Cyberbullying Detection: Identifying Hate Speech using Machine Learning

Abstract—**Bullying has been prevalent since the beginning of time, It's just the ways of bullying which have changed over the years, from physical bullying to cyberbullying. According to Williard(2004), there are eight types of cyberbullying such as harassment, denigration, impersonation, etc. It's been around 2 decades since social media sites came into the picture, but there hasn't been a lot of effective measures to curb social bullying and it has become one of the alarming issues in recent times.**

In this paper, we present a systematic review of some published research on cyberbullying detection approaches and examine methods to detect hate speech in social media, while distinguishing this from general profanity. We aim to establish lexical baselines for this task by applying supervised classification methods using a manually annoted open source dataset for this purpose. This paper does a comparative study of various Supervised algorithms, including standard, as well as ensemble methods. The evaluation of the result shows that Ensemble supervised methods have the potential to perform better than traditional supervised methods. A number of directions for future work are also discussed.

Index Terms—**Machine Learning, Cyberbullying, Supervised, Ensemble, Hate Speech, Natural Language Processing**

I. INTRODUCTION

Hate speech refers to words whose intent is to create hatred towards a particular group, that group may be a community, religion or race. This speech may or may not have meaning, but is likely to result in violence. Hate speech online has been linked to a global increase in violence toward minorities, including mass shootings, lynchings, and ethnic cleansing.

Due to the massive rise of user-generated web content, particularly on social media networks, the amount of hate speech is also steadily increasing. Over the past years, research into cyberbullying detection has increased, due in part to the proliferation of cyberbullying across social media and its detrimental effect on the younger generation. A growing body of work is emerging on automated approaches to cyberbullying detection. These approaches utilise machine learning and natural language processing techniques to identify the characteristics of a cyberbullying exchange and automatically detect cyberbullying by matching textual data to the identified traits.

Natural language processing focusing specifically on this phenomenon is required since basic word filters do not provide a sufficient remedy: What is considered a hate speech message might be influenced by aspects such as the domain of an utterance, its discourse context, as well as context consisting of co-occurring media objects (e.g. images,videos, audio), the exact time of posting and world events at this moment, identity of author and targeted recipient.

This paper provides a comprehensive and structured overview of automatic hate speech detection, and compares few of its current approaches in a systematic manner, along presenting an insightful review of some published research on cyberbullying detection approaches.

II. RELATED WORK

For Detecting Cyberbullying, numerous approaches have been developed, majorly using Natural Language Processing and Information Retrieval which are then used to classify textual data by extracting it's features by using TF-IDF, Sentiment Analysis, Dimensionality Reduction etc. and they have received commendable accuracies.

Sambhagadi et al. [2] tries ways to detect nastiness on social media using NLP techniques to detect and deter cyberbullying eventually. NLP techniques are used in such a way that they can even detect when profanities in the data are used in an insulting way or in a neutral way. Annotations used in the paper are iteratively revised using in lab annotations and crowdsourcing. Data was crawled from English posts on social media sites even including semi-anonymous social media sites such as ask.fm. A ranked list of profanities along with NLP helped in crawling in an effective way. To classify the data, modified linear SVM was used to distinguish bad words in a casual way, multiple other features that could have gone unnoticed were also considered such as, Question answer posts and Emoticons. In the end, F1-Score came out to be 0.59 (which although is less than the Kaggle's winner but considering the fact that this study didn't use customized data and a new and better dataset, F1-Score of 0.59 still looks promising). Challenges faced in this study were -

- In ask.fm, comments are question-answer pairs which are shorter in other datasets and both question-answer may contain only one word making it hard for the algorithm to classify without understanding the full context.
- People use informal language and slang on social media which are full of misspellings and abbreviation, making processing them very difficult

Acknowledging the repetitive nature of cyberbullying on social media i.e. a sequence of aggressive messages sent from bully to a victim with the intent of harm, the paper by Yao et al. [3] uses sequential hypothesis testing formulation to drastically reduce the number of features used in classification, while still maintaining high accuracy. This approach focuses High accuracy, Timeliness, and scalability. Models are trained using semi-supervised ML algorithms, using an Instagram dataset collected using snowball sampling, labeled manually(to a small extent) by a group of experts. The limitation of this approach was the use of a single data set that was only valid for Instagram, with no way to check the validity of labels, and the time overhead due to difficulty in capturing comment based labels.

