ACKNOWLEDGEMENT

A very sincere and honest acknowledgement to Mohd. ZeeshanAlam, Assistant Professor, Department of Computer Engineering, JamiaMilliaIslamia, New Delhi for his invaluable technical guidance and support, great innovative ideas and overwhelming moral support. We are also very grateful to our HOD, Prof. Tanveer Ahmad for his valuable support throughout the project. We also express our gratitude to the Department of Computer Engineering and entire faculty members, for their teaching, guidance and encouragement.

We are also thankful to our classmates and friends for their valuable suggestions and support whenever required.

We regret any inadvertent omissions.

Adeel Masood Palak Khandelwal Shazia Khan

(15BCS0018) (15BCS0037) (15BCS0074)

Department of Computer Engineering

Faculty of Engineering and Technology

Jamia Millia Islamia

New Delhi

**ONLINE CENSOR FOR OFFENSIVE LANGUAGE  
(Offensive Language Identifier)**

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER ENGINEERING**



UNDER THE SUPERVISION OF: SUBMITTED BY:

Mohd. ZeeshanAlam, Adeel Masood (15BCS0018)

(Assistant Professor) Palak Khandelwal (15BCS0037)

Shazia Khan (15BCS0074)

**DEPARTMENT OF COMPUTER ENGINEERING**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**JAMIA MILLIA ISLAMIA**

**NEW DELHI**

**(YEAR-2018)**

CERTIFICATE

This is to certify that the project entitled “Online Censor for Offensive Language (Offensive Language Identifier)” by Adeel Masood (15BCS0018), Palak Khandelwal (15BCS0037) and Shazia Khan (15BCS0074) is a record of bonafide work carried out by them, in The Department of Computer Engineering, JamiaMilliaIslamia, New Delhi, under my supervision and guidance in partial fulfillment of requirement for the award of Bachelor of Engineering in Computer Engineering, JamiaMilliaIslamia in academic year 2018.

**Head**   **Mohd. ZeeshanAlam**

(Assistant Professor)

Department of Computer Engineering Department of Computer Engineering

Faculty of Engineering and Technology Faculty of Engineering and Technology

Jamia Millia Islamia Jamia Millia Islamia

New Delhi New Delhi

ABSTRACT

Since the textual contents on online social media are highly unstructured, informal, and often misspelled, existing research on message level offensive language detection cannot accurately detect offensive contents.

Social Networking Sites are booming as never before. Apart from the numerous new opportunities that are provided, also hazards such as messages containing offensive, abusive or sarcastic attacks have to be taken into account. Since manually monitoring and analyzing all messages separately is unattainable, solutions using automated methods are sought.Offensive language can be defined as “expressing extreme subjectivity” and this project mainly focuses on two categories ‘offensive’ and ‘abusive’. This method appliespython modules and custom methods to perform automated offensive language detection.

A corpus of some collections of offensive/abusive words is collected so that the application is custom and runs live over the internet (this means that everytime the program is run, live tweets are fetched).

1. **Introduction**

**1.1 About Offensive Language**

Offensive language is defined as the speech that attacks a person or a group on the basis of attributes such as race, religion, ethnic religion, national origin, gender, disability, sexual orientation, or gender identity and includes the use of swear words, curse words, cuss words, bad language, strong language, offensive language, crude language, coarse language, foul language, bad words, [blasphemous](https://en.wikipedia.org/wiki/Blasphemy) language, [vulgar](https://en.wikipedia.org/wiki/Vulgarity) language, [lewd](https://en.wikipedia.org/wiki/Lewd) language or [expletives](https://en.wikipedia.org/wiki/Expletive_attributive).

Here are several types of offensive language in this category:-

Taunts/ Rant: These phrases try to condemn or ridicule the reader in general.

References to handicaps: These phrases attack the reader using his\her shortcomings i.e. physically or mentally challenged.

Squalid language: These phrases target sexual fetishes or physical filth of the reader.

Slurs: These phrases try to attack a culture or ethnicity in some way.

Homophobia: These phrases are usually talking about homosexual sentiments.

Racism: These phrases intimidate race or ethnicity of individuals.

Extremism: These phrases target some religion or ideologies.

Colour: Vituperating humans on the basis of their skin complexion.

