Toxic Comment Classification for Social Media Using IBM Services

1 INTRODUCTON

1.1 Overview

Nowadays, the flow of data over the internet has grown dramatically, especially with the appearance of social networking sites. Social networks sometimes become a place for threats, insults, and other components of cyberbullying. A huge number of people are involved in online social networks.

Toxic comments are textual comments with threats, insults, obscene, racism, etc. In recent years there have been many cases in which authorities have arrested some users of social sites because of the negative (abusive) content of their personal pages. Hence, the protection of network users from anti-social behavior is an important activity. One of the major tasks of such activity is automated detecting the toxic comments.

Bag of words statistics and bag of symbols statistics are the typical source information for the toxic comments detection. Usually, the following statistics-based features are used: length of the comment, number of capital letters, number of exclamation marks, number of question marks, number of spelling errors, number of tokens with non-alphabet symbols, number of abusive, aggressive, and threatening words in the comment, etc. A neural network model is used to classify the comments.

1.2 Purpose

To build a multi-headed model that's capable of detecting different types of toxicity like threats, obscenity, insults, and identity-based hate.The goal is to create a classifier model that can predict if input text is inappropriate (toxic).

1. Explore the dataset to get a better picture of how the labels are distributed, how they correlate with each other, and what defines toxic or clean comments.
2. Create a baseline score with a simple logistic regression classifier.
3. Explore the effectiveness of multiple machine learning approaches and select the best for this problem.
4. Select the best model and tune the parameters to maximize performance.
5. Build a the final model with the best performing algorithm and parameters and test it on a holdout subset of the data.

2 LITERATURE SURVEY

2.1 Existing Problem

Sentiment classification regarding toxicity has been intensively researched in the past few

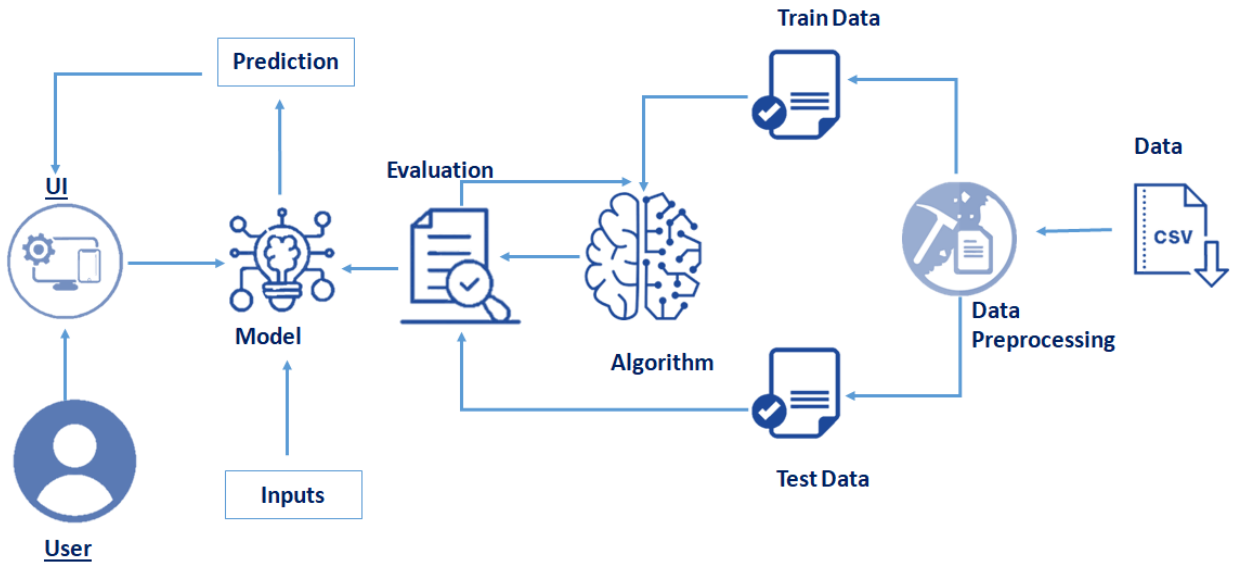
years,largely in the context of social media data where researchers have applied various machine learning systems to try and tackle the problem of toxicity as well as the related,more well known,task of sentiment analysis.Comment abuse classification research initially began with yin et al's application of combining TF-IDF with sentiment/contextual features.They compared the performance of this model with a simple TF-IDF model and reported a 6% increase in F1 score of the classifier on chat style datasets.

