DEVELOPING A MACHINE LEARNING BASED SYSTEM FOR CYBERBULLYING DETECTION ON SOCIAL MEDIA PLATFORMS

Chijioke Franklin Emejuru

ABSTRACT

In this paper which encompasses the development of a machine based system for the detection of cyberbullying on social media platforms, two machine learning models and a deep learning model were developed and compared; the logistic regression and decision trees as the machine learning models and the transformer neural network as the deep learning model. The neural network was developed leveraging artificial intelligence which focuses on the utilization of deep learning to capture subtle and complex information in the dataset. In this study, the performance metrics leveraged for the evaluation of the machine learning model performance would be purely statistical measures. The dataset obtained from kaggle is the cyberbullying dataset. The results from the evaluation indicate that the transformer neural network exhibited superior accuracy being the state of the art model with a training accuracy of 99% and validation accuracy of 97%, followed by the logistic regression model and the decision trees with a 79.55% and 73% accuracy respectively.

Keywords: Cyberbullying, transformer neural network, decision trees, logistic regression, kaggle, accuracy.

INTRODUCTION

Social media has seen an exponential increase in its popularity and its users and this exponential growth has been accompanied with some pros and cons. One of the cons of this new growth in social media is the issue of cyberbullying which refers to the use of technology to harass or threaten an individual or a group of individuals. This online harassment has become more prevalent and has led to the damage of many lives hence the need to protect individuals from it. Recent studies have proven that a significant amount of young people have experienced some form of cyberbullying on social media platforms which led to emotional and psychological distress. The introduction of the idea of cyberbullying mitigation is not only a matter of checking negative comments online, it also involves the introduction of a safe space and environment which safeguards the mental health of people and brings about serenity in the digital space. Several stakeholders ranging from parents in local households, teachers, companies to law enforcement agencies play a vital role in the process of creating a safe environment in the digital world. Effective strategies for the mitigation of cyberbullying encompass initiatives that raise awareness, technological tools that detect cyberbullying and prevent it hence the reason in which this paper was written.

The advent of machine learning brought about various approaches for the detection and mitigation of cyberbullying. With the availability of data, machine learning algorithms have the ability to perform analysis across multiple platforms and other digital channels, identifying patterns which point to bullying behavior. This technological approach enables monitoring by training a classifier on previously existing data which helps in future referencing and detection. The integration of these models which can accurately detect cyberbullying into digital platforms is one of the most critical steps in the fight against cyberbullying. There are challenges which accompany the fight against online bullying and harassment such as the anonymity provided by the internet. This anonymity strengthens the bully, which makes them more aggressive because of the fact that they believe there would be complications with their identification. Another challenge is seen in the rapid explosion of human language which has an effect on the ability of the machine learning system to understand what people are saying and this necessitates the constant adaptation of language models to accommodate these languages thus, maintaining its efficacy in the prevention of this problem. Addressing the issue of cyberbullying through machine learning requires a lot of effort which

brings together some sectors such as education, policy and technology. The acknowledgement of the importance of mitigating cyberbullying and working together, society can make significant progress in exploiting machine learning to detect and prevent cyberbullying, which results in the development of a safer online environment for all individuals.

RELATED WORK

In this section, few research papers which have previously focused on the use of machine learning to detect and prevent cyberbullying will be reviewed and analyzed in the coming paragraphs.

The first paper, social media cyberbullying detection using machine learning by Hani, John et al. (2019), the authors proposed a predictive approach for the detection of cyberbullying in social media platforms. The authors utilized a cyberbullying dataset which was obtained from the kaggle repository for machine learning. The dataset was labeled by the authors Kelly Reynolds et al. in their paper. The authors performed the usual text processing steps for machine learning algorithms which are in the order, tokenization, lower casing, removal of stopwords and white spaces and then lastly, word correction. The second step used by the authors is the feature extraction step where the text data is transformed into a suitable format to be used as the model input. The feature extraction step used by the authors was the TF-IDF vectorizer. After the training phase, the authors achieved an accuracy of 91.76% for the neural network and 89.87% for the SVM.

The second paper, Improving cyberbullying detection using twitter users' psychological features and machine learning proposed by Balakrishnan et al., (2020). In this research, the authors thought that empirical evidence which links users' psychology such as personal traits and cybercrimes are numerous. In this research, user's personalities were determined using the big five and dark triad models and then naive bayes, random forest performance of the cyberbullying improved when the user sentiments and personalities were used and not the emotion. They recorded the performance of the random forest and the j48 to be impressive although the j48 was slightly better.

In the third paper, A multilingual system for cyberbullying detection: Arabic content detection using machine learning proposed by Haidar et al., (2017), the authors extended a previously published paper to shed new light on the solution detection and prevention of cyberbullying. In this paper, the researchers outlined several categories of cyberbullying such as flaming, masquerade, denigration, impersonation, harassment, outing, trickery, exclusion and cyberstalking. In the data preprocessing stage, the data was scraped from both Facebook and twitter. Two machine learning models were utilized by the authors for the detection and classification of the cyberbullying dataset, the naive bayes model and the support vector machine. After the model training, the authors found the support vector machine to outperform the naive bayes by some slight difference.