Huang et al. [4] focuses on analyzing the social network structure between users and deriving features such as a number of friends, network embeddedness, and relationship centrality, by integrating textual features with social network features, detection of cyberbullying can be achieved. The study claims that past researches haven't fully utilized social media features, this paper proposes cyberbullying detection beyond textual analysis to also consider the social relationships in which these bullying messages are exchanged. It uses twitter corpus from Dec 2008 to Jan 2009 and uses SMOTE(synthetic minority oversampling technique) approach to create a balanced data, with Naive Bayes, J48, SMO, Bagging and dagging being applied on it.

TABLE I
RESULTS OF [4]

	ROC	TP
Bagging	0.700	0.211
J48	0.628	0.259
SMO	0.703	0.733
Dagging	0.755	0.763
Naive Bayes	0.695	0.723

Using Reddit's comment corpus for cyberbullying detection, Rakib et al. [5] extracted the corpus and cleaned it from the Reddit database, followed by training a word embedding model based on word2vec skip-gram model. Then, the features of this model were used to train a random forest classifier for classifying cyberbully comments, This new word embedding model made using domain knowledge performed better than 4 pre-trained word embedding models, as well as handcrafted feature extraction methods.

Silva et al. [6] proposed a model for cyberbullying identification that uses research based on psychology; it describes the design for an app referred as BullyBlocker, which aims to intimate the parents of the user if cyberbullying is detected. It uses traditional methods to analyze social media data of the user by going through their messages and comments and rank them as warning signs or give them a bullying rank. It is specifically made for adolescents and uses old methods for detection in Facebook, but it has the potential to grow by acting as a data-collecting app over which ML classification can be run.

III. DATA

This section contains all the aspects of Data from collection to preprocessing and features extraction.

A. Data Collection

We have used Dataturks' Tweet Dataset for Cybertroll Detection obtained from Kaggle [1] for reaching the final results. Because of the seriousness of the issue we aim to resolve, it was crucial to choose a dataset that was complete, reliable, relevant, and to the point. While we considered many other datasets as well, many of them either had missing attributes, were too low in quality, or were found to have irrelevant data after manual inspection. Thus, after having tried out of many other open sourced datasets, we came down to [1] as it seemed in line with all the parameters required.

Here is the Detailed Description of the dataset:

1) It is a partially manually labelled dataset.
2) Total Instances: 20001

The dataset has 2 attributes- tweet and label [0 corresponds to No while 1 corresponds to Yes]

B. Data Cleaning

The dataset used was set in a json format. Since the fields of the dataset were relatively simple to interpret, the original set of fields in the annotation attribute was removed, and filled with the label values to simplify the next step. The number of instances for each class are mentioned in table 1.

TABLE II
INSTANCES

	Twitter
Total Instances	20001
CyberBullying instances	7822
Non-CyberBullying instances	12179

C. Data Preprocessing

The preprocessing steps were done as follows using the nltk library along with regex:

1) Word Tokenization: A Token is a single entity that is building blocks for sentence or paragraph. Word Tokenization converts our text to separate words in a list.
2) Stop words filtering is done using nltk.corpus.stopwords.words('english') to fetch a list of stopwords in the English dictionary, after which they are removed. Stop words are words such as "the", "a", "an", "in", which are not significant and do not affect the meaning of the data to be interpreted.
3) To remove punctuation, we save only the characters that are not punctuation, which can be checked by using string.punctuation .
4) Stemming: Stemming is a process of linguistic normalization, which reduces words to their word root word. We stem the tokens using nltk.stem.porter.PorterStemmer to get the stemmed tokens. For example, connection, connected, connecting word reduce to a common word "connect".
5) Digit removal: We also filtered out any numeric content as it doesn't contribute to cyberbullying.
6) Now the next step was to extract features so that it can be used with ML algorithms, for which we used TF-IDF Transform using Python's sklearn libary. TF-IDF is a statistical measure to evaluate the relevance of a word, which is basically calculated by multiplying the number of times that words appeared in the document by the inverse document frequency of the word. TF-IDF uses the method diminishing the weight (importance) of words appeared in many documents in common, considered them incapable of discerning the documents, rather than simply counting the frequency of words as CountVectorizer does. The outcome matrix consists of each document (row) and each word (column) and the importance (weight) computed by tf * idf (values of the matrix). If a word has high tf-idf in a document, it has most of the times occurred in given documents and must be absent in the other documents. So the words must be a signature word.