2.2 Proposed solution

The following statistics-based features are used: length of the comment, number of capital letters, number of exclamation marks, number of question marks, number of spelling errors, number of tokens with non-alphabet symbols, number of abusive, aggressive, and threatening words in the comment, etc. A Machine Learning model is used to classify the comments.

3 THEORETICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

Hardware : Windows 10

Software :
1. Anaconda Navigator

2. Jupyter notebook
3.Spyder notebook

To build Machine learning models you must require the following packages

Sklearn: Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms.

NumPy: NumPy is a Python package that stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object

Pandas: pandas is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

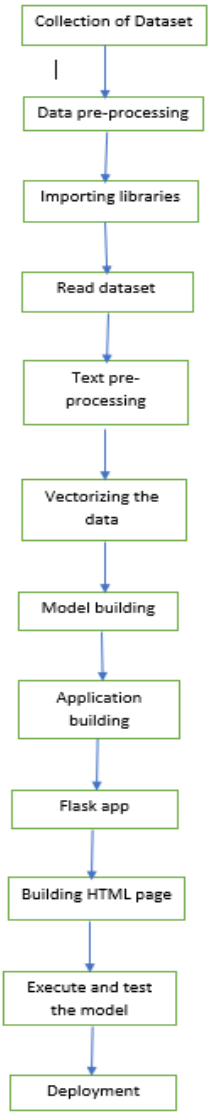
Matplotlib: It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits

Flask: Web framework used for building Web applications.

