AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY
DHAKA, BANGLADESH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CYBERBULLYING CLASSIFICATION AND VULNERABILITY DETECTION ON SOCIAL MEDIA

CSE 4100

Project & Thesis-I

Supervised By

Ms. Syeda Shabnam Hasan

Assistant Professor

Department of Computer Science & Engineering

Presented By

Sudipta Biswas	15.01.04.112
Rakib Hossain Ayon	15.01.04.120
Raphael Elvis Rozario	15.01.04.121
Fahim Md. Mahfuzur Rahman	15.01.04.047

Index

List of Figures

Chapter 1

Introduction

1.1 Preface

Cyberbullying means the use of electronic communication to bully a person, typically by sending a message to an intimidating or threatening nature. Cyberbullying allows bullies to easily and anonymously harass victims online. They do this by flaming, harassing, outing, exclusion, impersonation, and stalking [1]. Harmful bullying behavior can include posting rumors, threats, sexual remarks, a victims' personal information, or pejorative labels (i.e., hate speech) [2]. Victims may have lower self-esteem, increased suicidal ideation, and a variety of emotional responses, including being scared, frustrated, angry, and depressed. [3].

The internet is not the elusive commodity it was 10 years ago. The expansion of the use of internet has brought multiple opportunities to the youth; access to educational content has become easier. Of course, part of the internet that attracts the youth the most is the social media. It's hard to find any youngster without a Facebook or a Twitter or an Instagram account. Because of how exposed someone's life is on any social media, safety has become a gnawing concern, especially for minors [5].

To better understand online behaviors, Telenor Group released the results of its Safe Internet study, which examined internet safety knowledge among students across three of its markets in the region: Bangladesh, Thailand and Malaysia [4]. The Group released the results of its Safe Internet study, which examined internet safety knowledge among 1,896 students in Bangladesh, aged between 12 – 18 years, with respondents concentrated in key cities. It has been revealed that 49% of school-goers in Bangladesh have faced cyber-bullying [5]. Moreover, State Minister for Post and Telecommunications, Tarana Halim stated that around 73 percent women who use the internet in Bangladesh are subjected to cyber-bullying [6].

1.2 Problem Statement

Observing the present conditions of cyberbullying we aim to extinguish the fire before it spreads. Detecting cyberbullying contents accurately would fulfill our main objective. We will use various machine learning approaches to see which performs better for detecting cyberbullying contents.

1.3 Motivations:

Cyberbullying causes severe depression and sometimes even leads the victim to suicidal activities. It increases crime and violence and deteriorates the victim mentally. As a response to these cyber-crimes, a number of national and multi-national teenager protective projects (e.g., The Suicide Prevention Centre (http://www.preventiezelfdoding.be/), Child Focus (http://www.childfocus.be/)) have been starting over the last few years to increase online safety especially for children [7].

But it's too difficult to track cyberbullying in the vast online platforms manually. So, we are profoundly motivated to detect these vicious crime with an automatic system that enables to monitor the online platforms on a larger scale. The outcome will eventually assist the organizations in the maintenance of cyber-security.

Chapter 2

Scope of Thesis

As per our previous discussion we know that, cyberbullying is a process through which persons harass, threaten another person via social media. Detection of cyberbullying is a system which can identify those aggressive, bully words used by the person. In our paper, we are to use an algorithm named Support Vector Machine (SVM) with several kernel. Other algorithms like Naïve Bayes, Logistic regression etc. will also be used for comparing SVM model.

Detecting cyberbullying is always challenging. Several problems has to be faced with the dataset, algorithm, building e perfect model, accuracy level of result etc. And social media's security terms and policies are volatile. Our purpose is to surpass those challenges with a proficient and effective system with the following features:

High Detection Rate: Detection rate must be very much high otherwise it will be of no use. We are going to build SVM model with the capability of detecting cyberbullying with maximum accuracy level.

Reducing unusual Act: As per research, many case studies have been done on the impact of cyberbullying. One of the severe impacts was the suicidal rate of teenagers because of being bullied. If system can highly be skilled that people won't bully one at least in fear of being detected, this rate can be reduced. We want to build this kind of strong system.

False Identification of bully word: This means if a word is a bully. But still a system detecting it as a non-bully one. This happens when the system trains more non-bully word than bully. This kind of false positive detection can lessen the possibility of detecting real cyberbullying. So, we will try our best to look into this issue and prevent the amount of false positive detection rate.

Reduced Performance and computation time: The system has to deal with a vast dataset. So, saving real processing time is a big deal and important. We want to build a model with optimal performance and processing time.

Getting Direct Resultant classes: Some model gives the probability as result. But in social media it is very much significant to get the direct result of cyberbullying. Probability is sort of ambiguous. So, we are working with SVM which gives direct resultant classes.

Chapter 3

Preliminaries

3.1 Introduction to Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves. The process of learning begins with observations or datasets; such as examples, direct experience, or instruction; in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Machine learning algorithms are often categorized as the followings:

- **Supervised machine learning** algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
- **Unsupervised machine learning algorithms** are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.
- **Semi-supervised machine learning algorithms** fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning

accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it.

- **Reinforcement machine learning algorithms** is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance.

Machine learning enables analysis of massive quantities of data. It's far superior to the traditional database and handmade programmed methods as it can learn and take actions on its own keep up to date features, no external help necessary. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources for training properly. [19]

3.2 Text Classification with Machine Learning

Machine learning algorithms can only train and predict for numerical data. So when it comes down to handling texts, documents and strings, we have to take a different approach and somehow create a numerical context for the respective text so the machine can understand and analyze. The goal of text classification is to automatically classify the text documents into one or more defined categories. For a general classification problem for a supervised learning model, we import, instantiate, fit and then predict where as for text we have to import, instantiate, fit, transform and then predict.

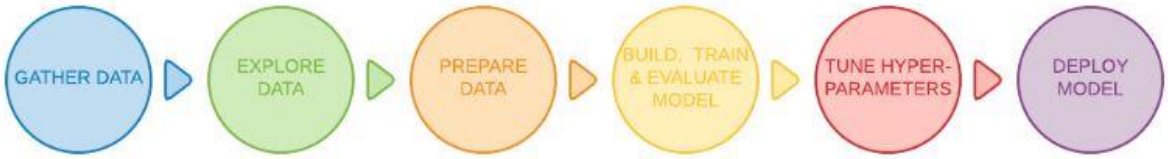

Fig 3.1: Workflow for a machine learning problem

3.2.1 Dataset Preparation

The first step is the Dataset Preparation step which includes the process of loading a dataset and performing basic pre-processing. The dataset is then split into train and validation sets called training and testing data. The train-test splitting can be done in any ratio but by convention, its (75%-80%) training data and (25%-20%) testing data. Data is collected from various trustworthy sites and then hand labeled in supervised learning. As the data is collected raw from various sites, sometimes from social media, it is filled with various unnecessary characters, punctuations, spaces or exclamations which may make accuracy of the model of the track. So we need to first clean the data and make it more machine friendly, it's called preprocessing. Preprocessing includes deleting inverted commas or other unnecessary punctuations, deleting extra spaces, deleting emoticons, substitute missing value with dummy value, replace erroneous values, decomposing data meaning making complex data simpler and into multiple parts that will help in capturing more specific relationships, rescaling data that aim at improving the quality of a dataset by reducing dimensions and avoiding the situation when some of the values overweight others, inserting or deleting data based on the users' needs etc. Some datasets are made public, a user can do independent work with this data. Datasets are usually in the form of csv, json or xml data format. CSV (comma separated value) files are most popular.

3.2.2 Feature Engineering:

The next step is the Feature Engineering in which the raw dataset is transformed into flat features which can be used in a machine learning model. This step also includes the process of creating new features from the existing data.

Tokenization: It is the process of cutting or separating each word that compiles a document or conversation. Generally, each word is spilt from another word by putting a space character, single quoting character (‘), dot (.), colon (:) and semicolon (;). So, it is the process of dividing text into a set of meaningful pieces. These pieces are called tokens. Depending on the task at hand, we can define our own conditions to divide the input text into meaningful

tokens. And the non-letters perform the separation. The challenges of tokenizing depends on the language type.

Stemming: Stemming is the process of conflating the variant forms of a word into a common representation, the stem. For example, the widely used procedure in text processing for information retrieval (IR) based on the assumption that posing a query with the term presenting implies an interest in documents containing the words presentation. There are several stemming algorithms like **Porter stemmer algorithm, Table Lookup Approach** etc.

