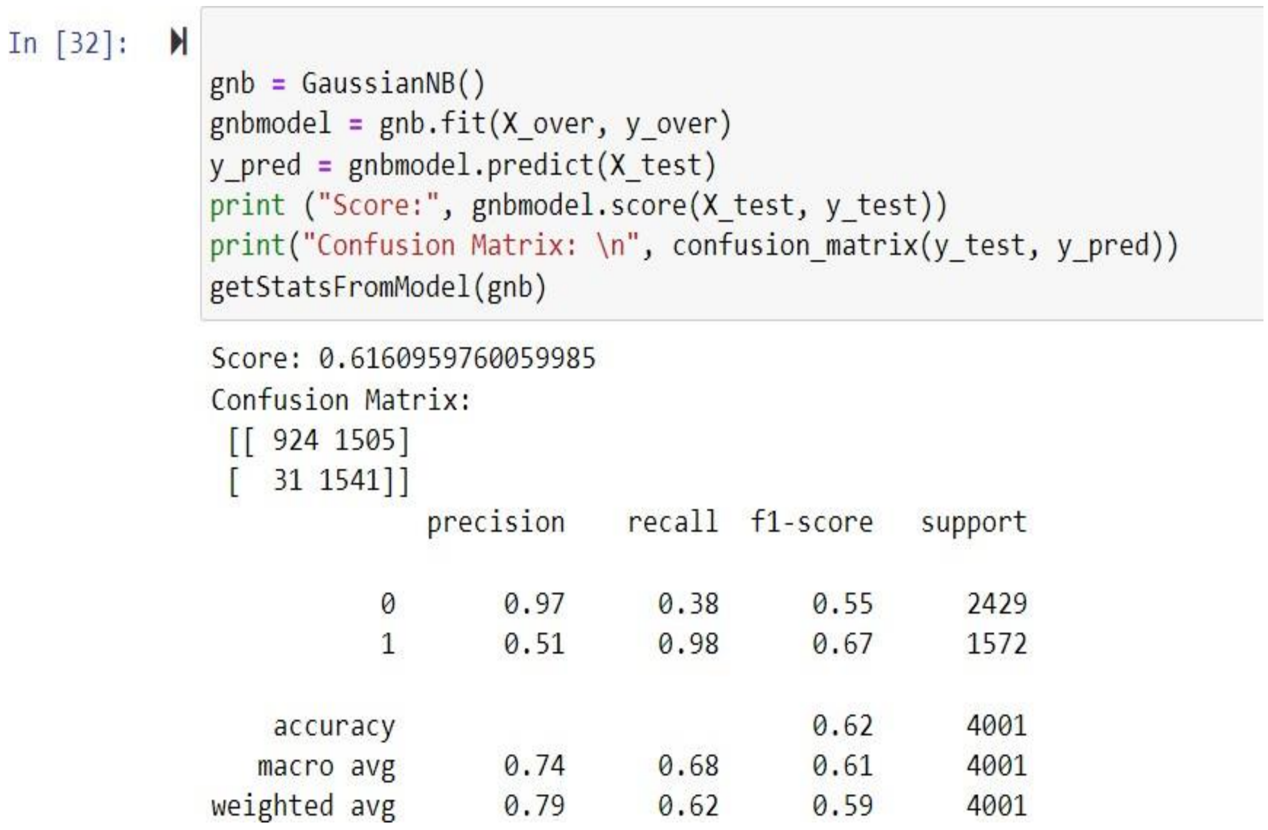
```
Accuracy:  0.8455386153461635
Confusion Matrix:
 [[1883  546]
 [  72 1500]]
              precision    recall  f1-score   support

           0       0.96      0.78      0.86      2429
           1       0.73      0.95      0.83      1572

    accuracy                           0.85      4001
   macro avg       0.85      0.86      0.84      4001
weighted avg       0.87      0.85      0.85      4001
```

Support Vector Machine:



```
In [69]:  lin_svc = LinearSVC()

In [70]:  lin_svc_cv_score = cross_val_score(lin_svc,X_train_tf,y_train,cv=5,scoring='f1_macro',n_jobs=-1)
          mean_lin_svc_cv = np.mean(lin_svc_cv_score)
          mean_lin_svc_cv

Out[70]:  0.8220066371295554
```