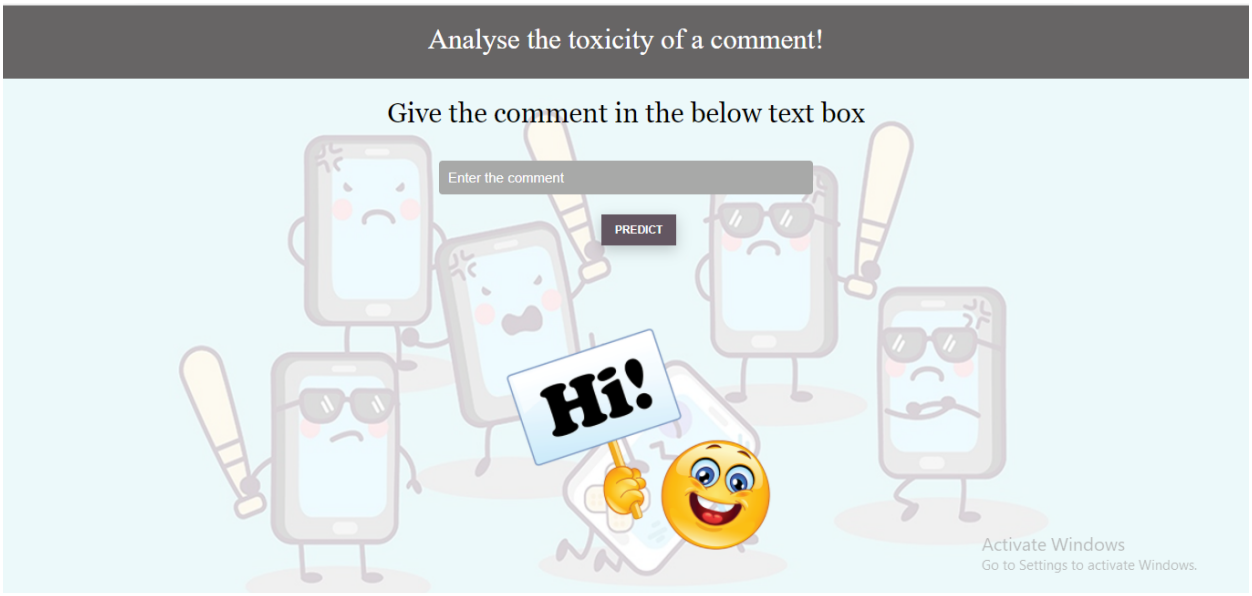
4 EXPERIMENTAL INVESTIGATIONS

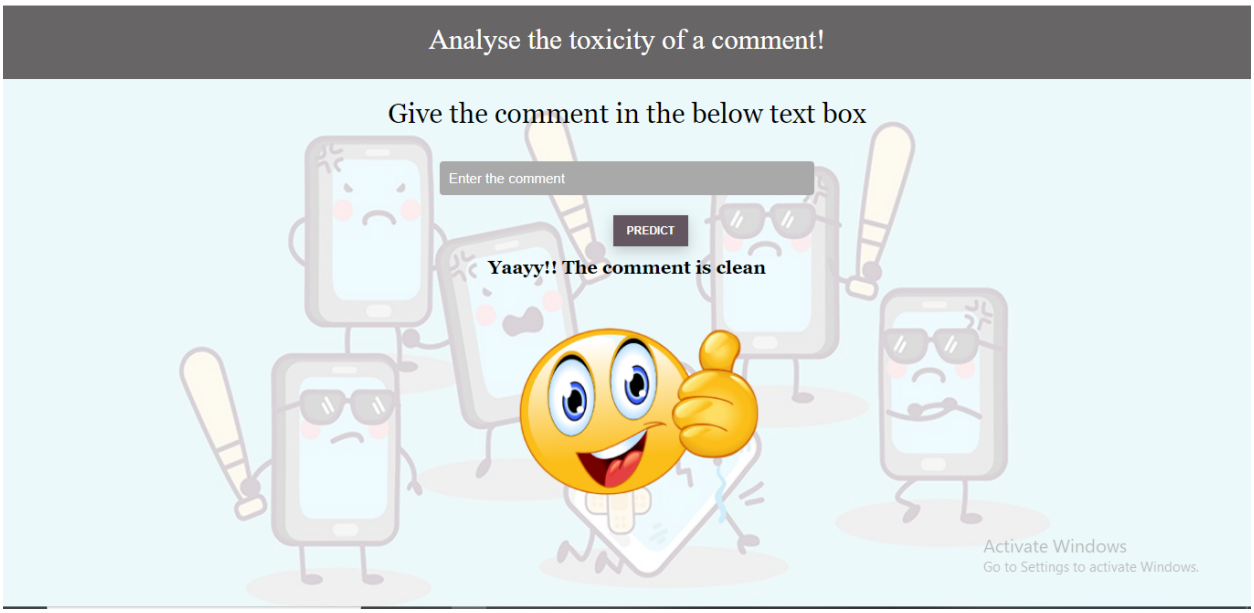
Here we are going to build a machine learning model that predicts toxicity of a comment.For this purpose we searched for the comments that falls under the category of insult,toxic,severe_toxic,identity_hate,threat,obscene.Also tried to understand the concept of Natural Language Processing and artificial neural networks.

5 FLOWCHART


Collection of Dataset
Data pre-processing
Importing libraries
Read dataset
Text pre-processing
Vectorizing the data
Model building
Application building
Flask app
Building HTML page
Execute and test the model
Deployment

6 RESULT


Analyse the toxicity of a comment!
Give the comment in the below text box
Enter the comment
PREDICT
Hi!
Activate Windows
Go to Settings to activate Windows.


Analyse the toxicity of a comment!
Give the comment in the below text box
Enter the comment
PREDICT
Yaayy!! The comment is clean
Activate Windows
Go to Settings to activate Windows.

Analyse the toxicity of a comment!