Stop word removal: This process eliminates unnecessary words on every text conversation in accordance with English vocabulary by using English stop word filter. Stop words are very frequently used common words like **'are', 'or', 'and', 'this'** etc. They are not useful in classification of texts as the return useless information. So, they must be removed.

Transform Case: This step is to **transform into lower case** in order to neutralize next processes with purpose not to differentiate between uppercase and lowercase letters.

Lemmatization: Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. In Lemmatization root word is called **Lemma**. A lemma (plural lemmas or lemmata) is the canonical form, dictionary form, or citation form of a set of words. It is one of the important steps of data preprocessing.

Generating n-grams: An N-gram is an N-character slice of a longer string. Although in the literature the term can include the notion of any co-occurring set of characters in a string (e.g., an N-gram made up of the first and third character of a word). Using n-gram of **size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "diagram"); size 3 is a "trigram".**

Feature Extraction:

CountVectorization: CountVectorizer builds a count matrix where rows are occurrences counts of different words taking into account the high-dimensional sparsity. It provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. We can use it as follows:

- Create an instance of the **CountVectorize*r*** class.
- Call the **fit()** function in order to learn a vocabulary from one or more documents.
- Call the **transform()** function on one or more documents as needed to encode each as a vector

An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document. Because these vectors will contain a lot of zeros, we call them sparse. Python provides an efficient way of handling sparse vectors in the scipy.sparse package.[26]

TF-IDFvectorization: Term Frequency-Inverse Document Frequency (TF-IDF) is a numerical statistics that intends to reflect how important a word is to a document in a collection of corpus. The tf-idf value increases proportionally to the number of times a word appears in the document [17]. It is an alternative way to calculate word frequencies and the most popular one. [26]

- **Term Frequency**: This summarizes how often a given word appears within a document.
- **Inverse Document Frequency**: This downscales words that appear a lot across documents.

Inverse document frequency is defined as:

$$\mathbf{idf(t,D) = logN|d \in D: t \in d|idf(t,D) = logN|d \in D: t \in d|}$$

Where N is the total number of documents in the corpus, and |d∈D:t∈d||d∈D:t∈d| is the number of documents where t appears. Without the inverse document frequency part commen, less meaning giving words such as “the” (assuming you don’t filter stopwords) would bear a higher weight than these less frequent words. TfidfVectorizer combines all options of CountVectorizer and TfidfTransformer in a single model.

HashVectorizer: Counts and frequencies can be very useful, but one limitation of these methods is that the vocabulary can become very large. [26]This, in turn, will require large vectors for encoding documents and impose large requirements on memory and slow down algorithms.

A clever work around is to use a one way hash of words to convert them to integers. The clever part is that no vocabulary is required and you can choose an arbitrary-long fixed length vector. A downside is that the hash is a one-way function so there is no way to convert the encoding back to a word (which may not matter for many supervised learning tasks).

The HashingVectorizer class implements this approach that can be used to consistently hash words, then tokenize and encode documents as needed.

3.2.3 Model Training

The final step is the Model Building step in which a machine learning model is trained on a labelled dataset. In our next section we talk in depth about a few popular classification algorithms that we have used and compared later in our thesis project.

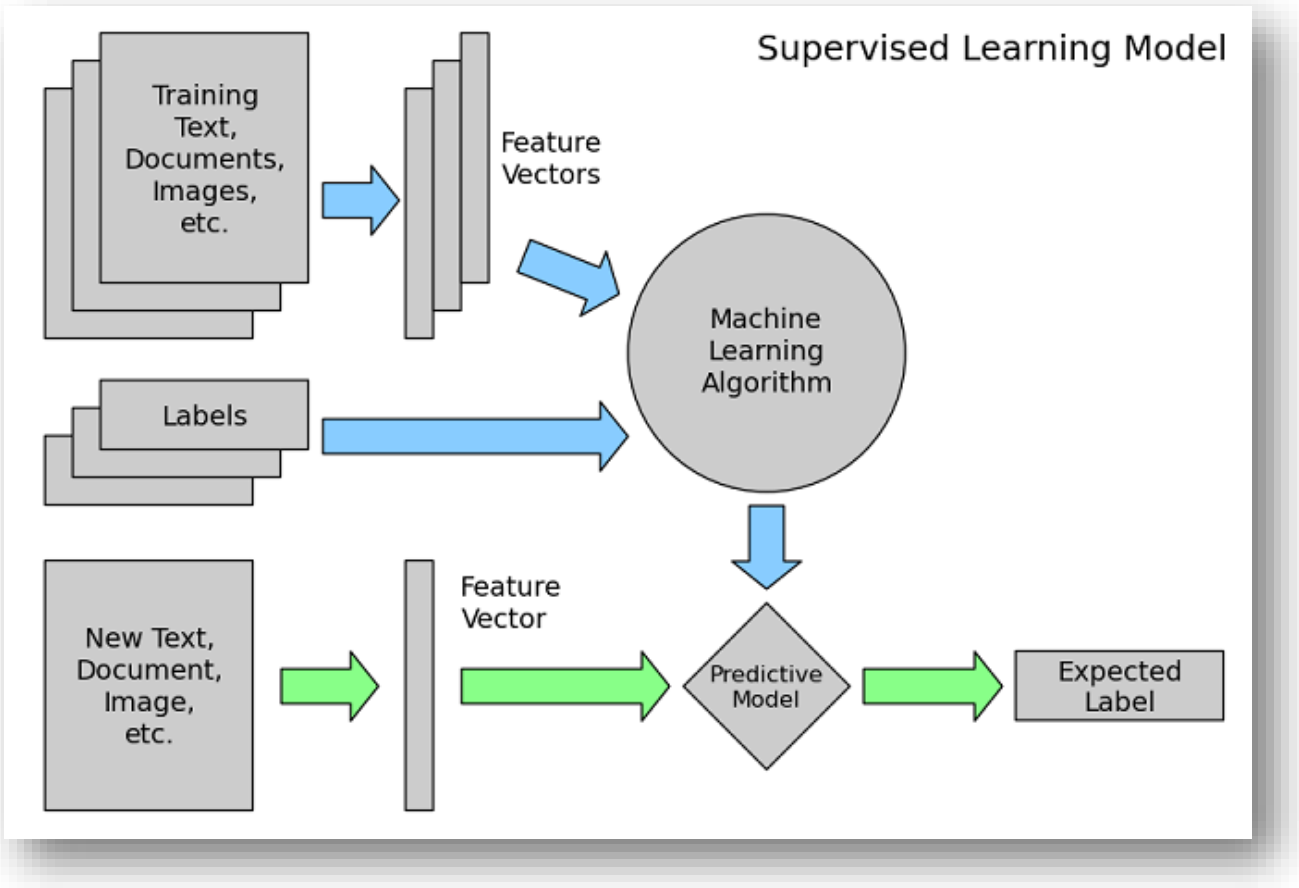

Fig 3.2: workflow for a text classification problem

3.3 Some Popular Classifiers:

3.3.1 Naïve Bayes Classifier:

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable y and dependent feature vector $\mathbf{x_1}$ through **xn** :

$$P(y|x1,\ldots,xn) = P(y)\ P(x1,\ldots,xn|y)\ P(x1,\ldots,xn)$$

Using the naive conditional independence assumption that

$$P(xi|y, x1,\ldots,xi-1, xi+1,\ldots,xn) = P(xi|y),$$

for all i, this relationship is simplified to

$$P(y|x1,\ldots,xn) = P(y) \prod_{i=1}^{n} P(xi|y)\ P(x1,\ldots,xn)$$

Since P(x1,…,xn) is constant given the input, we can use the following classification rule:

$$P(y|x1,\ldots,xn) | P(y) \prod_{i=1}^{n} P(xi|y) | y\hat{} = \arg|\max y P(y) \prod_{i=1}^{n} P(xi|y),$$

And we can use Maximum A Posteriori (MAP) estimation to estimate **P(y)** and **P(xi|y)**; the former is then the relative frequency of **class y** in the training set.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of **P(xi|y).**

Naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters. Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods.

Though Naive Bayes is known as a decent classifier, it is known to be a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.

Gaussian Naïve bayes:

Gaussian Naïve bayes implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i|y) = 12\pi\sigma y2exp|(-(xi-\mu y)22\sigma y2)$$

The parameters **σy** and **μy** are estimated using maximum likelihood.

Bernoulli Naïve Bayes:

Bernoulli Naïve Bayes implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions.