Other types of offensive language:-

Crude language: expressions that embarrass people, mostly because it refers to sexual matters or excrement.

Disguise: expressions for which the meaning or pronunciation is the same as another more offensive term.

Four-letter words: there are five or six words which consist of only four letters.

Provocative language: expressions that may cause anger or violence.

Taboos: expressions which are forbidden in a certain society/community. There are lots of expressions that are forbidden because of what they refer to, not necessarily there is some particular taboo words used in the expression.

Unrefined language: some expressions that lack polite manners and the speaker is harsh and rude.

Based on the above definitions, flame or offensive language detection, implicitly falls into one or more of the defined cases. Terms which are highly offensive to certain groups are quite prevalent. For example, the African – Americans use this term n\*\*\*\*a quite habitually. People use slangs like h\*e and bi\*\*\* in their music to ridicule anybody or while playing video games. By seeing the usage of these slangs, the developers can use developing tools to better find, filter, and manage this electronic information.

**1.2 Need for Offensive Language Detection**

With the rapid growth of social media, users especially adolescents are spending significant amount of time on various social networking sites to connect with others, to share information and to pursue common interests.

It has been found that 70% of teens are social media sites on a daily basis and nearly one in four teens hit their favorite social media sites 10 or more times a day.19% of teens report that someone has written or posted mean or embarrassing things about them on social networking sites.

As adolescents are more likely to be negatively affected by biased and harmful contents than adults, detecting online offensive contents to protect adolescents online safety becomes an urgent task.

Every day we have to deal with texts (emails or different types of messages) in which there are a variety of attacks and abusive phrases. Social media platforms like Facebook and Twitter have raised concerns about emerging dubious activity such as the intensity of hate, abusive and offensive behavior among its users. However, they are designated as a public space that provides greater opportunities to re-broadcast messages to large audience and even strangers can reply or put their views, opinion and can engage in public debates. In extreme cases, this may lead to violence also.

To overcome all this, an automatic discriminative software for offensive or abusive language detection would be a useful tool and would save a lot of time.

**1.3 Challenges**

Social media is a vital part of our lives and expressing our views or text messaging through the Internet and cellular phones has become a major medium of personal and commercial communication. Every day we have to deal with websites, emails, texts or different types of messages which are inculcated with a plethora of offensive language or abusive language. In the same time, offensive/abusive phrases might attack or offend the users for a variety of reasons. Although a human could recognize these sorts of useless annoying texts among the useful ones, it is not an easy task for computer programs. The social media platforms are unable to tackle the problem of this persistent online abuses, hate speech and offensive language on their platform in an efficient manner. To address concerns on access to offensive contents over Internet, administrators of social media often manually review online contents to detect and delete offensive materials, however, manually reviewing are labor intensive and time consuming.

Therefore, an automatic system for discriminating between regular texts and offensive text would save time and energy during our browsing on the web or in our everyday emails or text messages.

**1.4 Offensiveness Content Filtering Methods in Social Media**

Popular online social networking sites apply several mechanisms to screen offensive contents. For example, Youtube’s safety mode, once activated, can hide all comments containing offensive languages from users. But pre-screened content will still appear—the pejoratives replaced by asterisks, if users simply click "Text Comments." On Facebook, users can add comma-separated keywords to the "Moderation Blacklist." When people include blacklisted keywords in a post and/or a comment on a page, the content will be automatically identified as spam and thus be screened. Twitter client, “Tweetie 1.3,” was rejected by Apple Company for allowing foul languages to appear in users’ tweets. Currently, Twitter does not prescreen users’ posted contents, claiming that if users encounter offensive contents, they can simply block and unfollow those people who post offensive contents.

**1.5 Programming Environment and Tools Used**

Programming Language: Python

Technology: Python modules

Operating System: Windows Operating System

Software Tool: Spyder,Jupyter Notebook, Anaconda Command Prompt

**1.6 Preprocessing and Natural Language Processing**

Processing on the text includes:

* Alphanumeric cleaning
* Special characters removal
* Tokenization (breaking sentences into words)
* Stopwords (most common, short [function words](https://en.wikipedia.org/wiki/Function_word), such as *the*, *is*, *at*, *which*, *on etc.*) removal
* Trailing whitespaces

**Natural language processing (NLP)** is a collective term referring to automatic computational processing of human languages. This includes both algorithms that take human-produced text as input, and algorithms that produce natural looking text as outputs.