The fourth paper, A comparative analysis of machine learning techniques for cyberbullying detection on twitter, a paper proposed by Amgad Muneer and Suliman Mohamed Fati (2020). In this paper, the authors utilized a global dataset which contained 37,373 tweets and evaluated this dataset on seven classifiers. The authors split the dataset into training and testing sets in the ratio of 70:30. They performed the data preprocessing steps used to clean text data which are removal of punctuation and stop words which was followed by stemming. The feature extraction method utilized in this article was the TF-IDF feature extraction method and word2vec method. The seven classifiers utilized in this paper were logistic regression, light LGBM, stochastic gradient descent, random forest, adaboost, naive bayes and support vector machine. After the training and testing phase,

and j48 models were used for the tweet classification. The authors utilized a dataset which consists of 5453 tweets. The authors also noted that during the execution of the j48 model, the overall
The fifth paper by Desai, Aditya, et al. "Cyber bullying detection on social media using machine learning.", proposed a model based on various features that could be used for classifying cyber bullying threats unlike other traditional methods which cannot utilize these external features. This paper utilized the transformer neural network particularly the BERT which is the bidirectional encoder representations from transformers. The features in which the authors took into consideration are the sentimental, sarcastic, syntactic, semantic and social features. In the result section, the Bert was compared to the naive bayes model and support vector machine and the testing phase of the Bert showed the model to have an accuracy of 91.90%.

In the sixth paper by Alabdulwahab, Aljwharah, Mohd Anul Haq, and Mohammed Alshehri. "Cyberbullying Detection using Machine Learning and Deep Learning." The authors proposed a comparative approach towards the detection of cyberbullying using machine learning and deep neural networks in the aspect of natural language processing. The machine learning models used were k nearest neighbor, support vector machine, naive bayes, decision trees and random forest in contrast to the deep neural networks. The authors utilized a tweet dataset with two classes, cyberbullying and not cyberbullying. The results showed the deep learning model achieved an accuracy of 96%, followed by the support vector machine and the k nearest neighbor with a 92% and 90% accuracy respectively.

The seventh paper Keni, A., and M. Kini. "Cyber-bullying detection using machine learning algorithms.", proposes the use of supervised machine learning algorithms. The authors utilized a dataset from the kaggle online repository for machine learning which contained a high amount of bullying content, which was then preprocessed using the text preprocessing methods such as lowercase conversion, removal of special characters, stop words removal and stemming of words to their root form. The authors confirmed the the logistic regression model achieved the best accuracy of 90.57%.

METHODOLOGY

In this section, a detailed overview from the data collection to the model selection for training will be discussed. The choice of machine learning model for the task of the detection of cyberbullying are the transformer neural networks, the logistic regression and the decision trees classifiers.
The choice for the models are based on some assumptions such as the transformer neural network is a state of the art model for natural language processing and thus, is expected to perform better, the logistic regression model is built for binary classification because of its original sigmoid output and the decision trees, being a tree based algorithm would have a higher depth in understanding the constraints in the data.
The dataset utilized in this research was obtained from kaggle which is an online database where several datasets are available for different machine learning tasks.
The first step taken was to read the dataset using the pandas library for data frame manipulation where the index and date columns were dropped as they were deemed as redundant. As this research is text based, it was mandatory to first initialize stop words and the stemmer which are regularly used words and the library for reducing texts to their root form respectively. A function was then created which housed all the text preprocessing methods. The methods employed in this research for the preprocessing are:

- Conversion of text from uppercase to lowercase
- Removal of punctuation marks
- Removal of special characters
- Word tokenization
- Removal of stop words
- Stemming

The dataset is a binary classification dataset where 1 represents cyberbullying and 0 represents non-cyberbullying.
Before the machine learning models were employed, the text column of the dataset was turned into vectors using the TF-IDF vectorizer after which the data was

support vector machine to outperform all the other models.

Upon testing, the logistic regression performed better than the decision trees classifier with an accuracy of 80% whereas the latter had a 73% accuracy.
The transformer neural network was implemented using the PyTorch framework for deep learning which uses object oriented based programming in Python. Two classes were written for the dataset and the BERT model initialization. As it's a deep learning approach, it took the regular convention of splitting the dataset into train, Val and test set which was passed into a data loader before taking the right shape as input for the model. Upon training and testing, the transformer neural network performed well with a train accuracy of 99.81%, a validation accuracy of 97.42% and a test accuracy of 80%. The transformer is a data hungry model so requires heavy data for training, it is prone to overfitting.

EVALUATION METRICS

In order to remove bias and minimize variation, machine learning models are assessed using mathematically equivalent criteria, creating a uniform foundation for operational enhancement. In this research, the performance of the three models used were evaluated using the following metrics:

- Accuracy
- Precision
- Recall
- F1 score

1. Accuracy: Since accuracy is essentially the degree to which a model properly identifies an instance, it might be considered the most widely used performance metric. It is used in the evaluation of a classification model in machine learning. It is gotten by taking the ratio of the number of correct predictions to the total number of predictions. The choice of accuracy was due to the fact that it is easy to understand and calculate and it is also

split into train and test set. The machine learning algorithms, the logistic regression and the decision trees classifiers were imported using the sci-kit learn library for machine learning. The models were fit with the X and y train and tested with the X test.

   predicts the positive outcomes. Another name for it is the "true positives" ratio, which is calculated by dividing the total number of correctly detected positive events by the sum of the true and false positives. It is important to use precision because it determines how many of the predicted positive outcomes are relevant and when the precision is high, it means that the model is making fewer false positive errors and is important especially in cases where false positives are costly.