The decision rule for Bernoulli naive Bayes is based on

$$P(x_i|y) = P(i|y)x_i + (1-P(i|y))(1-x_i)$$

Which differs from multinomial NB's rule in that it explicitly penalizes the non-occurrence of a feature that is an indicator for class, where the multinomial variant would simply ignore a non-occurring feature.

In the case of text classification, word occurrence vectors (rather than word count vectors) may be used to train and use this classifier. Bernoulli Naïve Bayes might perform better on some datasets, especially those with shorter documents. It is advisable to evaluate both models, if time permits.

Multinomial Naïve Bayes:

Multinomial Naïve Bayes implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors $\theta y=(\theta y1,\ldots,\theta yn)$ for each class y, where n is the number of features (in text classification, the size of the vocabulary) and θyi is the probability $P(x_i|y)$ of feature i appearing in a sample belonging to **class y**.

The parameters **θy** is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:

$$\hat{\theta}_{yi} = N_{yi} + \alpha N_y + \alpha n$$

Where $N_{yi} = \sum x|T_{xi}$ is the number of times feature i appears in a sample of **class y** in the training set **T**, and $N_y = \sum_{i=1}^{n} N_{yi}$ is the total count of all features for **class y**.

The smoothing priors $\alpha \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting $\alpha = 1$ is called Laplace smoothing, while $\alpha < 1$ is called Lidstone smoothing.

3.3.2 Logistic regression:

Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Basically, the dependent variable is categorical.

There are 3 types of logistic regression:

1. **Binary Logistic Regression:** The categorical response has only two 2 possible outcomes. Example: Spam or Not.
2. **Multinomial Logistic Regression:** Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan).
3. **Ordinal Logistic Regression:** Three or more categories with ordering. Example: Movie rating from 1 to 5.

Logistic regression is named for the function used at the core of the method, the logistic function [20]. The logistic function, also called the **sigmoid function** was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

$$1 / (1 + e^{-value})$$

Where e is the base of the natural logarithms (Euler's number or the EXP () function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.

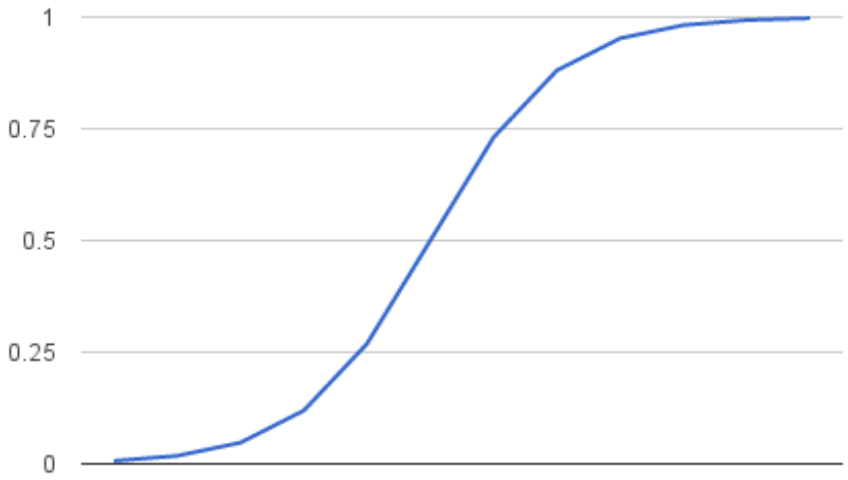

Fig 3.3: plot using logistic regression function

Logistic regression uses an **equation as the representation**, very much like linear regression. Below is an example logistic regression equation:

y = e^ (b0 + b1*x) / (1 + e^ (b0 + b1*x))

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data. The actual representation of the model that you would store in memory or in a file are the coefficients in the equation (the beta value or b's). The **application of Logistic Regression** was mostly in the biological sciences in early twentieth Century. It was then used in many social science applications [21]. Now it is also used in financial forecasting, Software cost prediction, software effort prediction and software quality assurance etc.

3.3.3 K-Nearest Neighbor Classifier:

K-NN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution. In other words, the model structure determined from the dataset. This is very helpful in practice where most of the real world datasets do not follow mathematical theoretical assumptions. Lazy algorithm means it does not need any training data points for model generation. All training data used in the testing phase. This makes training faster and testing phase slower and costlier. Costly testing phase means time and memory. In the worst case, KNN needs more time to scan all data points and scanning all data points will require more memory for storing training data.

A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor. Distance Functions for various method are:

Euclidean: $\sqrt{\sum_{i=1}^{k}(xi - yi)^2}$

Manhattan: $\sum_{i=1}^{k}|xi - yi|$

Minkowski: $\left(\sum_{i=1}^{k}(|xi - yi|)^q\right)^{i/q}$

It should also be noted that all three distance measures are only valid for continuous variables. In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and categorical variables in the dataset.

Hamming Distance:

$$D_H = \sum_{i=1}^{k} |x_i - y_i|$$

$$x = y \rightarrow D = 0$$

$$x = y \rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise but there is no guarantee. Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. Historically, the optimal K for most datasets has been within 3-10. That produces much better results than 1NN.

3.3.4 Decision Tree

Decision tree is one of the most popular machine learning algorithms. Decision trees are used for both classification and regression problems. Decision tress often mimic the human level thinking so it's simple to understand. A decision tree is a tree where each node represents a feature (attribute), each link (branch) represents a decision (rule) and each leaf represents an outcome (categorical or continues value). The whole idea is to create a tree for the entire data and process a single outcome at every leaf (or minimize the error in every leaf). There are couple of algorithms to build a decision tree, we only talk about 2 important ones which are:

CART (Classification and Regression Trees):

Gini Index is used as the cost function to evaluate splits in the dataset. The target variable is Binary variable which means it take two values (Yes and No). A Gini score gives an idea of how good a split is by how mixed the classes are in the two groups created by the split. A perfect separation results in a Gini score of 0, whereas the worst case split that results in 50/50 classes. Max Gini Index value is 0.5. Here, the lowest answer is taken. Gini Index for Binary Target variable is = 1—P^2(Target=0)—P^2(Target=1). Suppose, for a data of 14 instances where class [0] = 5 and class [1] = 9; gini index = 1 - (5/14) ^2 – (9/14) ^2 = .46.

ID3 (Iterative Dichotomiser 3):

It uses Entropy function and Information gain as metrics. It performs top-down, greedy search through the space of possible decision trees. In order to define information gain precisely, we begin by defining a measure commonly used in information theory, called entropy that characterizes the impurity of an arbitrary collection of examples. Here, if all examples are positive or all are negative then entropy will be 0 and if half of the examples are of positive class and half are of negative class then entropy 1.The highest value is taken. Here the used equation for calculating entropy is; $H(S) = \sum_{c=\{0,1\}} p(c)\ log_2\ p(c)$.

Meaning, for a data of 14 instances where class [0] = 5 and class [1] = 9; P (1) = (9/14)*$\log_2$ (9/14) = .41; P (0) = (5/14)*$\log_2$ (5/14) = .53; H(S) = .41+.53 = .94.We get IG by subtracting summation of target entropies from 1.

Steps for this classifier for both algorithms:

1. Compute the entropy for data-set

2. For every attribute/feature:

- Calculate gini index/entropy for all categorical values
- Take average information entropy for the current attribute
- Calculate gain for the current attribute

3. Pick the highest gain attribute.
4. Repeat until we get the tree we desired.

3.3.5 SVM:

Support Vector Machine (SVM) is a supervised learning model that is used for classification and regression analysis [15]. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Poly kernel: Polynomial Kernel is a kernel function commonly used with support vector machines (SVMs) and other kernel models that represents the similarity of vectors (training samples) in a feature space over polynomials of the original variables. It allows learning of non-linear models. The equation is:

$$\mathbf{K\ (xi,\ xj) = (xi + xj + 1)^{d}}$$

Where d is the degree of the polynomial. Then xi and xj are vectors in the *input space*, i.e. vectors of features computed from training or test samples and $1 \geq 0$ is a free parameter trading off the influence of higher-order versus lower-order terms in the polynomial. When the parameter will be 0, the kernel is called homogeneous [16].

Linear Kernel: The linear kernel is good when there is a lot of features. That's because mapping the data to a higher dimensional space does not really improve the performance. [22] In text classification, both the numbers of instances (document) and

features (words) are large. Training a SVM with a linear kernel is faster than with another kernel. Particularly when using a dedicated library such as LibLinear. [22] The function of linear SVM is :

$$\mathbf{K(xi,xj)= xi * xj}$$

Where, xi and xj are the vectors in the input space.