Give the comment in the below text box

Enter the comment

PREDICT

The comment is toxic

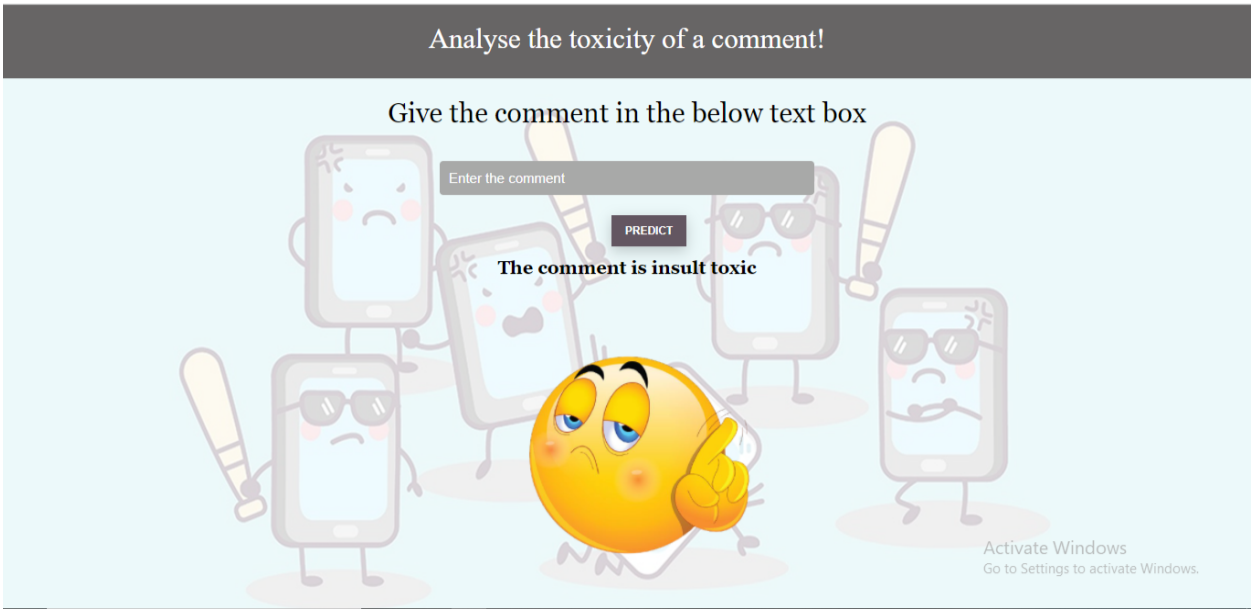
Analyse the toxicity of a comment!

Give the comment in the below text box

Enter the comment

PREDICT

The comment is obscene toxic



7 ADVANTAGES & DISADVANTAGES

Advantages:

1.To protect users from being exposed to offensive language on online forums or social media sites.

2.Protect internet users from becoming victims of online harassment and cyberbullying.

3.The protection of network users from anti-social behavior.

Disadvantages:

1. Out-of-vocabulary words: A common problem for the task is the occurrence of words that are not present in the training data. These words include slang or misspellings, but also intentionally obfuscated content.

2. **Long-Range Dependencies**: The toxicity of a comment often depends on expressions made in early parts of the comment. This is especially problematic for longer comments (>50 words) where the influence of earlier parts on

the result can vanish.

3.**Multi-word phrases:** We see many occurrences of multi-word phrases in both datasets. Our algorithms can detect their toxicity only if they can recognize multiple words as a single (typical) hateful phrase.

8 APPLICATIONS

We use this in social media networks to find the toxicity of a comment.In the application, the user provides any text(comment) to check and analyzed by calling the different label model files generated during cleaning. Finally, the toxicity of the comment is displayed on the UI.

