Classification of Cyberbullying on Twitter

Patryk Maik [redacted]

Abstract

The cyberbullying issue is rapidly increasing among social media platforms. It is a factor that could contribute to depression or even suicide. There are a lot of social media platforms with billions of users worldwide. Human limitations in the ability to process big data encourage the usage of computers and machine learning to solve this issue. The proposed ensemble methods include combining machine learning algorithms and using a majority vote to obtain the best result among classifiers. The first approach combined seven algorithms as follows Multinominal Naive Bayes (MNB), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVC), Gradient Boosting (GB), Ada Boosting (ADA) and Bagging. The second voting classifier combined only five algorithms as follows Multinominal Naive Bayes (MNB), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVC), and Random Forest (RF) with parameter tunning using Grid Search. The best result was accomplished by the first approach where the bag of words was used for feature extraction and four cross-validation techniques Kfold, StratifiedKfold, ShuffleSplit, and StratifiedShuffleSplit for data splitting equally yield 94% of accuracy. Both of the voting classifiers performed well for this text classification issue. The difference in performance was only 1% whereas second approach was limited due to computational requirements.

1. Introduction

Cyberbullying is a common issue among current social media platforms. A manual approach for searching for bullying comments online will require a significant amount of time and money. Machine learning is a core branch of Artificial Intelligence that allows computers to learn to think like humans (Xue and Zhu 2009). Supervised learning is equivalent to humans learning from past experience (Liu 2011). Unlike humans, machines learn from data that is relevant to the problem domain. Natural Language Processing(NLP) is processing text that occurs naturally by using a vast number of computational methods(Liddy 2001). The main goal of NLP is to recreate computer-human-like language processing that could be applied to different assignments and deployed to applications. Term Frequency Inverse Document Frequency (TF-IDF) finds applicability in text relevancy recognition that distinguishes the importance of words, sentences etc in a document (Yun-tao, Ling, and Yong-cheng 2005). Ensemble learning is a method that combines multiple models to obtain more reliable results. One of the ensemble learning examples is voting. This approach takes into consideration the majority vote whereas each of the classifiers contributes the same by a single vote. The class that is most frequent is the final prediction(Dogan and Birant 2019).

2. Background literature

There are many approaches for interrogating the issue of sentiment analysis. Authors in (Wisesty et al. 2021) implemented different techniques to conquer this problem. Twitter data was studied with three approaches, bag of words and TF-IDF with Support Vector Machine, word embedding word2vec and Glove with Long Short-Term Memory, and Bidirectional Encoder Representations from Transformers BERT. Among all tested models, BERT outperformed the other two with an 85 % F1-score of three sentiment classes. Another study was conducted by (Hossain, M. A. R. Talukder, and Jahan 2021) where seven different supervised learning algorithms were tested on tweeter data to predict depression. The worst performance was noted by Valence Aware Dictionary for Sentiment Reasoning VADER to analyze the polarity of each sentence. Other algorithms include Multinominal Nayive Bayes (MNB), Linear (SVC), K-nearest neighbour (KNN), Random Forest (RF), Decision Tree (DT) and Logistic Regression (LR) gave decent results with MNB and LR performing the best 95 % of accuracy. The VotingClassifier methods presented by (Alam, Bhowmik, and Prosun 2021) were built from machine learning models including MNB, LR, DT and LinearSVC and three ensemble methods GBoost, AdaBoost and Bagging. The Bag of words and TF-IDF with four different techniques n-gram models Unigram and Bigram and Word and Character as analyzers were used to extract features. The VotingClassifier made out of combined models outperformed singular models with an accuracy of 96 % where TF-IDF Unigram was used as feature extraction

and K-fold as a cross-validation technique. Evaluation of machine learning models that were used to detect cyberbullying with TF-IDF feature extraction was conducted by (Rahman, K. H. Talukder, and Mithila 2021). The best model was the Random Forest classifier which achieved 89 % of accuracy. Another text classification study where supervised learning methods were compared to the ensemble methods proved that the Random Forest ensemble classifier outperformed other models with 92 % of accuracy (Agrawal and Chakravarthy 2022). Two embeddings TF-IDF and BoW for text classification on Twitter were compared with the usage of four different algorithms LR, NB, SVM and RF. All algorithms were performing better when TF-IDF embedding was applied (Mitra et al. 2021).

3. Methods

The development of machine learning ensemble models was grouped into the following stages: data collection, data preprocessing, dataset splitting, feature extraction, building classifier and predicting results.