RBF Kernel: Radius Basis Function (RBF) or Gaussian function is one of most popular kernel function used in SVM. The RBF kernel on two input vector xi, xj, represented as future vectors in some input space.So, the function defines as,

$$\mathbf{K(xi, xj)=exp(-\Upsilon\ ||xi - xj||^2)}$$

Here, Υ is equivalent to - $\frac{1}{2\sigma^2}$.

Sigmoid Kernel: This type of kernel is used as proxy in neural network. It is also known as hyperbolic tangent kernel. It is interesting to note that a SVM model using a sigmoid kernel function is equivalent to a two-layer, perceptron neural network [23]. This kernel was quite popular for support vector machines due to its origin from neural network theory. Also, despite being only conditionally positive definite, it has been found to perform well in practice. The equation for this kernel is:

$$\mathbf{k(xi,xj) = \tanh(\alpha xi^T xj + c)}$$

There are two adjustable parameters in the sigmoid kernel, the slope alpha and the intercept constant c. A common value for alpha is 1/N, where N is the data dimension.

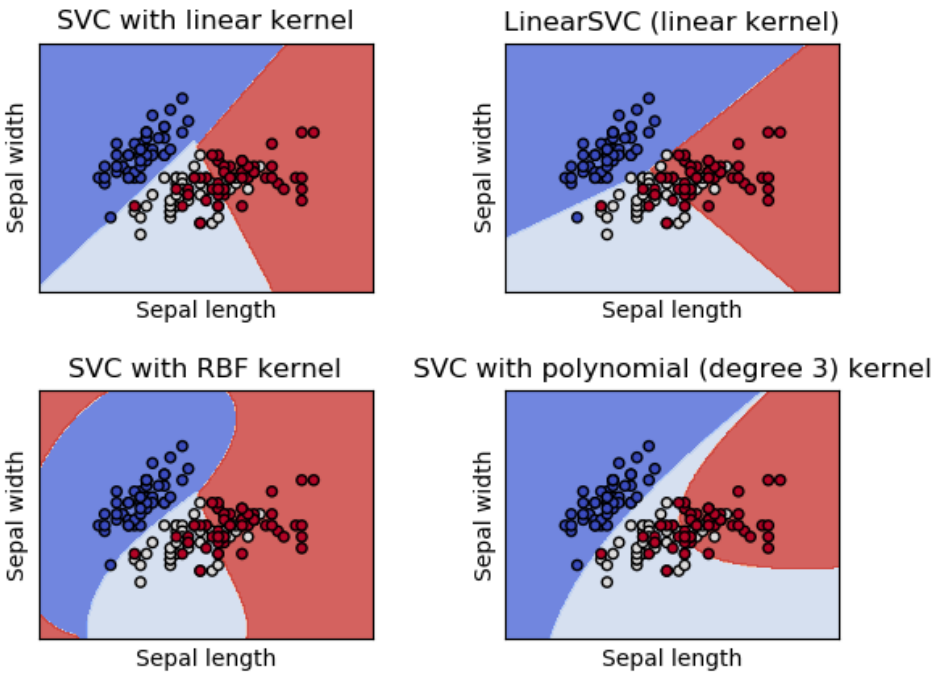

Fig 3.4: Support Vector Classifier with different kernels

3.4 Result Analysis

The result in the accuracy test with one of the machine learning algorithms doesn't paint the full picture whether the result is good or not. We need to see some other criteria's also. Some important terms of such are discussed below:

Confusion Matrix: A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. Let's see the following confusion matrix:

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Fig 3.5: example of a confusion matrix

Here we see that, there are total of 165 test data, 60 of them were of class ‘no’ and 105 of them were of class ‘yes’. In the class ‘no’ 50 of the answers were predicted correctly and in class ‘yes’ 100 of them were predicted correctly.

True positives (TP): These are cases in which we predicted yes and they were actually correct.

True negatives (TN): We predicted no, and they were actually correct.

False positives (FP): We predicted yes, but they would have been no.

False negatives (FN): We predicted no, but would actually be yes.

Accuracy: Overall, how often is the classifier correct?

- (TP+TN)/total = (100+50)/165 = 0.91

Misclassification Rate: Overall, how often is it wrong?

- (FP+FN)/total = (10+5)/165 = 0.09
- equivalent to 1 minus Accuracy
- also known as "Error Rate"

True Positive Rate: When it's actually yes, how often does it predict yes?

- TP/actual yes = 100/105 = 0.95
- also known as "Sensitivity" or "Recall"

False Positive Rate: When it's actually no, how often does it predict yes?

- FP/actual no = 10/60 = 0.17

True Negative Rate: When it's actually no, how often does it predict no?

- TN/actual no = 50/60 = 0.83
- equivalent to 1 minus False Positive Rate
- also known as "Specificity"

Precision: Precision talks about how precise/accurate your model is out of those predicted positive, how many of them are actual positive. Precision is a good measure to determine, when the costs of False Positive is high

Precision= True Positive/ (True Positive +False Positive)

=True Positive/Total Predicted Positive

= 100/110 = 0.91

Recall: Recall actually calculates how many of the Actual Positives our model capture through labeling it as Positive (True Positive). Applying the same understanding, we know that Recall shall be the model metric we use to select our best model when there is a high cost associated with False Negative.

Recall= True Positive/ (True Positive +False Positive)

=True Positive/Total Actual Positive

= 100/105 = .95

Prevalence: How often does the yes condition actually occur in our sample?

- *actual yes/total = 105/165 = 0.64*

A couple other terms are also worth mentioning:

Null Error Rate: This is how often you would be wrong if you always predicted the majority class. (In our example, the null error rate would be 60/165=0.36 because if you always predicted yes, you would only be wrong for the 60 "no" cases.) This can be a useful baseline metric to compare your classifier against. However, the best classifier for a particular application will sometimes have a higher error rate than the null error rate, as demonstrated by the Accuracy Paradox.

Cohen's Kappa: This is essentially a measure of how well the classifier performed as compared to how well it would have performed simply by chance. In other words, a model will have a high Kappa score if there is a big difference between the accuracy and the null error rate.

F Score: This is a weighted average of the true positive rate recall and precision. F1 Score is needed when you want to seek a balance between Precision and Recall especially in an unbalanced dataset.

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

= 2.91*.95/.91+.95 = .92*

ROC & AUC Curve: This is a commonly used graph that summarizes the performance of a classifier over all possible thresholds. It is generated by plotting the True Positive Rate (y-axis) against the False Positive Rate (x-axis) as you vary the threshold for assigning observations to a given class. An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve.

Chapter 4

Related Work

Paper [8] being the first to detect cyberbullying in Arabic text as per their literature review. They have approached through text mining and lexicon based technique. At first they have gone through some case studies. Then they surveyed on this issue like how much aware people are about cyberbullying and so on. In their survey they found that girls (45%) are having more experience on cyberbullying than of boys (28%). For applying methodologies, they used data set of Arabic tweet conversation. At first they develop a corpus of Arabic tweet conversations. In text mining approach, firstly they applied preprocessing techniques. Then they used it to train and test classifier that can detect cyberbullying. In lexicon based approach, firstly they developed an Arabic lexicon with bully words. Then the corpus of Arabic tweet conversation got checked with the lexicon. If they found number of bully words beyond some threshold, it could label as cyberbullying. In the paper, basically they have discussed the challenge of cyberbullying in modern world after surveying and studied several cases and showed up two methods to detect cyberbullying. But they didn't implement the methods yet. So, how the results will be with these methods are ambiguous. Besides, the corpus of lexicon on Arabic language still doesn't exist. It must be developed first. The methods also can detect words as bully which was not meant to say as cyberbullying.

Paper [9] comes up with a research on classification of cyberbullying comment on Instagram social media. They approached with the methodology, Support Vector Machine (SVM) which is created with R language. As dataset, they have chosen comments from accounts of Indonesian selebgram/celebrities after surveying that people get bullied in Instagram more (42%) than other social media. More precisely they used accounts of Karin Novilda and Samuel Alexandar who are Indonesian selebgram. 1053 comments were taken as training documents and 34 as test documents. For implementing the method, firstly, they created a document term matrix with R in order to form SVM model. Once the formation of SVM

completed, they used it to predict comment whether it is cyberbully or not. They specified a comment by captions -1 if it was cyberbullying and 1 if it was not bullying. Finally they got accuracy percentage of 79, 412% as result using SVM. They didn't use any kernel of SVM and in paper [11] SVM with poly kernel shows a better accuracy then this papers' result (99.41%).SO, using kernel might give better accuracy.