**1.7Sentiment Analysis**

Sentiment Analysis is the process of ‘computationally’ determining whether a piece of writing is positive, negative or neutral. It’s also known as **opinion mining**, deriving the opinion or attitude of a speaker. In essence, it is the process of determining the emotional tone behind a series of words, used to gain an understanding of the attitudes, opinions and emotions expressed within an online mention.

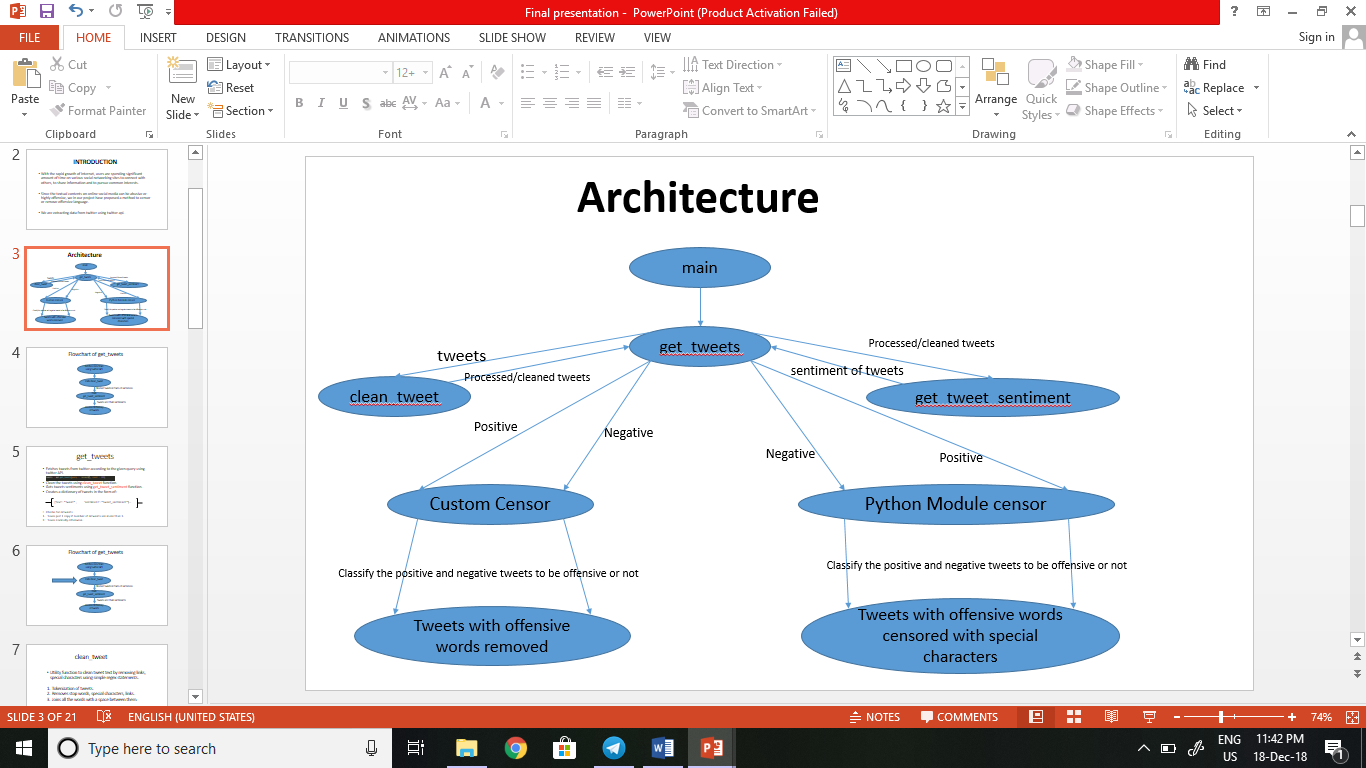
1. **Implementation**

**2.1 Steps Involved**

We follow these major steps in our program:

* Authorize twitter API client.
* Make a GET request to Twitter API to fetch tweets for a particular query.
* Extracting data from twitter using twitter api.
* Text mining using NLP and some other techniques.
* Parse the tweets. Classify each tweet as positive, negative or neutral.
* Creating our own corpus of offensive words: Treat each word and phrase as an entity. Use Bag-of-Words (BoW) in offensiveness detection. The BoW approach treats a text as an unordered collection of words.