3. Recall: Another name for this is true positive rate or sensitivity. It concentrates on the positive examples/classes of a dataset since it assesses a model's capacity to accurately identify all pertinent occurrences in a dataset. It is essential to us in this research as we want to capture as many relevant instances as possible in the cyberbullying dataset.

4. F1 score: In order to evaluate the effectiveness of a classification machine learning model, the precision and recall are combined to create the F1 score, which is also frequently used in the fields of statistics and machine learning. It is described as the precision and recall metrics' harmonic means. When working with unbalanced datasets where one class ratio is greater than the other, this statistic is especially helpful.

CONCLUSION AND FUTURE RECOMMENDATIONS

In this research, a comparative analysis between two machine learning models alongside the transformer neural network which is a deep learning model was conducted on the cyberbullying dataset which was obtained from kaggle. After the completion of the study, the transformer neural network was found to be better than the other two models used in the

applicable to both binary and multi class classification tasks.

2. Precision: This is another essential performance statistic for classification model analysis that measures how well the model

comparative study in terms of its accuracy. Prior to this experiment, the bias was on the transformer neural network as the one which would have the highest accuracy in the analysis because of the performance of the model on a classification dataset in 2017 when the dawn of the transformers came upon us in the attention paper. From this research, it is clear that future models should rely more on the attention model for its accuracy and generalizability supporting the initial hypothesis and bias of the transformers being the most suitable model for the task. In this research, a few limitations were identified such as compute power, the transformer neural network requires a large amount of data and also GPU compute resources. In the future, this research could bridge gaps by integrating a web scraper which automatically scrapes and generates data for the transformers model, easing the process of data collection by e-commerce companies and retail businesses to improve a safer environment for online interaction. Subsequently, the next models will be taken into account in order to regularly assess and modify the three models to ensure optimal performance. This includes feature engineering, hyperparameter tuning, and testing new algorithms or techniques that may enhance the model's accuracy and efficiency. Collect and update new data often to keep the models up to date. Verify that the data is correct, relevant, and representative of the relevant field. A larger data set may aid in improving the model's generality and robustness. Provide a dependable mechanism for monitoring the three models' performance in real time. Finally, regularly verify the models for injustice and bias. Examine their outputs and performance parameters to identify any degradation over time and undertake maintenance if needed. Provide policies to prevent or mitigate harm to individuals or communities, and ensure that all legal requirements and ethical principles are fulfilled.

REFERENCES

- Hani, J., Nashaat, M., Ahmed, M., Emad, Z., Amer, E., & Mohammed, A. (2019). Social and machine learning. Computers & Security, 90, 101710.
- Haidar, B., Chamoun, M., & Serhrouchni, A. (2017). A multilingual system for cyberbullying detection: Arabic content detection using machine learning. Advances in Science, Technology and Engineering Systems Journal, 2(6), 275-284.
- Muneer A, Fati SM. A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter. Future Internet. 2020; 12(11):187.
- Desai, A., Kalaskar, S., Kumbhar, O., & Dhumal, R. (2021). Cyber bullying detection on social media using machine learning. In ITM Web of Conferences (Vol. 40, p. 03038). EDP Sciences.
- Alabdulwahab, A., Haq, M. A., & Alshehri, M. (2023). Cyberbullying Detection using Machine Learning and Deep Learning. International Journal of Advanced Computer Science and Applications, 14(10).
- Keni, A., & Kini, M. (2020). Cyber-bullying detection using machine learning algorithms. Computer Science, Psychology.

media cyberbullying detection using machine learning. International Journal of Advanced Computer Science and Applications, 10(5), 703-707.

- Balakrishnan, V., Khan, S., & Arabnia, H. R. (2020). Improving cyberbullying detection using Twitter users' psychological features

APPENDIX 1

Reading the dataset

READING THE DATA

```
In [2]: data = pd.read_csv("kaggle_parsed_dataset.csv") # Reading data from 'kaggle_parsed_dataset' into data DataFrame

In [3]: # Displaying the DataFrame
        data.head()
```

Out[3]:

	index	oh_label	Date	Text
0	0	1	20120618192155Z	"You fuck your dad."
1	1	0	20120528192215Z	"i really don't understand your point.\xa0 It ...
2	2	0	NaN	"A\\xc2\\xa0majority of Canadians can and has ...
3	3	0	NaN	"listen if you dont wanna get married to a man...
4	4	0	20120619094753Z	"C\xe1c b\u1ea1n xu\u1ed1ng \u0111\u01b0\u1edd...