TABLE III
TEST AND TRAINING INSTANCES

	Test	Training
Total Instance	4001	16000
CyberBullying Instances	2429	9750
Non-CyberBullying instances	1572	6250

Attribute evaluation is done manually as can be seen where we have printed the top 25 words according to the calculated tf-idf score. Some Top ranked words for the dataset were: [hate, fuck, damn, suck, ass, that, lol, im, like, you, it, get, what, no, would, bitch]

D. Data Resampling

As the data was skewed, Resampling had to be performed on the training data, Firstly the data was split into Training and Test in 80:20 ratio and resampling was performed on the training data.

- As we had ample data to work with, we used oversampling of the minority class. This means that if the majority class had 1,000 examples and the minority class had 100, this strategy would oversampling the minority class so that it has 1,000 examples.
- For Oversampling, RandomOverSample function is used from imblearn package for all the "not majority" classes which in our case, was only the 1 minority class.

After resampling, the training data had 9750 CB & NON-CB instances.

IV. DESCRIPTION OF METHODS

A. Gaussian Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem of mathematics. In simple words, the Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values. It is called naive Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable.

Naive Bayes can be extended to real-valued attributes, most commonly by assuming a Gaussian distribution. This extension of Naive Bayes is called Gaussian Naive Bayes. Beside the Gaussian Naive Bayes there are also existing the Multinomial naive Bayes and the Bernoulli naive Bayes. We picked the Gaussian Naive Bayes because it is the most popular one and one of the simplest to implement because we only need to estimate the mean and the standard deviation from the training data.

The classifier was implemented using *sklearn.naive_bayes* package.

B. Logistic Regression

Regression analysis is a predictive modelling technique that analyzes the relation between the target or dependent variable and independent variable in a dataset. Regression analysis techniques get used when the target and independent variables show a linear or non-linear relationship between each other, and the target variable contains continuous values. Regression analysis involves determining the best fit line, which is a line that passes through all the data points in such a way that distance of the line from each data point is minimized.

Logistic regression is one of the types of regression analysis technique, which gets used when the dependent variable is discrete. Example: 0 or 1, true or false, etc. This means the target variable can have only two values, and a sigmoid curve denotes the relation between the target variable and the independent variable, by mapping any real value to a value between 0 and 1. We chose Logistic Regression as the size of our data set was large, and it had almost equal occurrence of values to come in target variables. Moreover, there was no correlation between independent variables in the dataset.

The classifier was implemented using *sklearn.linear_model* package.

C. Decision Tree Classifier

A Decision Tree is constructed by asking a series of questions with respect to the dataset. Each time an answer is received, a follow-up question is asked until a conclusion about the class label of the record. The series of questions and their possible answers can be organised in the form of a decision tree, which is a hierarchical structure consisting of nodes and directed edges. It has 3 types of nodes: Root, Internal, and Leaf nodes.

In a decision tree, each leaf node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics.

Using the decision algorithm, we start at the tree root and split the data on the feature that results in the largest information gain (IG) (reduction in uncertainty towards the final decision). In an iterative process, we can then repeat this splitting procedure at each child node until the leaves are pure. This means that the samples at each leaf node all belong to the same class.

The classifier was implemented using *sklearn.tree* package.

D. Adaboost Classifier

AdaBoost is an iterative ensemble method. The general idea behind boosting methods is to train predictors sequentially, each trying to correct its predecessor. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set.

At a high level, AdaBoost is similar to Random Forest as they both tally up the predictions made by each decision trees within the forest to decide on the final classification. There however, lie some subtle differences. In AdaBoost, the decision trees have a depth of 1 (i.e. 2 leaves). In addition, the predictions made by each decision tree have varying impact on the final prediction made by the model. Rather than taking the average of the predictions made by each decision tree in the forest (or majority in the case of classification), in the AdaBoost algorithm, every decision tree contributes a varying amount to the final prediction.