Paper [10] worked on application of machine learning to detect Bangla texts which are abusive. Actually, it looked into the machine learning algorithms and compared which one is better. Their experiment includes algorithm Multinomial Naïve Bayes (MNB), Random Forest (RF) and Support Vector Machine (SVM). The kernels of SVM used are linear, Radial Bias Function (RBF), Polynomial and Sigmoid. For this exploration they collected data from the account of Bangladeshi Facebook celebrities. Comments on their posts have been used. Then they removed the special characters like @, - etc. Only Bengali Unicode were taken. And for the validation of results, 10 times 10 folds cross-validation method was conducted. Applying this method they found 50% abusive words. Next, the tests are conducted with three types of string features that are unigram, bigram, trigram. After that, unigram, bigram, trigram features are extracted from all the comments and vectorized using CountVectorizer and TfidfVectorizer. After vectorization they have shown that in all cases SVM with linear kernel shows the highest accuracy level. Lastly, they concluded that TfidfVectorizer features with SVM linear kernel holds the maximum accuracy among all the algorithms and MNB is the 2nd best. It didn't mention anything about spelling mistaken comments. This is the major limitation in this paper.

Paper [11] constructs a classification method using SVM with several kernel and Naïve Bayes. They have compared their methodology with the research of Kelly Reynolds who used decision tree (J-48) and k-NN (k = 1 and k = 3). The data used to create a data set is a textual conversation taken from the Kaggle (www.kaggle.com) which provides 12,729 data. They proceeded with data preprocessing, extraction, classification and lastly evaluation. They divided data into 2, 4 and 11 classes. After completing extraction text got classified through

Naïve Bayes, SVM with linear, RBF, poly, sigmoid kernels. Then they evaluated the accuracy rate with the method of confusion matrix. Based on Model, SVM-Linear gave the best average result (95.91%) and SVM-RBF gave the worst average result (77.41%). On the basis of n-grams, the best average result was attained by n-gram 5 (92.75%) and worst one was attained by n-gram 1 (89.05%). Lastly, 4 classes achieved the best average result (92.50%) considering the number of classes in the dataset. So, Poly kernel is the most optimal SVM kernel in classifying cyberbullying and n-gram 5 is the best suited in this scenario for increasing accuracy level. Yet, this research has some constraints. Text conversations usually have shortened words and numerous spelling mistakes. These cases were not taken care of in here.

Paper [12] is the first study in Turkey which detects cyberbullying from Turkish texts. They created a dataset from Instagram and twitter messages and applied machine learning techniques such as: Support Vector Machine (SVM), Decision Tree (C4.5), Naive Bayes Multinomial (MNB) and k-Nearest Neighbor (kNN) to detect and classify cyberbullying. The aim of this study was to show the success of cyberbully detection in Turkish text messages. The dataset was constructed manually consisting of 900 twitter and Instagram messages. Half of the messages (450 messages) were cyberbullying content and another half was cyberbullying unrelated content. Half of the cyberbullying contents (225 messages) were written by male another half were written by female users. Classification performance is measured with the F-measure Value, $F\text{-}measure = (2 \times recall \times precision) / (recall + precision)$.Two well-known feature selection method Chi-Square (CH12) and Information Gain (IG) were applied to show whether feature selection improves the classification accuracy or not. Then they applied J48(Decision Tree), NBM(Naïve Bayes Multinomial), SVM(Support Vector Machine), IBk(k- Nearest Neighbor) classifiers to each fold for both datasets, calculated the F-measure values and took the average of the F-measure values for five folds. It's given as Baseline result. There is also a Threshold result. There were two types of datasets, one with emoticons and another without emoticons. In comparison the dataset with emoticons had the better classification accuracy. The feature selection methods CH12 and IG both gave quite similar results but IG had slightly better accuracy. In terms of

accuracy, NBM performed the best when features are not applied and IBk was the most accurate when features were applied. Accuracy of all classifiers improves except for J48 when feature selection is applied. The accuracy of SVM was lower than NBM and IBk in most of the cases because the parameters were not optimized, default parameters were being used. They suggested that if the parameters were optimized it might have given better results. In terms of running time, NBM became the best classifier in terms of both training and testing time. SVM was second best. IBk had a lazy approach as total running time was too high but if feature selection was applied it would be the second best choice. So altogether NBM was the best performer. The study is well versed but there are some drawbacks. In the datasets there were male and female written messages collected but it didn't carry any significance as no study was done on it afterwards. Stemming and spelling correction was not done due to the fact that it could have changed the meaning of the word but if they could have performed those tasks accuracy would have been much better.

Paper [13] is the 2nd research of the authors on cyberbullying. Here, they worked on prediction of cyberbullying occurrence in media-based social media, therefore, they predicted cyberbullying from an image typically with a text caption also the comments followed by the image in America. They have chosen Instagram as social media. For data set, they used 25k public account in Instagram. They collected the user's profile data which includes image with caption and comments of other users. For labeling, used a dictionary with profane words and marked that at least one negative word exist in the texts. Then they selected a lower bound of 15 based on comments posted by other users. All 922 media session then with more than 40% (set40+) negativity and less than 40% (set0+) negativity. After that they filtered two label. Thus getting labeled data they designed and evaluated classifiers to predict and detect cyberbullying. A majority of criterion was assigned to detect. To design and train classifier fivefold cross validation method was applied. Also a logistic regression applied to train predictor. 98% of cyberbullying activities captured for set0+. It showed that cyberbullying incidents can be predicted with 0.99 recall for Set0+ and 0.61 recall for Set40+. The best false positive rate over Set0 is 3%, using only the image contents, media and user metadata based on a ridge regression classifier. This paper didn't obtain

greater detail from the labeling surveys. Also they didn't consider the commenting history of users in previously shared media. These are the basic limitation of these research paper.

Paper [14] is about cyberbullying detection in Arabic language. They have shown how NLP and some machine learning algorithm works to detect cyberbullying. The machine learning algorithms include Naïve Bayes, K-NN, support vector machine, Decision Tree etc. They discussed detail of these algorithms. Then they gave an idea about natural language processing (NLP), common sense reasoning and performance measurement. They proposed a multilingual cyberbullying detection system in Arabic language on Facebook and Twitter. Then they intended to collect dataset from Facebook and Twitter and classifying data with ML algorithms. For the performance measurement, they wished to use re-call, precision and F-measure to reach a system with optimum performance. They also didn't implement any methodology to detect cyberbullying. They only proposed to apply the above methods.

Paper [15] develops a hate speech classifier for Italian Language. They have built a corpus of comments from Facebook public pages of Italian newspapers, politicians, artists, groups etc. They collected 17,567 comments from 99 posts from these pages. Some of the comments were annotated to one of the three levels of hate: no hate, weak hate and strong hate. Rest of the comments were annotated to one of the two levels of hate: hate and no hate. They tested these dataset with two classifiers: I) SVM and II) Recurrent Neural Network named Long Short Term Memory (LSTM). They followed 10-fold cross validation process for each dataset. On the three-class dataset, SVM and LSTM gave 64.61% and 60.50% of accuracy respectively. On the two-class dataset, SVM and LSTM attained the accuracy of 80.60% and 79.81% respectively. We see that, they produced a better result with SVM classifier. But the results of three-class dataset were not satisfactory using any of the classifiers. In future, they would like to examine the results also for different types of hate like religion, politics, handicap, racism, sex etc.

Paper [16] is about a research on cyberbullying detection in Twitter. They approached with a whole new method called embedding's enhanced Bag-of-Words model (EBoW). As data set, they used texts or posts from Twitter. For implementing EBoW they first defined a list of insulting words based on expert knowledge and linguistic resources. Then extended insulting seeds to define bullying features. Different weights are assigned to bullying features based on the cosine similarity between word embedding's. After that, based on weight they classified the intense of cyberbullying. They took 1762 sample data from Twitter and they got 684 data as bullying instances. They trained and tested with 5-fold comparing with BoW, sBoW, LDA, LSA. The result of EBoW came out best of all. Precision was 76.8%, Recall 79.4% and F1 Score 78.0%. After this representation they intended to use SVM linear to classify and detect the texts. As, they didn't classify yet, the result of detecting bully words is ambiguous.