9 CONCLUSION

Communication is one of the basic necessities of everyone's life. People need to talk and interact with one another to express what they think. Over the years,social media and social networking have been increasing exponentially due to an upsurge (rise) in the use of the internet. Flood of information arises from online conversation on a daily basis, as people are able to discuss, express themselves and express their opinion via these platforms While this situation is highly productive and could contribute significantly to the quality of human life, it could also be destructive and enormously dangerous. The responsibility lies on the social media administration,or the host organization to control and monitor these comments

This research work focuses on developing a model that would automatically classify a comment as either toxic or non-toxic using logistic regression.Therefore, this study aim to develop a multi-headed model to detect different types of toxicity like threat,obscenity, insults, and identity hate.

10 FUTURE SCOPE

By making use of Convolutional neural network(CNN) we can classify stickers,gif files into clean,toxic,threat,obscene etc.

11 BIBILOGRAPHY

https://www.researchgate.net/publication/331857893_A_Machine_Learning_Approach_to_Comment_Toxicity_Classification

https://aclanthology.org/W18-5105.pdf

https://jayspeidell.github.io/portfolio/project05-toxic-comments/

APPENDIX

A. Source Code

```
import numpy as np
import pandas as pd
import re
from sklearn.feature_extraction.text import CountVectorizer
import pickle
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from flask import Flask, request, jsonify, render_template, url_for

loaded=CountVectorizer(decode_error='replace',vocabulary=pickle.load(open('word_features.pkl'
,'rb')))

app = Flask(__name__)

def clean_text(text):
    text = text.lower()
    text = re.sub(r"what's", "what is ", text)
    text = re.sub(r"\'s", " ", text)
    text = re.sub(r"\'ve", " have ", text)
    text = re.sub(r"can't", "cannot ", text)
    text = re.sub(r"n't", " not ", text)
    text = re.sub(r"i'm", "i am ", text)
    text = re.sub(r"\'re", " are ", text)
    text = re.sub(r"\'d", " would ", text)
    text = re.sub(r"\'ll", " will ", text)
    text = re.sub(r"\'scuse", " excuse ", text)
    text = re.sub('\W', ' ', text)
    text = re.sub('\s+', ' ', text)
    text = text.strip(' ')
    return text

@app.route('/')
```

```
def landingpage():
    img_url = url_for('static',filename = 'images/hello.png')
    print(img_url)
    flag=0
    return render_template('toxic.html',flag=flag)

@app.route('/predict')
@app.route('/', methods = ['GET','POST'])
def predict():
    if request.method == 'GET':
        img_url = url_for('static',filename = 'images/hello.png')
        return render_template('toxic.html',url=img_url)
    if request.method == 'POST':
        comment = request.form['comment']
        new_row = {'comment_text':comment}
        user_df = pd.DataFrame(columns = ['comment_text'])
        user_df = user_df.append(new_row,ignore_index = True)
        user_df['comment_text'] = user_df['comment_text'].map(lambda com : clean_text(com))
        user_text = user_df['comment_text']
        user_features = loaded.transform(user_text)
        cols_target = ['obscene','insult','toxic','severe_toxic','identity_hate','threat']
        lst= []
        mapper = {}
        for label in cols_target:
            filename = str(label+'_model.sav')
            filename
            model = pickle.load(open(filename, 'rb'))
            print('... Processing {}'.format(label))
            user_y_prob = model.predict_proba(user_features)[:,1]
            print(label,":",user_y_prob[0])
            lst.append([label,user_y_prob])
        print(lst)
```

```
        final=[]
        flag=0
        for i in lst:
            if i[1]>0.5:
                final.append(i[0])
                flag=2
        if not len(final):
            text = "Yaayy!! The comment is clean"
            img_url = url_for('static',filename = 'images/happy.png')
            flag=1
            print(img_url)
        else:
            text="The comment is "
            for i in final:
                text = text+i+" "
            img_url = url_for('static',filename = 'images/toxic.png')
        print(text)
        return render_template('toxic.html',ypred = text,url= img_url,flag=flag)

if __name__ == '__main__':
    app.run(host = 'localhost', debug = True , threaded = False)
```