The classifier was implemented using *sklearn.ensemble* package.

E. Random Forest Classifier

As its name implies, Random Forest Classifier consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The low correlation between models is the key as they can produce ensemble predictions that are more accurate than any of the individual predictions, as the trees protect each other from their individual errors. The process of Bagging is used to diversify models as each individual tree is allowed to randomly sample from the dataset with replacement.

The classifier was implemented using *sklearn.ensemble* package.

V. Experiment and Results

For our supervised learning technique analysis, we've used Naive Bayes(Gaussian), Logistic regression, and

J48 Decision Tree as the standard methods. As Ensemble methods, we have used AdaBoost and RandomForest Classifiers. In our research, we found that the Gaussian Naive Bayes classifier performed the poorest, whereas the Random Forest Classifier gave the best result in terms of every metric.[fig1 & fig2].

It wasn't surprising to see the Random Forest classifier performing the best. The Decision Tree classifier performed better than Naive Bayes classifier and Logistic Regression. The Random Forest Classifier came out on top in all the performance metrics, which was expected as it is an extension of the Decision Tree classifier, averaging out results of multiple recursions of the same.

The Metrics used for determining the performance of models are as follows:

- $$F - Measure = \frac{(2 \times Precison \times Recall)}{Precision + Recall}$$
- $$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$
- $$Precision = \frac{TP}{TP + FP}$$
- $$Recall = \frac{TP}{TP + FN}$$
- ROCArea, which denotes the area under the curve formed by plotting TP rate.

where,
TP = No. of True Positives
TN = No. of True Negatives
FP = No. of False Positives
FN= No. of False Negatives

Traditional Supervised Learning used: NaiveBayes, Logistic Regression and J48 Decision Trees classifier. The Ensemble Learning Methods used: AdaBoost and Random Forest classifier

Figure 1 and Figure 2 shows a graphical comparison between the aforementioned algorithms.

Note: Table IV represents the weighted average using both the classes(hate speech and non hate speech) for Precision, Recall, and F1 score.

First column and row of the confusion matrices represents Cyberbullying class whereas the second row and column represents Non-cyberbullying class.

TABLE IV
SUPERVISED TRADITIONAL METHODS

	NaiveBayes	Regression	DecisionTree
Accuracy	0.62	0.80	0.85
Precision	0.79	0.81	0.88
Recall	0.62	0.80	0.85
F1-Score	0.59	0.81	0.85
ROCArea	0.68	0.81	0.87
Confusion	925 1504	1920 509	1896 533
Matrix	31 1541	274 1298	67 1505

TABLE V
SUPERVISED ENSEMBLE METHODS

	AdaBoost	Random Forest
Accuracy	0.71	0.92
Precision	0.74	0.92
Recall	0.71	0.92
F1-Score	0.72	0.92
ROCArea	0.73	0.92
Confusion	1616 813	2175 254
Matrix	332 1240	73 1499