Text preprocessing

CREATING A FUNCTION TO PREPROCESS THE WHOLE DATASET IN ONE CALL

```
In [9]: def preprocess_text(text):
            # convert to lowercase
            text = text.lower()

            # remove punctuations
            text = text.translate(str.maketrans("", "", string.punctuation))

            # remove special characters
            text = re.sub(r"\s+[^a-zA-Z]\s+", " ", text)

            # tokenize text
            tokens = word_tokenize(text)

            # remove stop words
            tokens = [word for word in tokens if word not in stop_words]

            # stemming
            tokens = [stemmer.stem(word) for word in tokens]

            # join tokens back to string
            text = " ".join(tokens)

            return text
```

Importing machine learning libraries

In [1]:
```
# these are the libraries used for text preprocessing and machine learning modeling
import numpy as np # Importing numpy library and aliasing it as np
import nltk
from nltk.stem import WordNetLemmatizer, PorterStemmer
from nltk import word_tokenize
from nltk.corpus import stopwords
nltk.download("wordnet")
nltk.download("stopwords")
nltk.download("punkt")
import re
from tqdm.auto import tqdm
import string
import pandas as pd # Importing pandas library and aliasing it as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_curve, roc_auc_score
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt # Importing pyplot module from matplotlib library and aliasing it as plt
import seaborn as sns # Importing seaborn library and aliasing it as sns
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns', None)   # Setting pandas option to display all columns in DataFrame
plt.style.use('ggplot')  # Setting plot style to 'ggplot' from matplotlib
```

Logistic Regression

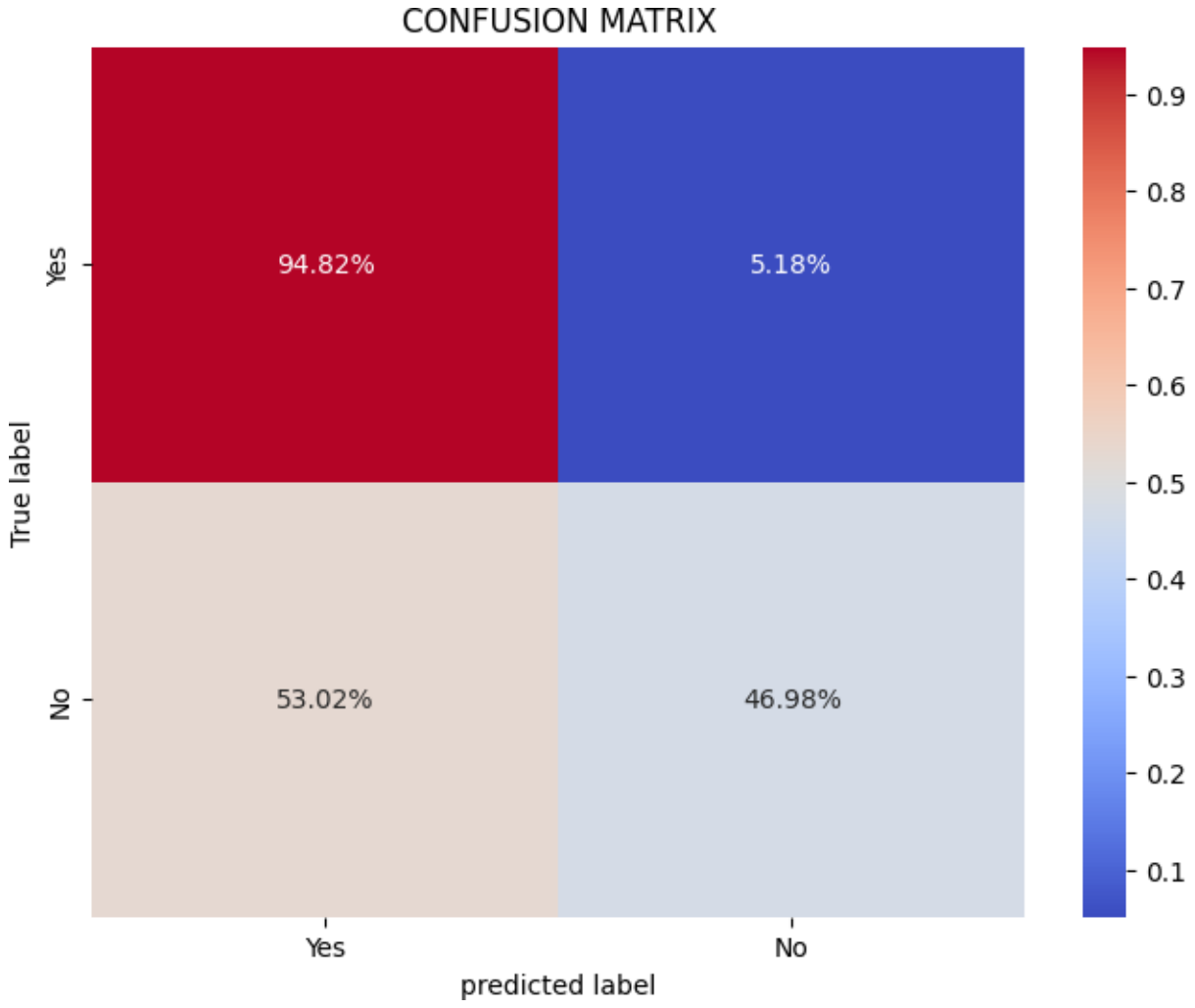
THE LOGISTIC REGRESSION

In [20]:
```
# train the naive model
model = LogisticRegression(max_iter=10000)
```