In paper [17], we see that the researchers improve the detection of cyberbully detection using collaborative computing. Their result indicate an improvement in time and accuracy of the detection mechanism over standalone paradigm. They created two datasets and both consisted from tweets from twitter. A balanced dataset using 170 bullying and equal non-bullying contents. Another was unbalanced using 177 bullying and 1163 non bullying contents. Then they applied Naïve Bayes, SVM and Logistic Regression machine learning technics with word tokenizer and bigram tokenizer parameter settings. With balanced dataset, Logistics Regression performed little better than others. More than 60% precision recall and accuracy. Naïve Bayes was close to Logistic regression and SVM had better recall but bad accuracy and precision. With the unbalanced data, Logistics Regression again performed wonderfully with more than 30% correct predictions in average where as In Naïve Bayes the values had dropped and SVM failed. After that collaboration method AND parallelism, OR parallelism and Random 2 Or parallelism were used to see any improvement on precision, accuracy and recall. Among the techniques used AND parallelism had the best accuracy and OR parallelism had the best recall and 7 out of 15 cases using collaboration techniques worked better than their sequential counterpart. This paper gave some new insights how to improve the result using collaboration techniques after using the machine learning techniques to detect cyberbullying. But they mentioned that the results achieved

were without any tuning to the algorithms used so if the algorithms were a little edited maybe the result would have been much better. One interesting future work of theirs mentioned was that the history of two twitter accounts were not considered which obviously plays a vital role in detecting cyberbullying. This is one of our concerns also. SVM classifier performed poorly in this research but in most other papers SVM was defined as the best, so if SVM was tuned and used in kernel it might have performed much better.

Paper [24] detected cyberbullying using support vector machine. As data set, the have used Facebook posts. Those posts were harvested from Facebook by using a web scrapper tool, Selenium. It is a customized tool basically. Selenium used Chrome driver and open the URL for doing queries for Facebook login, then requested data in the form of HTML format, and then parsed it to get the needed data. To harvest all the data they used a step called scroll event before parsing. After that, they preprocessed the harvested data. This step includes removing special characters, URLS, identical Facebook with similar text content and images and symbols from posts. Then they labeled the dataset. Posts then examined using TF-IDF and countvectorizer. For classification by SVM, they used scikit-learn to perform 10 fold cross validation to compute result. Their accuracy rate was 88% after training 2263 data.

Paper [25] aimed to detect cyberbullying actors in twitter based on texts and the credibility analysis of user and also notify them about the harm of cyberbullying. They collected the dataset from twitter. The labelled the data by building a web-based labelling tool. Their data labelling system includes registration of participants, adding negative words and swear words, calculating labeling score and updating corpus and finally labelled tweet as negative word corpus and swear word corpus. After that the preprocessed the data by tokenizing, removing symbols, number etc. Then they extracted the features and the result of this step is formed as a table. Finally, they trained the data to develop SVM and KNN. After detecting cyberbullying by these two model they found that SVM with RBF kernel(c=4) results in the highest f1-score, 67%. SVM with linear kernel and KNN is less than that of RBF kernel. During

feature extraction, they measures the credibility of users and found 257 normal users, 45 harmful bullying users, 53 bullying actors and 6 prospective bullying actors.

Table 1: Literature Review

Ref. No.	Dataset	Methods	Best Results	Limitations
[8]	Arabic Tweet conversations	Lexicon Based Approach	ambiguous	-labels few non-bullying words as bullying -Arabic lexicon doesn't exist
[9]	Comments from accounts of Indonesian Selebgrams	SVM	79.412% accuracy	Use of kernels might give better result
[10]	Comments from accounts of Banagladeshi Facebook celebrities	-MNB -RF -SVM	SVM linear kernel including TF-IDF	spelling checker was not implemented
[11]	Kaggle.com	-SVM -Naïve Bayes	SVM with poly kernel(99.41 %) and 5-gram	shortened words and spelling check was not handled
[12]	Instagram and Twitter messages	-SVM -NBM -C4.5 -K-NN	NBM with IG	-stemming and spelling check wasn't done -Male and female partitioned dataset was not of any use

[13]	Instagram posts of several account	Machine Learning classifier with bigram, unigram	Bigram gave highest accuracy	no certain classifier the mentioned
[14]	Arabic texts from Twitter and Facebook	-SVM -NB - Decision Tree -K-NN	ambiguous	No implementing
[15]	Facebook public pages of Italian newspapers, artists, politicians	SVM and LSTM	SVM for two-class (80.60%) and three-class (64.61%)	The result of three-class dataset is not satisfactory
[16]	Texts or posts from Twitter	SVM with Embedding's Bag of Words (EBoW)	EboW (Precision-76.8%, Recall 79.4%, F1 score 78.0%)	Didn't classify the dataset
[17]	Tweets from Twitter	-Logistic Regrssion -SVM -Naïve Bayes	Logistic Regression with AND parallelism	-No fine tuning to the classifiers -SVM kernels should have been used
[24]	Facebook Posts	-Selenium scrapper tool -SVM	SVM with 88% accuracy	Didn't try any other models which might give better result
[25]	From twitter user's account	-SVM linear -SVM RBF	SVM RBF with f1 score 67%	Dataset was too little. More dataset might be added

Chapter 5

Contribution of Thesis

5.1 Dataset Collection:

In this study, we collected the dataset from 'dataturks.com' which contains 1900 comments in total. Among them, 11162 comments have cyberbullying contents and other 7839 comments are not labeled as cyberbully. We split the dataset as 80% training data and 20% test data.

5.2 Platform and Tools: We are going to implement our classifier using python with Jupyter Notebook and Scikit-learn library. It is a free machine learning software which features various classification methods like SVM, Random Forest etc.

5.3 Cleaning and Preprocessing:

The dataset was a bit messy so it needed to be cleaned at first. Our dataset cleaning phase includes;

- Punctuations and special characters removal
- Digit or any numeric value removal
- Stop words removal
- Words with less than 3 characters removal
- 10 most frequent words removal

After cleaning completion, we preprocessed the dataset. Our preprocessing phase includes;

- Tokenizing
- Lemmatizing
- Stemming

We used the 'nltk' library for preprocessing the data. Tokenizing is the task of splitting up a sentence into pieces and throwing away punctuation characters.

The words like ‘go’, ‘goes’, ‘going’ indicate the same activity. So, we can replace all these words by a single word ‘go’. This process is called stemming. We used Porter Stemmer, which is a famous stemming algorithm.

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Fig 5.1: Porter Stemmer

Lemmatization is similar to stemming but it brings context to the words. So, it goes a step further by linking words with similar meaning to one word. For example, if a paragraph has words like cars, trains and automobile, then it will link all of them to automobile. We used ‘Wordnet’ lexical analyzer for lemmatization.

For better visualization of our dataset, we used ‘wordcloud’ package. We plotted figures with the most used words in bullying content and in non-bullying content separately.