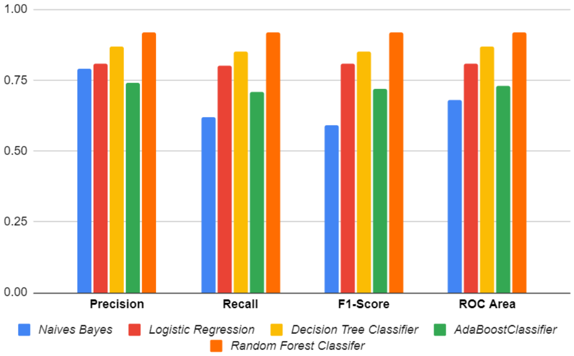

Fig. 1. Precision, Recall, F1-Score, ROCArea

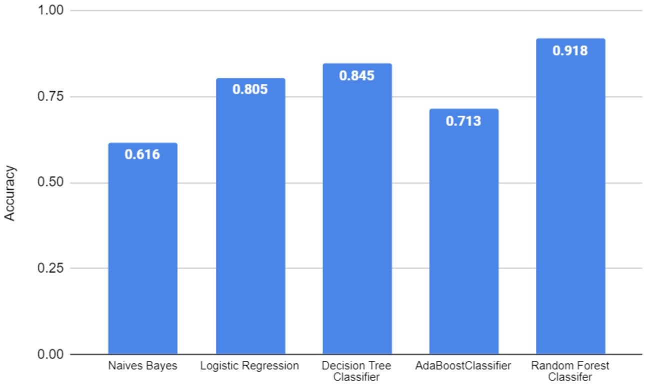

Fig. 2. Accuracy

VI. Conclusion and Future Scope

In this paper, We did a comparative study between various Supervised algorithms, additionally also comparing various Supervised Ensemble methods as well. The overall best performance was shown by Random Forest classifier, giving an accuracy of about 92%. The Ensemble methods performed equal to, or better than the Supervised methods but still, We observed a high True positive rate for the cyberbullying class in all the ensemble methods, which is much more desirable. Naive Bayes performed the worst, giving just 61% accuracy.

Through this paper, we evaluated our approach and compared it with other papers in the section “Related Work”. We also observed that none of the studied past researches used any semi-supervised methods, probably because they are not that popular or effective an didn’t give any commendable result in comparison to the supervised methods.

A very notable fact to be addressed is also the lack of labelled datasets and non-holistic consideration of cyberbullying by researchers when developing detection systems. These are two key challenges facing cyberbullying detection research. Another challenge faced was the lack of resources, due to which we were not able to analyze the performance of SVM(Support Vector Machine) or Multi Layer Perceptron(Neural Networks) classifiers. They have however been mentioned in our study for reference.

Future work on cyberbullying can also benefit by using Dimensionality Reduction as the number of features in this case can be quite high as seen in our example. PCA(Principal Component Analysis) and LDA(Linear Discriminant Analysis) are few common techniques used for this purpose which have the ability to play a really important role in machine learning, especially when working with thousands of features. Principal Components Analysis are one of the top dimensionality reduction algorithms, and in addition to making the work of feature manipulation easier, it can also help to improve the results of the classifier. The idea is to explore advantages and disadvantages of each one and check its results individually and combined as well.

References

[1] DataTurks. (2018, July 12). Tweets Dataset for Detection of Cyber-Trolls. Retrieved November 07, 2020, from https://www.kaggle.com/dataturks/dataset-for-detection-of-cybertrolls?select=Dataset+for+Detection+of+Cyber-Trolls.json

[2] Samghabadi, Niloofar Safi, et al. ”Detecting nastiness in social media.” Proceedings of the First Workshop on Abusive Language Online. 2017.

[3] Yao, Mengfan, Charalampos Chelmis, and Daphney? Stavroula Zois. ”Cyberbullying ends here: Towards robust detection of cyberbullying in social media.” The World Wide Web Conference. 2019.

[4] Huang, Qianjia, Vivek Kumar Singh, and Pradeep Kumar Atrey. ”Cyberbullying detection using social and textual analysis.” Proceedings of the 3rd International Workshop on Socially-Aware Multimedia. 2014.

[5] T. Bin Abdur Rakib, L. K. Soon, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Springer Verlag, 2018), vol. 10751 LNAI, pp. 180–189.

[6] Y. N. Silva, D. L. Hall, C. Rich, BullyBlocker: toward an interdisciplinary approach to identify cyberbullying. Social Network Analysis and Mining. 8 (2018), doi:10.1007/s13278-018-0496-z.

[7] E. Raisi, B. Huang, Weakly supervised cyberbullying detection with participant-vocabulary consistency. Social Network Analysis and Mining. 8 (2018), doi:10.1007/s13278-018-0517-y.

[8] Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, Shivakant Mishra. (2015). Detection of Cyberbullying Incidents on the Instagram Social Network. ”

[9] Dadvar, Maral Eckert, Kai. (2018). Cyberbullying Detection in Social Networks Using Deep Learning Based Models; A Reproducibility Study. 10.13140/RG.2.2.16187.87846.

[10] Nandhini, B. Sri, and J. I. Sheeba. ”Cyberbullying detection and classification using information retrieval algorithm.” Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering Technology (ICARCSET 2015). 2015.