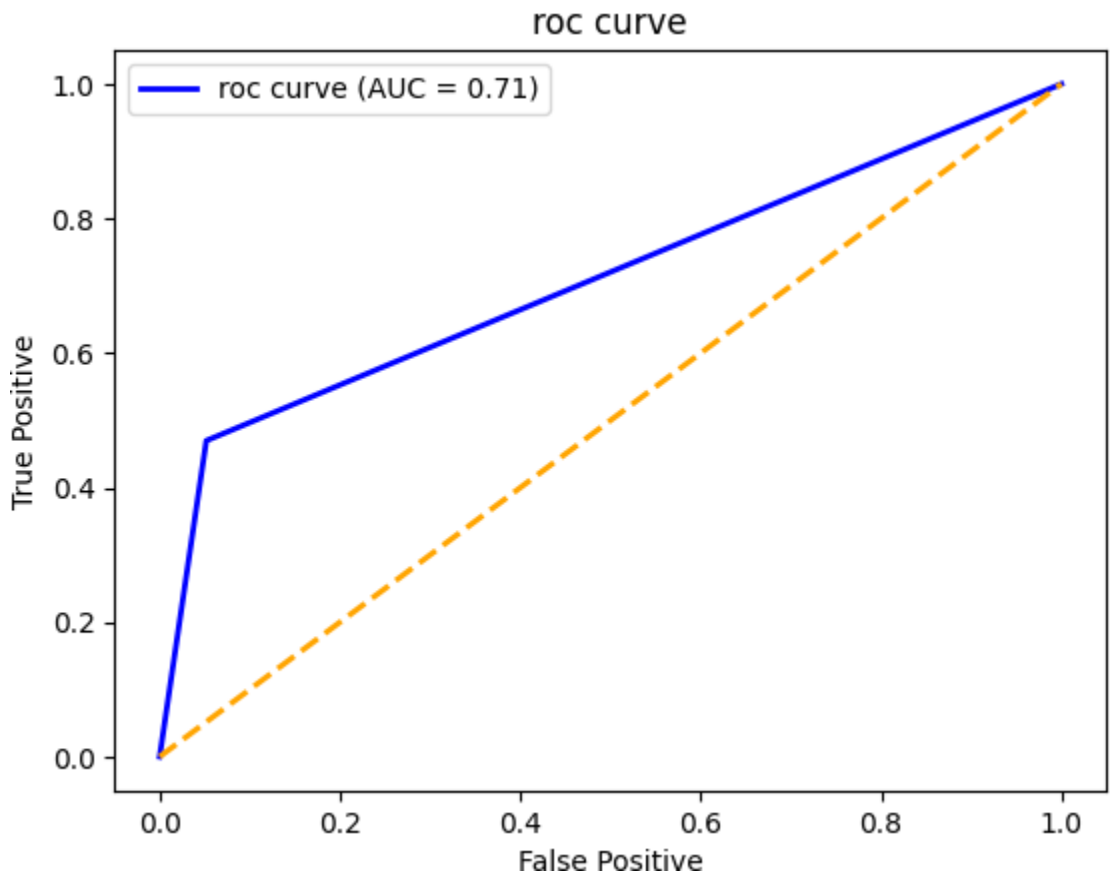
In [21]:
```
model.fit(X_train, y_train)
```