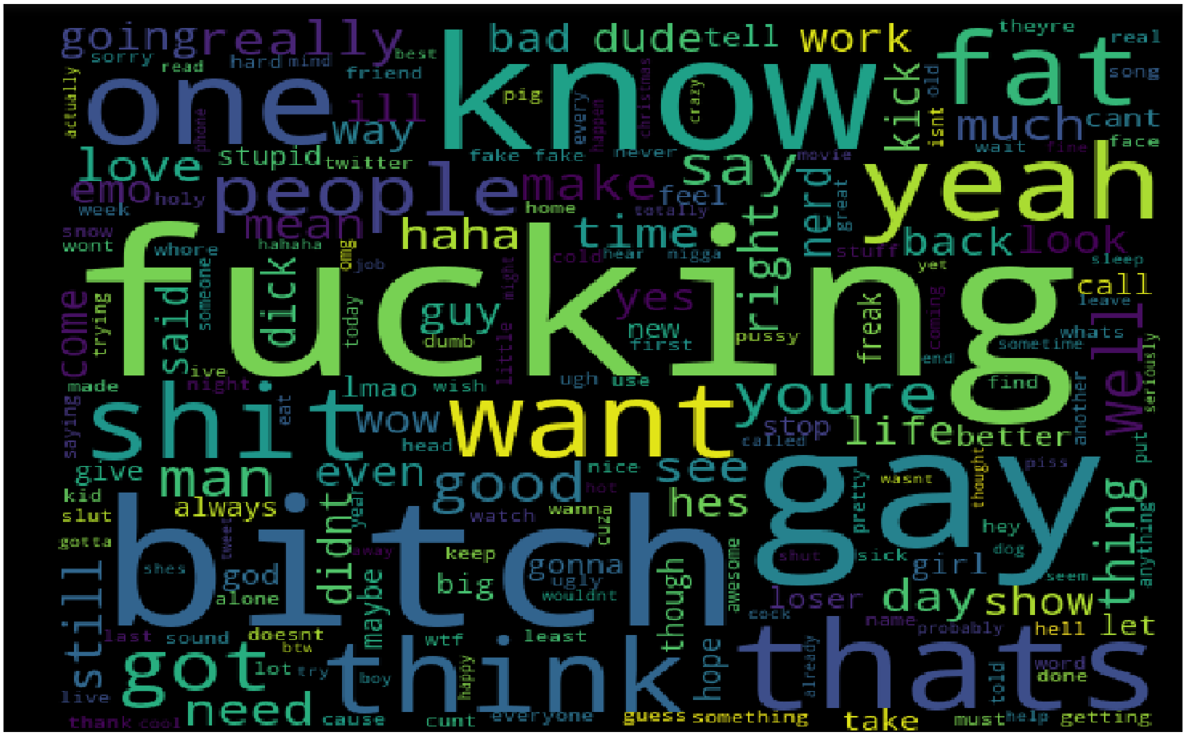

Figure 5.2: Most used words in bullying contents

Fig 5.3: Most used words in non-bullying contents

5.4 Processing:

Bag of Words (BoW):

After all the preprocessing tasks, we vectorized the tokens with CountVectorizer() and TfidfVectorizer() of 'sklearn'. But the CountVectorizer() is comparatively better suited for our dataset than the latter. CountVectorizer() is actually a representation of 'Bag of Words' model. This model uses 'Term Frequency' which indicates the number of occurrences of each word in the dataset.

We also introduced n-grams which is a contiguous sequence of n words. We examined with different n-grams. But n-gram 1 and 2 gave us the best output.

5.5 Classification:

Now the classification part comes. We used 'scikit-learn' to import classification algorithms. We classified our dataset with many classifiers like Naïve Bayes, Logistic Regression, Decision Tree, SVM etc. Some of the classifiers were quite efficient but some other were not up-to the mark. The F-Score, Recall and Precision are represented in the result analysis part below.

5.6 Result Analysis:

Naïve Bayes: We used Multinomial Naïve Bayes as one of our classifier. It gave 85.6% accuracy rate in detecting cyberbullying.

Logistic Regression: Using logistic regression we got an accuracy of 87.2%.

K-NN: We examined K-NN classifier with different values of K. But 2-nearest neighbors earned the best result among them. We also set the weight of the algorithm according to the distance between two points. The leaf size was restricted to 30. Setting all these parameters, we got an accuracy of 89%.

Decision Tree: This classifier worked with less efficiency for our corpus. We set the criterion of the algorithm to 'entropy' and got the result of 84.6% accuracy.

SVM: SVM is one of the best fitted classifier in our work. There are several kernels used in SVM but we implemented 'Linear' and 'RBF'. Linear SVM gained an accuracy of 87.9%. setting the gamma value of the algorithm to 10, SVM with RBF kernel gave us the best result of 93.8% accuracy.

Table: Confusion Matrix description for different Classifiers

	Naïve Bayes		Logistic Regression		K-Nearest Neighbor		Total
	Predicted Non-bullying	Predicted bullying	Predicted Non-bullying	Predicted bullying	Predicted Non-bullying	Predicted bullying	
Actually Non-bullying	1915	316	2017	214	2037	194	2231
Actually bullying	231	1339	273	1297	222	1348	1570
Total	2146	1655	2290	1511	2259	1542	

	Decision Tree		Linear SVM		RBF SVM		Total
	Predicted Non-bullying	Predicted bullying	Predicted Non-bullying	Predicted bullying	Predicted Non-bullying	Predicted bullying	
Actually Non-bullying	1786	445	1963	268	2205	26	2231
Actually bullying	162	1408	192	1378	211	1369	1570
Total	1948	1853	2155	1646	2416	1395	

	Naïve Bayes	Logistic Regression	Decision Tree	SVM Linear	SVM RBF	k-nn
Accuracy	0.86	0.87	0.85	0.88	0.94	0.89
F-Score	0.86	0.87	0.85	0.88	0.94	0.89
Precision	0.86	0.87	0.86	0.88	0.94	0.89
Recall	0.86	0.87	0.85	0.88	0.94	0.89

Fig 5.4: Outcomes for each classifier model

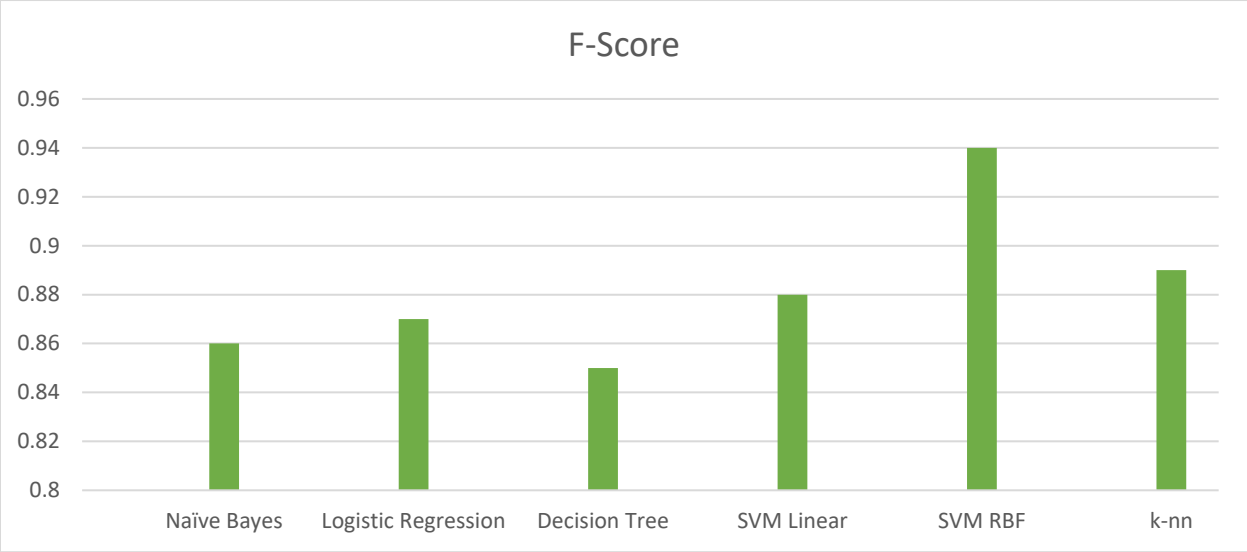

Fig 5.5: F-Score for each classifier model

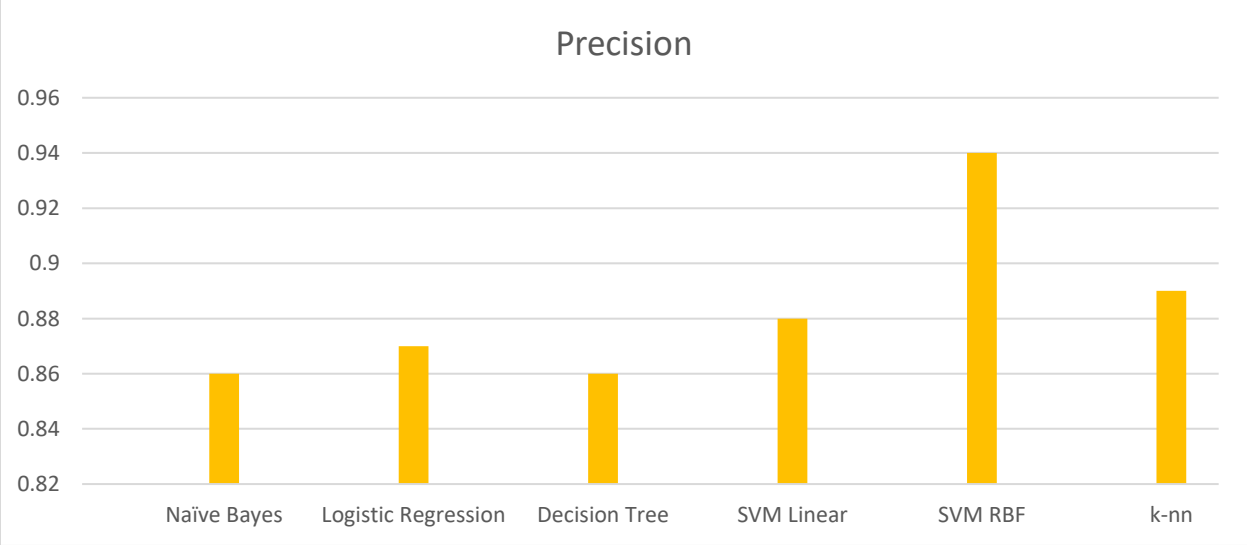

Fig 5.6: Recall for each classifier model

Fig 5.7: Precision for each classifier model

5.7 Experimentation:

We experimented the tokens with their bullying ratio to identify the impact that each one of them had on a sentence. Here is a graph to visualize the bullying ratio of 10 random tokens.