3.1. Data collection

The data we used for this study was downloaded from Kaggle and it was created by (Wang, Fu, and Lu 2020). The dataset contains more than 47000 tweets that were labelled accordingly to the class of cyberbullying as follows Age, Ethnicity, Gender, Religion, another type of cyberbullying and not cyberbullying. Each of the aforementioned classes has 8000 examples.

3.2. Data Preprocessing

In the dataset, all emojis were removed from the data with the emoji 1.7.0 package in python. Target variables were converted into numbers. Punctuations, links, stopwords, mentions and new line characters were extracted. The text was cleaned from contractions, hashtag symbols, special characters and additional spaces. The stemming process was implemented to reduce the words in dataset by simplifying them to the root word. For example, the words ”retrieved”, and ”retrieves” would be reduced to their root word form ”retrieve”. The lemmatization process was applied to perform a morphological analysis of the text. Unlike stemming, lemmatization would consider the context of the word to link it into one. For example, the word ”better” would be linked with the word ”good”. Figure 1 is an example of a comparison of the text before and after cleaning.

Length analysis revealed that the majority of the sentences are between 3-40 words long. The entire dataset was reduced to eliminate outliers. After the reduction and cleaning dataset contained 37092 examples.

	text	sentiment	clean_text
0	In other words #katandandre, your food was cra...	5	word katandandr food crapilici mkr
1	Why is #aussietv so white? #MKR #theblock #ImA...	5	aussietv white mkr theblock today sunris studi...
2	@XochitlSuckkks a classy whore? Or more red ve...	5	classi whore red velvet cupcak
3	@Jason_Gio meh. :P thanks for the heads up, b...	5	meh p thank head concern anoth angri dude twitter
4	@RudhoeEnglish This is an ISIS account pretend...	5	isi account pretend kurdish account like islam...

Figure 1. Text data before/after cleaning

3.3. Dataset Splitting

Dataset was split by using the ”train test split” method in Python. The size of the training dataset was set to 80 per cent and the test dataset to 20 per cent. The parameter ”stratify” was set to the target variable to keep the data equally distributed among classes and the parameter ”random state” was set to control the shuffling process in the dataset. Another dataset-splitting was implemented with the usage of cross-validation methods. K-fold cross-validation with 10 folds was applied to evaluate ensemble model performance. This means that the dataset was split into 10 equally sized subsets that were equivalent to test sets whereas the remaining data was used to train the models. The result from each fold performance was gathered and an average taken from these records is the final output. StratifiedKfold cross-validation with 10 folds differs from regular K-fold in class distribution. In StratifiedKfold cross-validation, classes are distributed equally in each fold to test and train sets. ShuffleSplit cross-validation that divides the dataset into random splits. StratifiedShuffleSplit cross-validation which returns stratified randomized splits with the respect to the target variable. All of these techniques were performed on the dataset with different models to find the best split.

3.4. Feature Extraction

Feature extraction was implemented with two techniques Bag of words (BoW) and term frequency-inverse document frequency (TF-IDF). The bag of words takes into consideration the frequency of the words in the dataset. The final output is the vector of numbers where each number depends on the frequency of the word in the sentence. TF measures how relevant is word to a document in a set of documents. IDF represents documents in the corpus that contain the term. The more unique word is to the document the higher value receives. Figure 2 presents how TF-IDF is calculated (Karabiber 2022). TF-IDF for this study was used in four different variations. First two by changing the n-gram parameter to ”Bigram” or ”Unigram”. Last two by initializing the analyzer to ”character” or ”word”.

3.5. Classifiers

The first voting classifier method was constructed using seven machine learning classifiers: MNB, LR, DT, LSVC, GB, ADA and Bagging. The idea for this combination of

$$TF = \frac{\text{number of times the term appears in the document}}{\text{total number of terms in the document}}$$

$$IDF = log(\frac{\text{number of the documents in the corpus}}{\text{number of documents in the corpus contain the term}})$$

$$TF\text{-}IDF = TF * IDF$$

Figure 2. TF-IDF calculation

algorithms was influenced by (Alam, Bhowmik, and Prosun 2021). The second experiment was created by using a voting classifier that combined: MNB, LR, DT, LSVC and RF. The RF classifier was used due to the high scores achieved among many studies in background literature (Rahman, K. H. Talukder, and Mithila 2021) (Agrawal and Chakravarthy 2022).