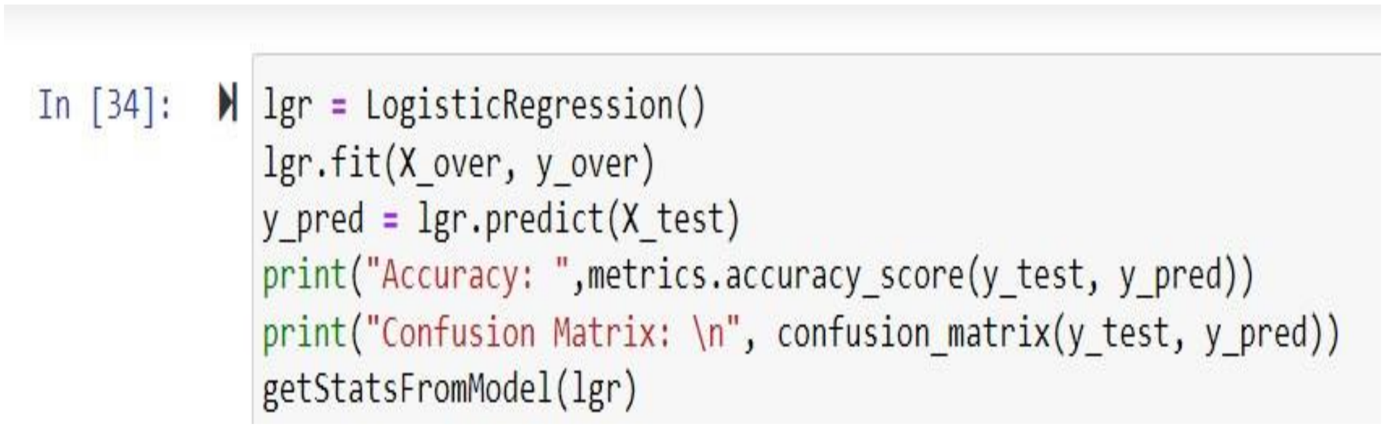
Naïve Bayes:



```
In [32]:
          gnb = GaussianNB()
          gnbmodel = gnb.fit(X_over, y_over)
          y_pred = gnbmodel.predict(X_test)
          print ("Score:", gnbmodel.score(X_test, y_test))
          print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
          getStatsFromModel(gnb)

          Score: 0.6160959760059985
          Confusion Matrix:
           [[ 924 1505]
           [  31 1541]]
                        precision    recall  f1-score   support

                     0       0.97      0.38      0.55      2429
                     1       0.51      0.98      0.67      1572

              accuracy                           0.62      4001
             macro avg       0.74      0.68      0.61      4001
          weighted avg       0.79      0.62      0.59      4001
```

Logistic Regression:

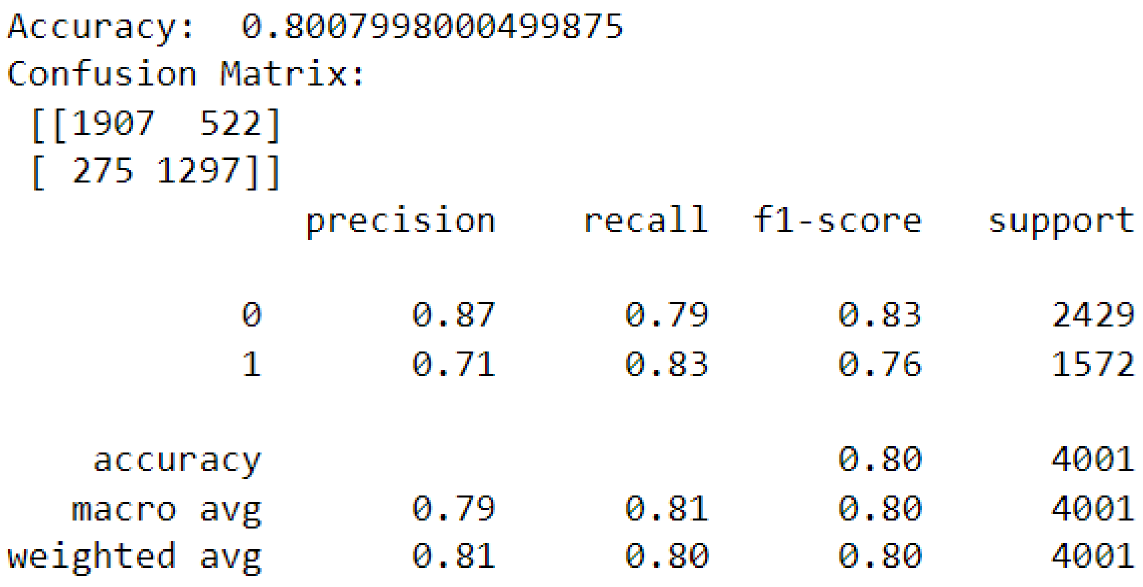
Input:



```
In [34]:  lgr = LogisticRegression()
           lgr.fit(X_over, y_over)
           y_pred = lgr.predict(X_test)
           print("Accuracy: ",metrics.accuracy_score(y_test, y_pred))
           print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
           getStatsFromModel(lgr)
```