Appendix A
Further Studies on Cyberbullying Detection

Raisi et. al [7] presents a unique model, referred to as participant vocabulary consistency (PVC) Model. This relational model is trained in a weakly supervised manner, as getting high quality labeled data is difficult, in this model, human experts only need to provide high fidelity annotations in the form of key phrases that are highly indicative of harassment. This algorithm then uses these annotations, by searching for patterns of victimization in unlabeled social interaction network- to find other likely key phrases indicators and specific instances of bullying.

Hosseinmardi et. al [8] addresses the cyberbullying incidents on a popular social media platform Instagram by analysing the top comments on the user's public posts. The algorithm used Naive Bayes classification to separate the data and further differentiates it from Cyber Aggression. Cyberbullying was studied in the context of a media based social network, incorporating both images and comments in the labelling. They were also able to show that a Linear SVM classifier can significantly improve the accuracy of identifying cyberbullying to 87% by incorporating multi-modal features from text, images, and meta data for the media session.

Dadvar et. al [9] focuses on using Deep neural networks to detect instances of cyberbullying. Models like Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BLSTM) and BLSTM with attention which vary in complexity in their neural architecture were used on Formspring, Wikipedia and Twitter datasets.

Nandhini et al. [10] Proposes a naive Bayes based learning model and used the dataset of MySpace.com, they achieved a high accuracy of 91%. [8] uses Formspring data, available at Kaggle.com by Kelly Reynolds which initially had about 12000 instances, but after preprocessing, they got a total of 1608 instances where half of them corresponds to cyberbullying, basically, they used this small dataset to train a Neural network and an SVM Classifier.