Multinominal Nayive Bayes algorithm calculates the probability of an event according to the understanding of conditions associated with an event. The formula "P(C—X) = P(C) * P(X—C)/P(X)" calculates the probability "P(C)" where predictor "P(X)" is given. "P(C)" is a prior probability class "C" and "P(X)" prior probability of "X". "P(X—C)" presents how often predictor "X" is in class "A" probability (Leung 2007). The final output of the algorithm is the highest probability achieved and prediction will be based on that assumption.

Logistic Regression examines the connection between independent variables to classify data. Logistic Regression formula:

$$y = \frac{e^{(b0+b1X)}}{1 + e^{(b0+b1X)}} \quad (1)$$

Where: X - input value, y - predicted value, b0 - bias or intercept term, b1 - the coefficient for input (x) (Kanade 2022).

The decision Tree classifier starts from the root node and expands branches depending on a number of conditions grouped. It groups the search area into subsets by 'divide and rule. A tree is built for the modelling during the classification process (Zhong 2016).

The SVC classifier finds the hyperplane to boost the margin connecting the two classes. The aim of the algorithm is to find a hyperplane that separates data accurately (Shuzhanfan 2022).

The Gradient Boosting classifier logic is based on the idea that each predictor corrects its predecessor's error. The residual error of the predecessor's labels is used to train predictors (niki2398 2022).

The AdaBoost fits data on the classifier and then weights of faulty classified data are fixed so the following classifier is able to target more complex examples (scikit-learn 2022).

Bagging produces multiple predictors to obtain an aggregated predictor. The aggregation process takes the versions of predictors and averages them while predicting integer outcomes. In the case of predicting class, a plurality vote is conducted (Breiman 1996).

The Random Forest method uses bootstrap aggregation and randomization of predictors to enhance accuracy. The aim of the algorithm is to combine multiple decision trees to obtain the most efficient results(Rigatti 2017).

The hard voting ensemble method uses a majority vote of the models. Each time model predicts voting takes place and an instance with half or more votes is the output (Atallah and Al-Mousa 2019).

4. Experiments

The dataset for this experiment contained 10 000 records that were taken from the preprocessed dataset of 37 092 tweets. The dataset contained labels as follows Age, Ethnicity, Gender, Religion, and not cyberbullying where each instance has 2000 data records. This dataset was used to train and test models. From the 37 092 tweets, another dataset was created with 500 records whereas each label had 100 records. A smaller dataset was used to perform hyperparameter selection with Grid Search.

4.1. Experimental settings

This experiment will measure the performance of two different voting classifiers. The first classifier combined seven different models that include MNB, LR, DT, SVC, GB, ADA and Bagging. The experiment was influenced by (Alam, Bhowmik, and Prosun 2021) where authors used the same approach for their voting classifier. The second voting classifier contained models such as MNB, LR, DT, SVC, and RF. In this approach, three ensemble methods GB, ADA, and Bagging were replaced with one Random Forest. The RF classifier proved to be efficient in tackling issues of text classification. Each of the models in experiment two was tunned using Grid Search Cross-Validation and then combined. This approach allows models to find ideal parameters to prevent underfitting or overfitting of the models.

4.2. Evaluation criteria

Feature Extraction both experiments were evaluated with the same feature extraction techniques which include four cases for TF-IDF with the following parameters n-gram as unigram or bigram and analyzer as word or character. In addition, evaluation performance was measured with a bag of words feature extraction.

Cross Validation both experiments were evaluated with four cross-validation techniques: Kfold, StratifiedKfold, ShuffleSplit, and StratifiedShuffle. Only an accuracy measure was taken for cross-validation results.

Metrics For the evaluation of the models without cross-validation (only feature extraction), the performance of the model was measured in four metrics. The accuracy measure:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (2)$$

The precision measure is a number of correct positive predictions and can be calculated:

$$Precision = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \quad (3)$$

The recall:

$$Recall = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \quad (4)$$

The F1-score which combines precision and recall:

$$F1score = \frac{2 * \text{Precision * Recall}}{\text{Precision + Recall}} \quad (5)$$

The above-mentioned evaluation metrics are valid for machine learning models. We choose the accuracy metric to measure model performance overall. This method is not ideal for imbalanced data as it might show inaccurate results. Although the dataset is well-balanced (2000 data records in each class), the additional metrics will assure that model is evaluated well from each perspective.

Only the accuracy metric was used for cross-validation evaluation. This evaluation includes only accuracy as the cross-validation is assuring that robustness of the model is preserved.