Out[21]:
```
LogisticRegression(max_iter=10000)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Logistic Regression Confusion Matrix



ROC curve for the Logistic regression



Decision Trees Classifier

DECISION TREES CLASSIFIER

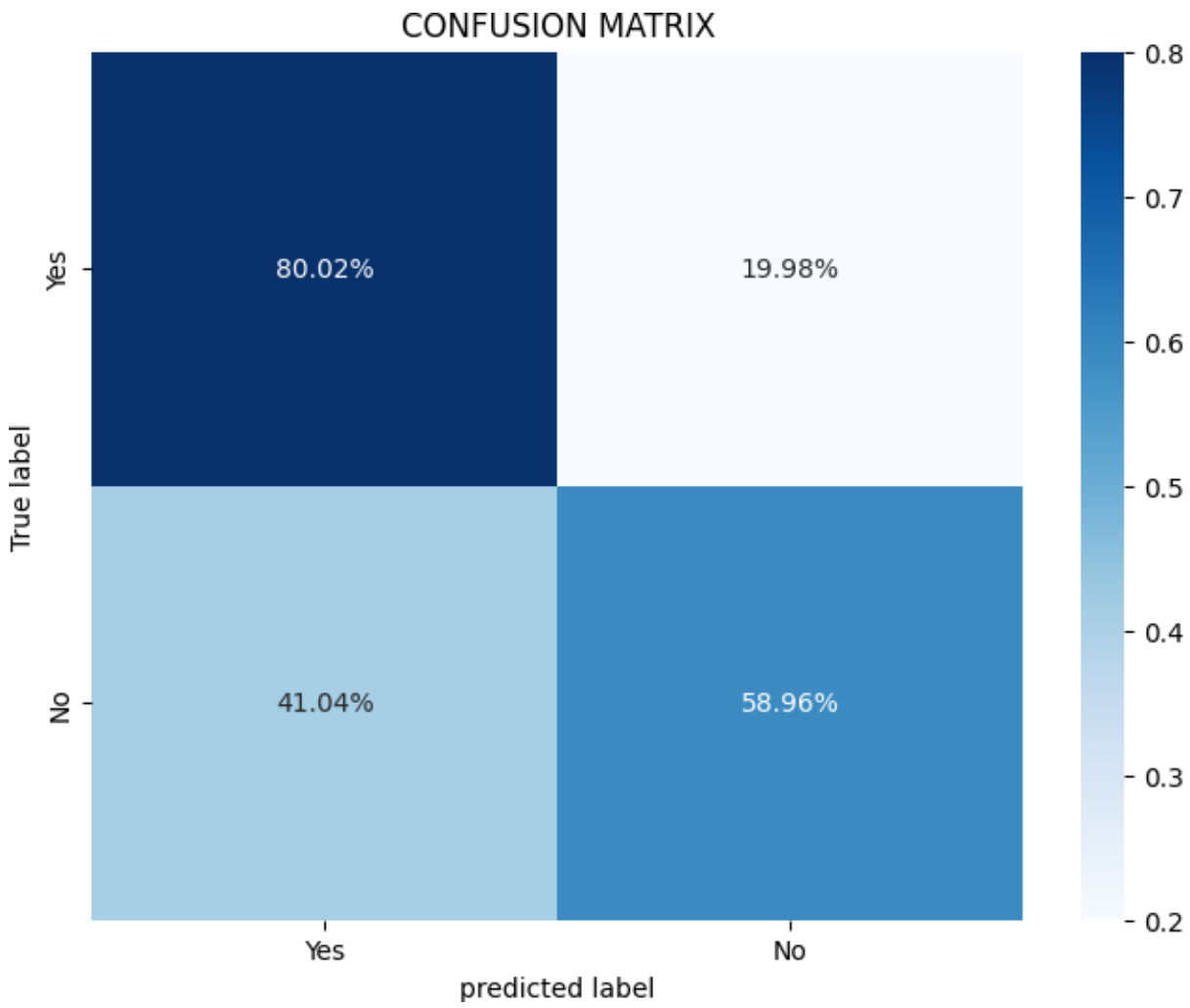
In [33]:
```
# train the naive model
model = DecisionTreeClassifier()
```

In [34]:
```
model.fit(X_train, y_train)
```

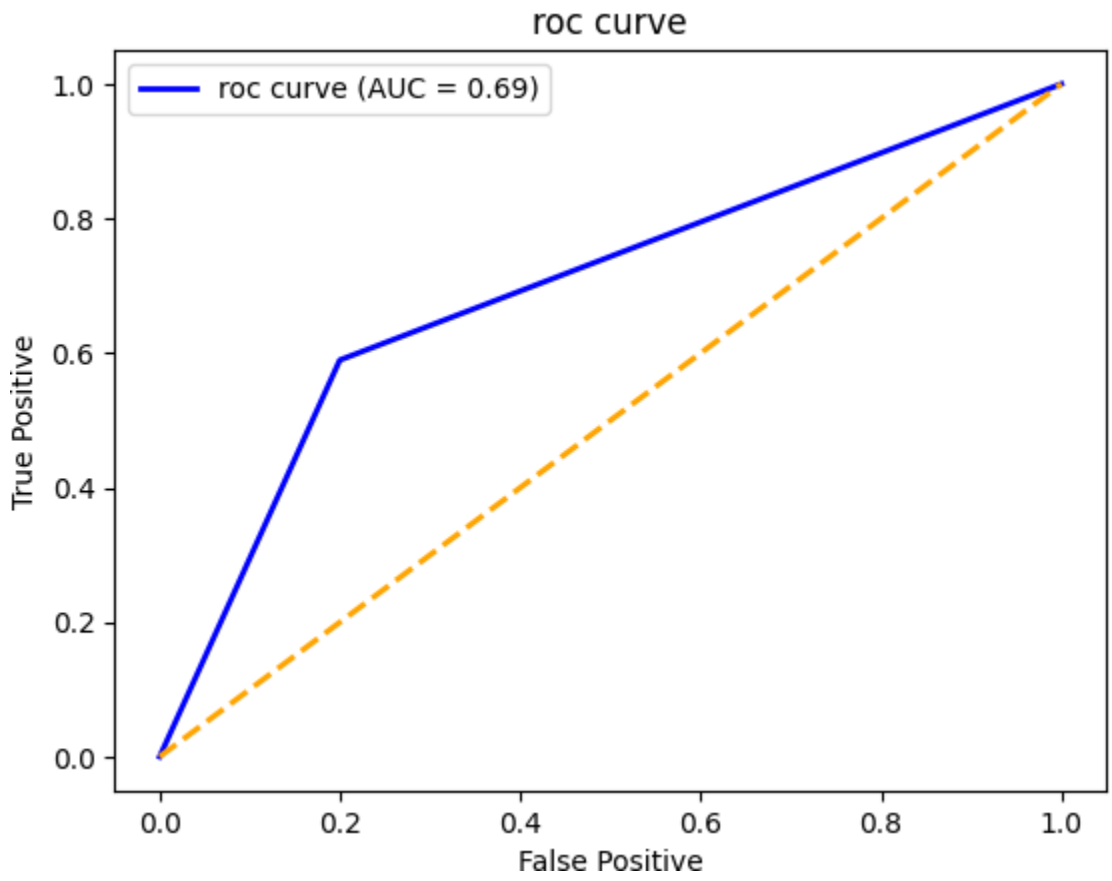
Out[34]:
```
DecisionTreeClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Decision Trees Confusion Matrix