Appendix B
Visualisation of Outcomes

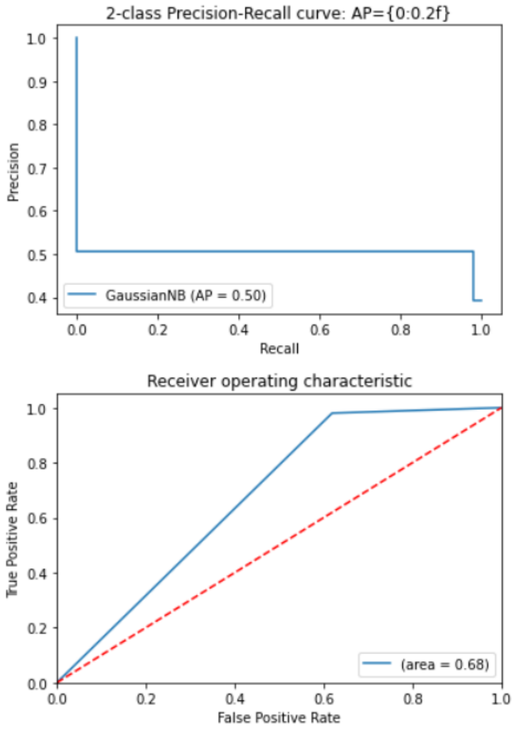

Fig. 3. Naive Bayes Classifier

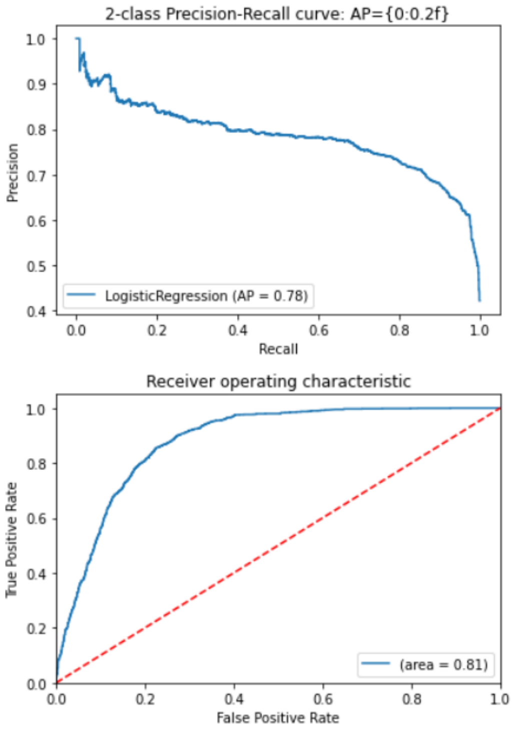

Fig. 4. Logistic Regression Classifier

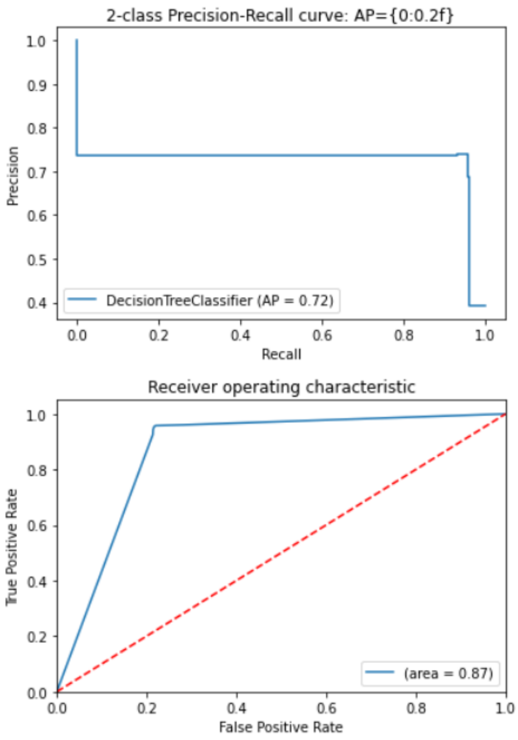

Fig. 5. Decision Tree Classifier

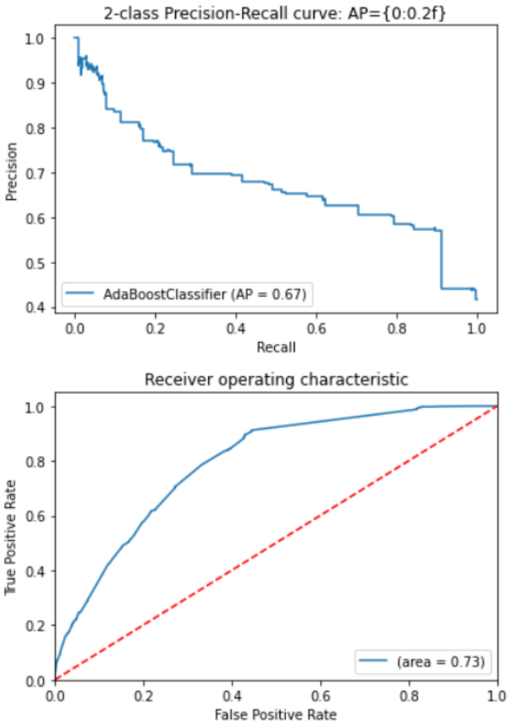

Fig. 6. Adaboost Classifier

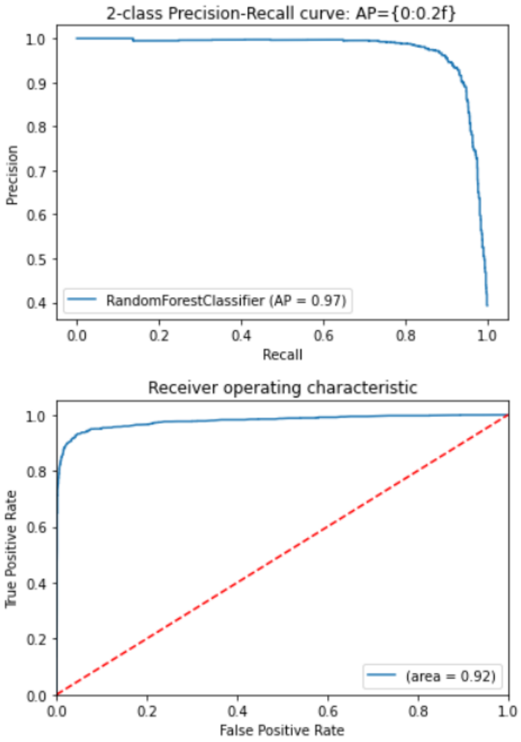

Fig. 7. Random Forest Classifier