4.3. Results

Voting classifier 1: MNB, LR, DT, SVC, GB, ADA and Bagging this method proved to perform well on the Twitter dataset (Alam, Bhowmik, and Prosun 2021). The results are shown in Figure 3 column V1 without cross-validation and Figure 4 column V1 with cross-validation. The best performance without cross-validation was noted for the bag of words feature extraction with an accuracy of 93%, F1-score 93%, precision 94%, and recall 93%. The best performance in all cross-validation techniques was 94% equally using a bag of words feature extraction. The same accuracy was estimated by ShuffleSplit and StratifiedShuffleSplit cross-validation using TF-IDF word and unigram.

Metric	Feature Extraction		V1	V2
Accuracy	BOW		**0.93**	0.92
	TF-IDF	Word	**0.93**	0.92
		Character	0.71	0.71
		Unigram	**0.93**	0.92
		Bi-gram	0.76	0.78
F1-Score	BOW		**0.93**	0.92
	TF-IDF	Word	**0.93**	0.92
		Character	0.70	0.70
		Unigram	**0.93**	0.92
		Bi-gram	0.77	0.80
Precision	BOW		**0.94**	0.92
	TF-IDF	Word	0.93	0.92
		Character	0.71	0.71
		Unigram	0.93	0.92
		Bi-gram	0.87	0.87
Recall	BOW		**0.93**	0.92
	TF-IDF	Word	**0.93**	0.92
		Character	0.71	0.71
		Unigram	**0.93**	0.92
		Bi-gram	0.76	0.78

Figure 3. Results without cross validation

Voting classifier 2: MNB, LR, DT, SVC. and RF in this approach before algorithms were combined the hyperparameter selection was performed for each model using Grid Search Cross-Validation. The results for this voting classifier are shown in Figure 3 column V2 without cross-validation and Figure 4 column V2 with cross-validation. The best performance in Figure 3 was accomplished by the bag of words and TF-IDF using the word analyzer or n-gram unigram for feature extraction with 92% of accuracy, F1score, precision and recall. In Figure 4 the best result was 93% of accuracy in **Kfold** with the bag of words, **StratifiedKfold** with the bag of words and TF-IDF word and unigram, **ShuffleSplit** with TF-IDF word, and **StratifiedShuffleSplit** with the bag of words and TF-IDF unigram.

4.4. Discussion

My experiment The most accurate feature extraction method for the ensemble technique text classification task is the bag of words. This method performed the highest scores in both of the voting classifiers V1 and V2 measured by accuracy, F1-score, precision, and recall. The cross-validation technique that has the highest number of top scorers was StratifiedShuffleSplit.

Cross Validation Technique	Feature Extraction		V1	V2
			Accuracy	Accuracy
K-fold	BOW		0.94	0.93
	TF-IDF	Word	0.93	0.92
		Character	0.71	0.71
		Unigram	0.93	0.92
		Bi-gram	0.76	0.79
Stratified K-fold	BOW		0.94	0.93
	TF-IDF	Word	0.93	0.93
		Character	0.71	0.71
		Unigram	0.93	0.93
		Bi-gram	0.77	0.79
Shuffle Split	BOW		0.94	0.92
	TF-IDF	Word	0.94	0.93
		Character	0.71	0.70
		Unigram	0.94	0.92
		Bi-gram	0.77	0.79
Stratified Shuffle Split	BOW		0.94	0.93
	TF-IDF	Word	0.94	0.92
		Character	0.71	0.71
		Unigram	0.94	0.93
		Bi-gram	0.77	0.79

Figure 4. Results with cross-validation

Team experiments We work in a team of four to examine different approaches towards cyberbullying text classification problem on Twitter data. The final results are shown below:

Table 1. Best models results in the team

Case	BERT	LSTM	SVM rbf	Voting
Acc	94%	93%	93%	94%

This study proved that the Voting classifier and BERT algorithm performed the best among the examined models for the text multiclass classification problem.

5. Conclusion

In this study report, two different voting classifiers were compared whereas each combined different machine learning models. The bag of words feature extraction was the most efficient as both algorithms showed the best results using that method. The StratifiedShuffleSplit cross-validation technique was the most accurate as it recorded the highest scores among all cross-validation techniques. Voting classifier 1 (V1) outperformed voting classifier 2 (V2) with a difference of 1% in accuracy measure. Although V2 did not accomplish the highest score, there were some limitations that have to be considered. The V2 used only 500 sample data records to perform hyperparameter selection due to computational limitations. If the experiment could be redone by using GridSearchCV on more data, there is a high possibility that V2 would outperform V1. In addition, both classifiers were trained and tested only on the 10 000 data records. The experiment could use a full dataset that contained 37092 records with a faster computer. The methods used in this experiment proved to solve problems efficiently for cyberbullying classification on Twitter. Further improvements could be done to use the full dataset for training and testing and more data for hyperparameter selection.