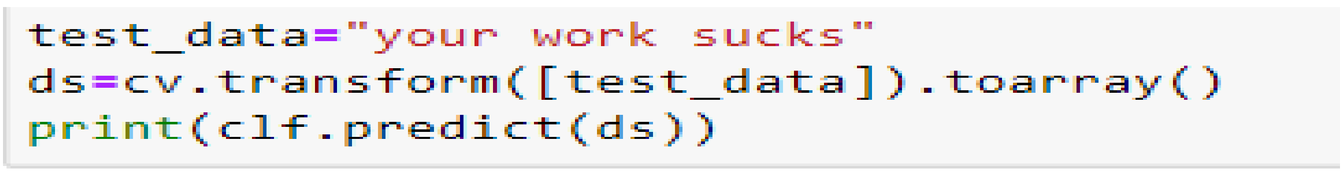
Output:

Logistic Regression Confusion matrix:



```
Accuracy:  0.8007998000499875
Confusion Matrix:
 [[1907  522]
 [ 275 1297]]
              precision    recall  f1-score   support

           0       0.87      0.79      0.83      2429
           1       0.71      0.83      0.76      1572

    accuracy                           0.80      4001
   macro avg       0.79      0.81      0.80      4001
weighted avg       0.81      0.80      0.80      4001
```

TESTING THE MODEL:



```
test_data="your work sucks"
ds=cv.transform([test_data]).toarray()
print(clf.predict(ds))
```

Output:

Cyber Bullying Detected

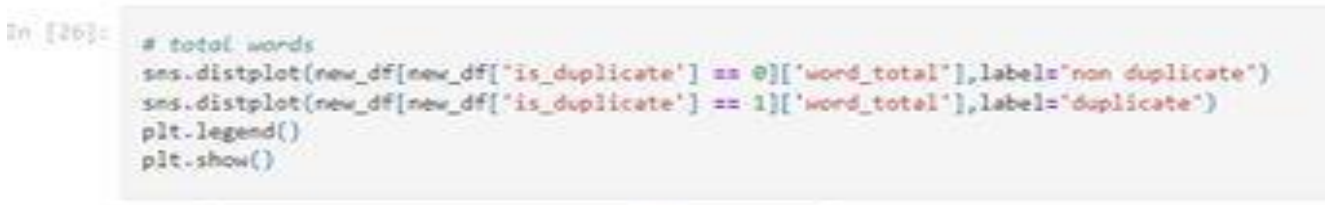
Input:

```
from sklearn import metrics
y_pred = clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

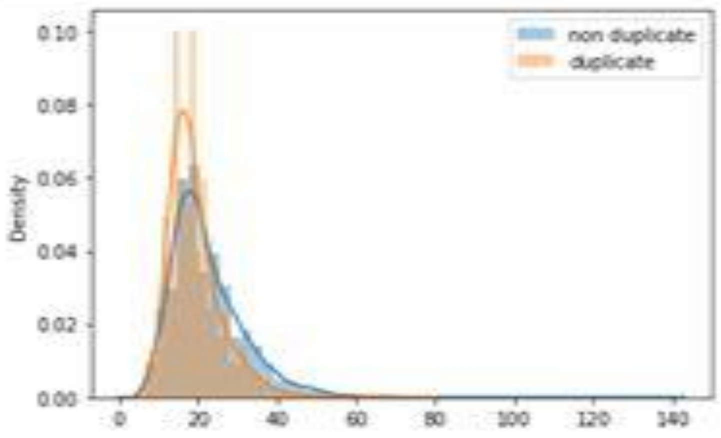
Output:

```
Accuracy: 0.8734309623430963
```

It describes which is cyberbullying, which is not



```
In [26]: # total words
         sns.distplot(new_df[new_df['is_duplicate'] == 0]['word_total'],label="non duplicate")
         sns.distplot(new_df[new_df['is_duplicate'] == 1]['word_total'],label='duplicate')
         plt.legend()
         plt.show()
```



:

CONCLUSION AND DISCUSSION:

CHALLENGES FACED:

1) Long Execution time: Running all of the 450 experiments in the first set was challenging since it required a huge amount of time to complete. To overcome the slow execution time of the models, the experiments were conducted in parallel on high-performance compute nodes in Compute Canada clusters.

2) Random under sampling: The use of random under sampling had a negative effect on model’s performance. This can be due to the huge loss of information that happens when using this under sampling method.
It is possible to experiment with over sampling techniques in the future to overcome this issue.

FUTURE WORK:

Along with the laws that are used to punish those people who cause cyber violence having an system that automatically detects the context of cyberbullying and changes it into a positive comment will be of great help as the saying goes prevention is the best .Using this will prevent lot of people from depression ,low self-esteem and also suicides.it also never let the users to use the social media as a tool to humiliate or bully others .The cyberbullying detecting system will lead to healthy environment on social media.
It can be embedded into all social media and messaging apps.

REFERENCES

BASE PAPER:

- Improving Cyberbullying Detection using Twitter Users' Psychological Featuresand Machine Learning
- Date Added to ELSEVIER: 31 December 2019
- Publisher: ELSEVIER

Authors: Vimala Balakrishnan Ph.D , Shahzaib Khan , Hamid R. Arabnia Ph.D Cyberbullying is the act of using technology, such as the internet and smartphones, to harass, humiliate, threaten, or intimidate someone. It can take many forms, including mean and hurtful messages, spreading rumors or lies, sharing personal information or photos without consent, and creating fake socialmedia profiles. The impact of cyberbullying can be severe, leading to emotional distress and even mental health problems.

LIST OF REFERENCE PAPERS

- ALBayari, Reem, Sharif Abdullah, and Said A. Salloum. "Cyberbullying classification methods for Arabic: A systematic review." Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2021). Cham: Springer International Publishing, 2021.

- Gencoglu, Oguzhan. "Cyberbullying detection with fairness constraints." IEEE Internet Computing 25.1 (2020): 20-29.

- Ali, Wan Noor Hamiza Wan, Masnizah Mohd, and Fariza Fauzi. "Cyberbullying detection: an overview." 2018 Cyber Resilience Conference (CRC). IEEE, 2018.

- Zhao, Rui, Anna Zhou, and Kezhi Mao. "Automatic detection of cyberbullying on social networks based on bullying features." Proceedings of the 17th international conference on distributed computing and networking. 2016

- Dadvar, Maral, et al. "Improving cyberbullying detection with user context." Advances in Information Retrieval: 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings *35*. Springer Berlin Heidelberg, 2013.
- M. Di Capua, E. Di Nardo, and A. Petrosino, "Unsupervised cyber bullying detection in social networks," in 2016 23rd International conference on pattern recognition (ICPR). IEEE, 2016, pp. 432–437.

- H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, "Detection of cyberbullying incidents on the instagram social network," arXiv preprint arXiv:1503.03909, 2015.

- V. Banerjee, J. Telavane, P. Gaikwad, and P. Vartak, "Detection of cyberbullying using deep neural network," in 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS). IEEE, 2019, pp. 604–607.

- H. Watanabe, M. Bouazizi, and T. Ohtsuki, "Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection," IEEE access, vol. 6, pp. 13 825–13 835, 2018.