CONFUSION MATRIX
80.02%
19.98%
41.04%
58.96%
Yes
No
True label
Yes
No
predicted label
0.8
0.7
0.6
0.5
0.4
0.3
0.2

ROC curve for the decision trees



Transformer Neural Network

THE TRANSFORMER NEURAL NETWORK

In [41]:

```
# this is the pytorch library for the transformer neural network
from transformers import BertModel, BertTokenizer, get_linear_schedule_with_warmup
from transformers import DataCollatorWithPadding
from torch.utils.data import Dataset, DataLoader
import torch
from torch import nn, optim
from torch.utils.data import Dataset, DataLoader
from torchvision import transforms
import torch.nn.functional as F
import numpy as np
from tqdm.auto import tqdm
```

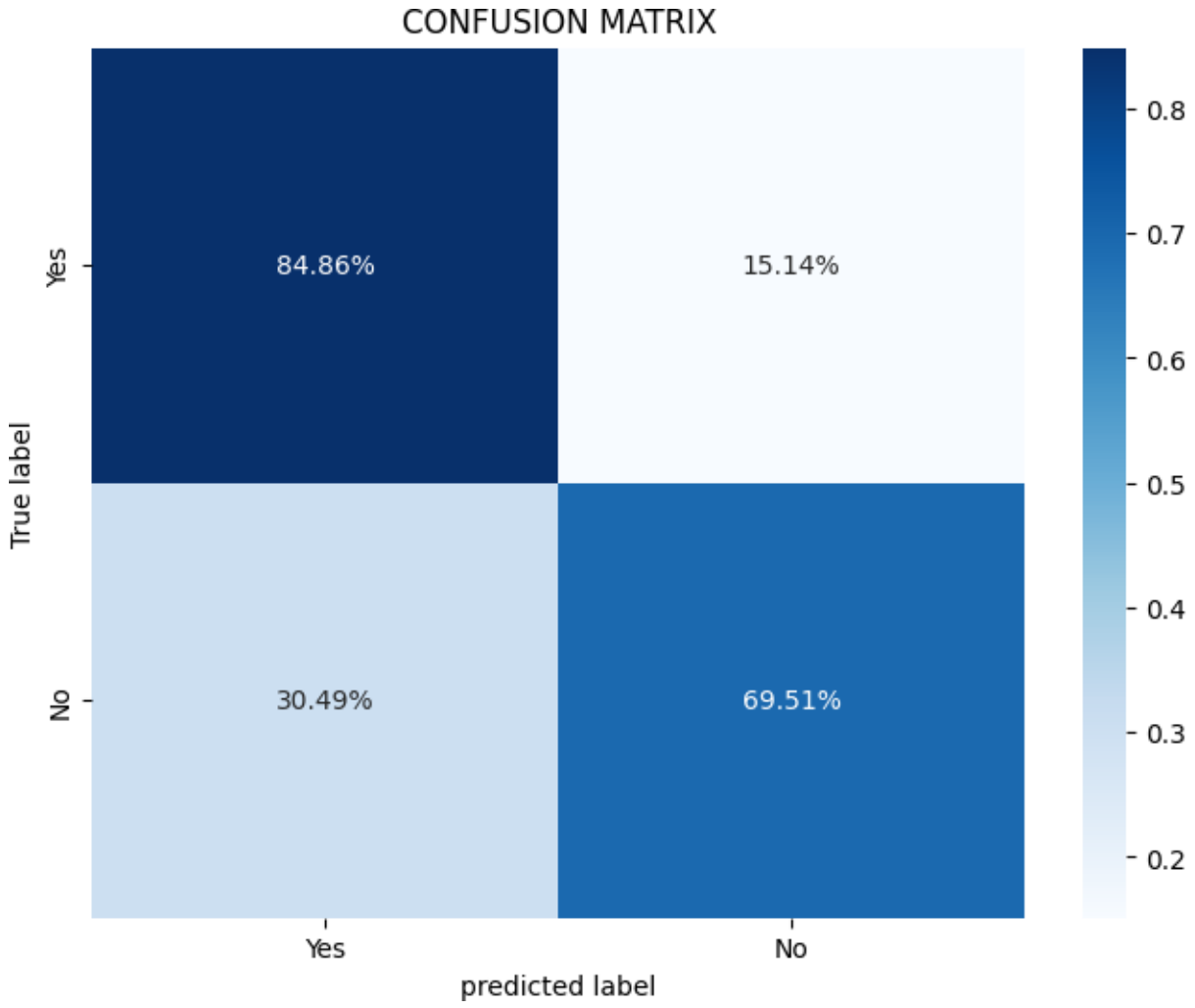
The model

ADDING A CLASSIFIER HEAD TO THE PRE-TRAINED BERT MODEL

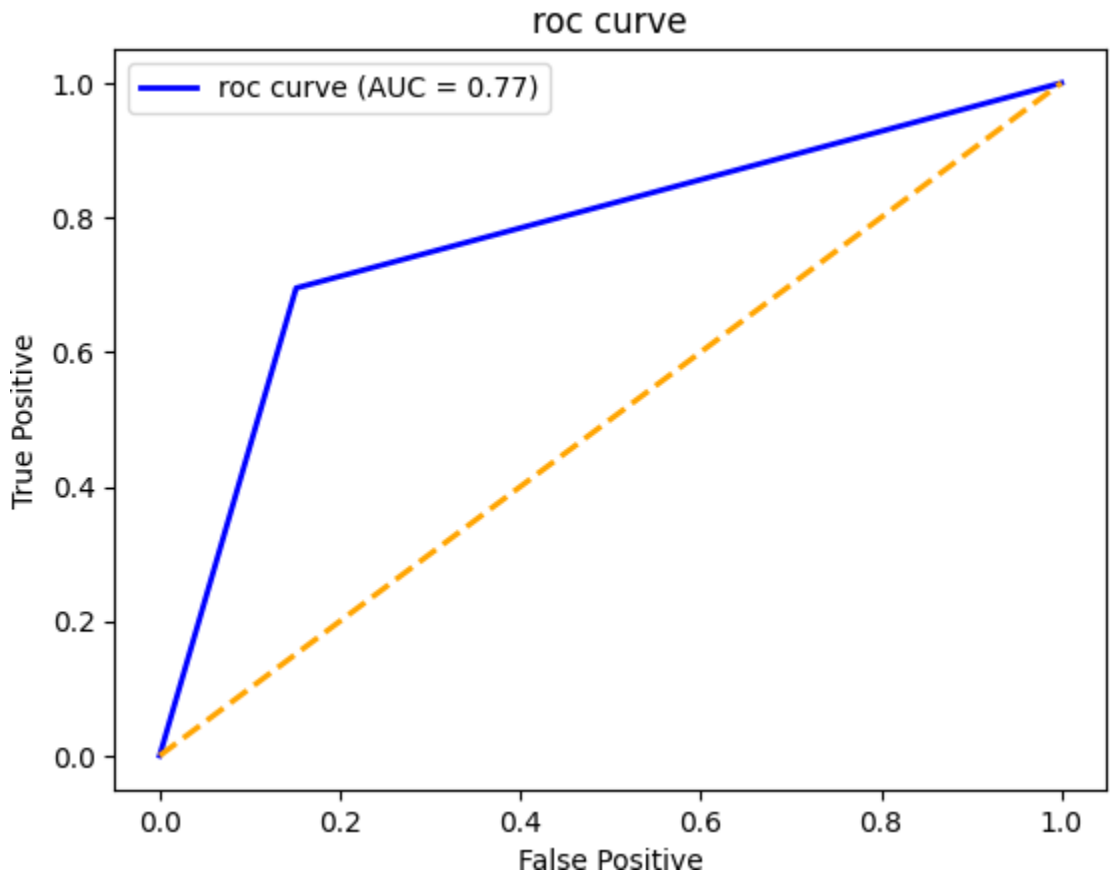
In [51]:

```
class BERTClassifier(nn.Module):
    """This class calls the BERT weight from the pytorch library for pretrained models, fine tunes it to suit
    the number of classes for our desired task then outputs the tuned model"""
    def __init__(self, num_classes):
        super(BERTClassifier, self).__init__()
        #Load the pre-trained BERT model
        self.bert = BertModel.from_pretrained("bert-base-uncased", return_dict = False, from_tf = False) #body
        self.drop = nn.Dropout(0.3)
        #Ths will replace the final fully connected layer with a new one
        self.out = nn.Linear(self.bert.config.hidden_size, num_classes)

    def forward(self, input_ids, attention_mask, token_type_ids):
        _, pooled_output = self.bert(
        input_ids = input_ids,
        attention_mask = attention_mask,
        token_type_ids = token_type_ids
        )

        output = self.drop(pooled_output)
        output = self.out(output)
        return output
```

Transformer Confusion Matrix



Transformer ROC curve



DataLoader

CREATING A DATALOADER

In [49]:

```
def Create_data_loader(data, tokenizer, max_len = MAX_LEN, batch_size = BATCH_SIZE, include_raw_text = False):
    dataset = CyberbullyingDataset(
        df = data,
        reviews = data["Text"].to_list(),
        targets = data["oh_label"].to_list(),
        tokenizer= tokenizer,
        max_len = max_len,
        include_raw_text=include_raw_text
    )
    return DataLoader(dataset, batch_size=batch_size, collate_fn=collator)
```