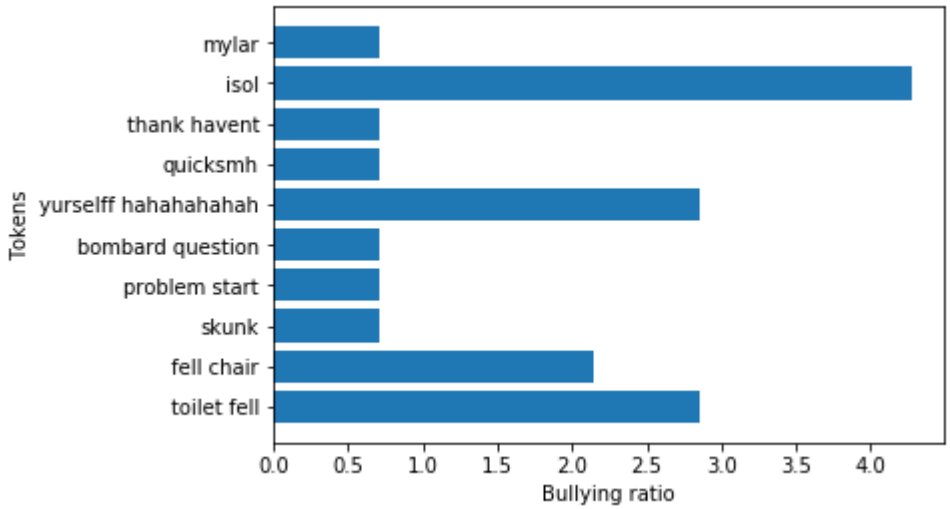

Fig 5.8: Bullying ratio of 10 random tokens

To understand the model better, we classified some sentences from ourselves, not from the corpus. It gave us good monitoring power through the model we created. This shows the behavior of the model if any sentence is added to the testing data.

As an example, we provided two sentences, ‘You are so dumb’ and ‘I feel proud of you’. As a result, the model gave us the correct output that the first one should be labelled as cyberbullying but the latter is non-cyberbullying.

Chapter 6

Conclusion

For our thesis work, to detect cyberbullying from English texts, we chose a quite balanced dataset that had reasonable amount of data; cleaned and preprocessed it. We used Bag of Words model to fit and transform our data into machine recognizable form. Extracted the best features by using suitable methods. Split the data and applied various supervised machine learning classifying algorithms such as Support Vector Machine, Logistic Regression, K-Nearest Neighbor, Naïve Bayes and Decision Tree to detect performance. We tweaked and experimented with our data and model to achieve best accuracy possible. Eventually, we got the best result using Support Vector Machine using 'RBF' kernel.

Chapter 7

Future Work

Now, we are only detecting whether it's cyberbullying or not. Our accuracy is quite satisfactory but we will try more data manipulation, preprocessing and also hybrid classifying classifiers to increase the accuracy if possible. We might even try using different approach for detecting cyberbullying if needed. Later, we will also try to measure the severity level of the bullying and detect the sentiment behind the comments. Some comments may seem like a bully but they actually are not. Some comments are posted with a view to criticizing constructively which are appreciable. So, these types of comments shouldn't be detected as cyberbullying.

Chapter 8

References

1. "Cyberbullying: Which 3 Social Networks Are the Worst? "www.momsteam.com. Retrieved 2018-04-09.
2. Cyberbullying – Law and Legal Definitions US Legal.
3. Hinduja, S., Patchin, J. W. (2009). Bullying beyond the schoolyard: Preventing and responding to cyberbullying. Thousand Oaks, CA: Corwin Press. ISBN 1-4129-6689-2.
4. https://www.thedailystar.net/bytes/%E2%80%9849-bangladeshi-school-pupils-face-cyberbullying%E2%80%99-287209 (LAT*- 29 June, 09:57pm)
5. https://sdasia.co/2016/02/10/cyber-bullying-in-bangladesh (LAT*- 29 June, 10:01pm)
6. https://bdnews24.com/bangladesh/2017/03/09/73-percent-women-subject-to-cyber-crime-in-bangladesh (LAT*- 29 June, 10:23pm)
7. W. D. Daelemans et al., "Automatic Detection and Prevention of Cyberbullying," The First International Conference on Human and Social Analytics, 2015.
8. Alanood Hamad Alduailej, Dr.Muhammad Badruddin Khan. “The Challenge of Cyberbullying and its Automatic Detection in Arabic Text.” 2017 International Conference on Computers and Application (ICCA) by IEEE.
9. Miftah Andriansyah, Ali Akbar, Afina Ahwan, Ardiono Roma Nugraha, Nico Ariesto Gilani, Rizki Nofita Sari and Remi Senjaya. “Cyberbullying Comment Classification on Indonesian Selebgram Using Support Vector Machine Method”. 2017 Second International Conference on Informatics And Computing By IEEE.
10. Shahnoor C. Eshan, Mohammad S. Hasan. “An application of Machine Learning to Detect Abusive Bengali Text”. 2017 20th International Conference of Computer and Information Technology (ICCIT) By IEEE.
11. Noviantho, Sani Muhamad Isa, Livia Ashianti.”Cyberbullying Classification using Text Mining”. 2017 1st International Conference on Informatics and Computational Sciences (ICICoS) by IEEE.

12. S. A. Özel, E. Saraç, S. Akdemir and H. Aksu, "Detection of cyberbullying on social media messages in Turkish". 2017 International Conference on Computer Science and Engineering (UBMK) by IEEE.
13. Mohammadreza Rezvan , Saeedeh Shekarpour , Lakshika Balasuriya , Krishnaprasad Thirunarayan , Valerie L. Shalin , Amit Sheth, “A Quality Type-aware Annotated Corpus and Lexicon for Harassment Research” , Proceedings of the 10th ACM Conference on Web Science, May 27-30, 2018, Amsterdam, Netherlands.
14. B. Haidar, M. Chamoun and F. Yamout, "Cyberbullying Detection: A Survey on Multilingual Techniques," 2016 European Modelling Symposium (EMS).
15. F Del Vigna12, A Cimino23, F Dell'Orletta, M Petrocchi,”Hate Me or Hate Me Not: Hate Speech Detection on Facebook”.
16. Rui Zhao, Anna Zhou, Kezhi Mao,“Automatic detection of cyberbullying on social networks based on bullying features”. ICDCN '16 Proceedings of the 17th International Conference on Distributed Computing and Networking.
17. Amrita Mangaonkar, Allenoush Hayrapetian, Rajeev Raje. “Collaborative Detection of Cyberbullying Behavior in Twitter Data”. 2015 IEEE International Conference on Electro/Information Technology (EIT).
18. https://en.wikipedia.org/wiki/Support_vector_machine(LAT*- 19 Nov, 10:01pm)
19. https://www.expertsystem.com/machine-learning-definition(LAT*- 19 Nov, 10:01pm)
20. https://machinelearningmastery.com/logistic-regression-for-machine learning(LAT*- 21 Nov, 7:09pm)
21. https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc(LAT*- 19 Nov, 10:01pm)
22. https://www.svm-tutorial.com/2014/10/svm-linear-kernel-good-text-classification (LAT*-23 Nov,8.42pm)
23. https://data-flair.training/blogs/svm-kernel-functions(LAT*-24 Nov,5pm)

24. H. Nurrahmi and D. Nurjanah, "Indonesian Twitter Cyberbullying Detection using Text Classification and User Credibility," 2018 International Conference on Information and Communications Technology (ICOI)
25. K. D. Gorro, M. J. G. Sabellano, K. Gorro, C. Maderazo and K. Capao, "Classification of Cyberbullying in Facebook Using Selenium and SVM," 2018 3rd International Conference on Computer and Communication Systems (ICCCS)
26. https://machinelearningmastery.com/prepare-text-data-machine-learning-scikit-learn (LAT*-25 Nov, 7.27 pm)