References

Agrawal, Tanmay and V Deeban Chakravarthy (2022). "Cyberbullying Detection and Hate Speech Identification using Machine Learning Techniques". In: *2022 Second International Conference on Interdisciplinary Cyber Physical Systems (ICPS)*. IEEE, pp. 182–187.

Alam, Kazi Saeed, Shovan Bhowmik, and Priyo Ranjan Kundu Prosun (2021). "Cyberbullying detection: an ensemble based machine learning approach". In: *2021 third international conference on intelligent communication technologies and virtual mobile networks (ICICV)*. IEEE, pp. 710–715.

Atallah, Rahma and Amjed Al-Mousa (2019). "Heart disease detection using machine learning majority voting ensemble method". In: *2019 2nd international conference on new trends in computing sciences (ictcs)*. IEEE, pp. 1–6.

Breiman, Leo (1996). "Bagging predictors". In: *Machine learning* 24.2, pp. 123–140.

Dogan, Alican and Derya Birant (2019). "A weighted majority voting ensemble approach for classification". In: *2019 4th International Conference on Computer Science and Engineering (UBMK)*. IEEE, pp. 1–6.

Hossain, Md Tazmim, Md Arafat Rahman Talukder, and Nusrat Jahan (2021). "Social Networking Sites Data Analysis using NLP and ML to Predict Depression". In: *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, pp. 1–5.

Kanade, Vijay (2022). *What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices Artificial Intelligence*. URL: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/ (visited on 04/18/2022).

Karabiber Lewis, Martin (2022). *TF-IDF — Term Frequency-Inverse Document Frequency*.

Leung, K Ming (2007). "Naive bayesian classifier". In: *Polytechnic University Department of Computer Science/Finance and Risk Engineering* 2007, pp. 123–156.

Liddy, Elizabeth D (2001). "Natural language processing". In.

Liu, Bing (2011). "Supervised learning". In: *Web data mining*. Springer, pp. 63–132.

Mitra, Shutonu et al. (2021). "A Framework to Detect and Prevent Cyberbullying from Social Media by Exploring Machine Learning Algorithms". In: *2021 International*

Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2). IEEE, pp. 1–4.

niki2398 (2022). *ML – Gradient Boosting Artificial Intelligence*. URL: https://www.geeksforgeeks.org/ml-gradient-boosting/ (visited on 09/02/2020).

Rahman, Shagoto, Kamrul Hasan Talukder, and Sabia Khatun Mithila (2021). "An Empirical Study to Detect Cyberbullying with TF-IDF and Machine Learning Algorithms". In: *2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*. IEEE, pp. 1–4.

Rigatti, Steven J (2017). "Random forest". In: *Journal of Insurance Medicine* 47.1, pp. 31–39.

scikit-learn (2022). *AdaBoostClassifier Artificial Intelligence*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html (visited on 2022).

Shuzhanfan (2022). *Understanding the mathematics behind Support Vector Machines Artificial Intelligence*. URL: https://shuzhanfan.github.io/2018/05/understanding-mathematics-behind-support-vector-machines/ (visited on 2022).

Wang, Jason, Kaiqun Fu, and Chang-Tien Lu (2020). "Sosnet: A graph convolutional network approach to fine-grained cyberbullying detection". In: *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 1699–1708.

Wisesty, Untari N et al. (2021). "Comparative study of COVID-19 tweets sentiment classification methods". In: *2021 9th International Conference on Information and Communication Technology (ICoICT)*. IEEE, pp. 588–593.

Xue, Ming and Changjun Zhu (2009). "A study and application on machine learning of artificial intellligence". In: *2009 International Joint Conference on Artificial Intelligence*. IEEE, pp. 272–274.

Yun-tao, Zhang, Gong Ling, and Wang Yong-cheng (2005). "An improved TF-IDF approach for text classification". In: *Journal of Zhejiang University-Science A* 6.1, pp. 49–55.

Zhong, Yurong (2016). "The analysis of cases based on decision tree". In: *2016 7th IEEE international conference on software engineering and service science (ICSESS)*. IEEE, pp. 142–147.