**2.2 Architecture**



**2.3 Coding**

First of all, we create a **TwitterClient** class. This class contains all the methods to interact with Twitter API and parsing tweets. We use **\_\_init\_\_**function to handle the authentication of API client.

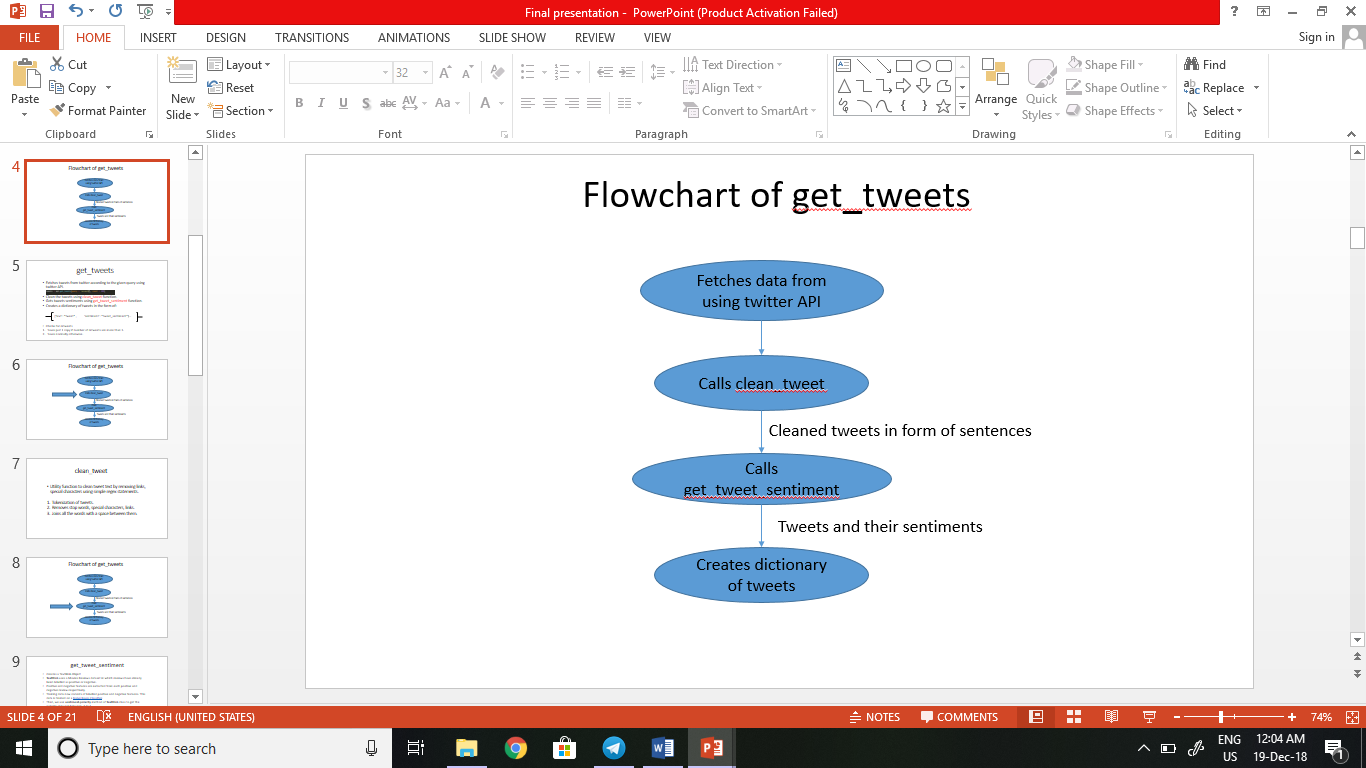
**2.3.1 get\_tweets**

* Fetches tweets from twitter according to the given query using twitter API.
* 
* Clean the tweets using **clean\_tweet**function.
* Gets tweets sentiments using **get\_tweet\_sentiment**function.
* Creates a dictionary of tweets in the form of:

[‘text’: \*tweet\* , ‘sentiment’: \*tweet\_sentiment\*]…

* Checks for retweets

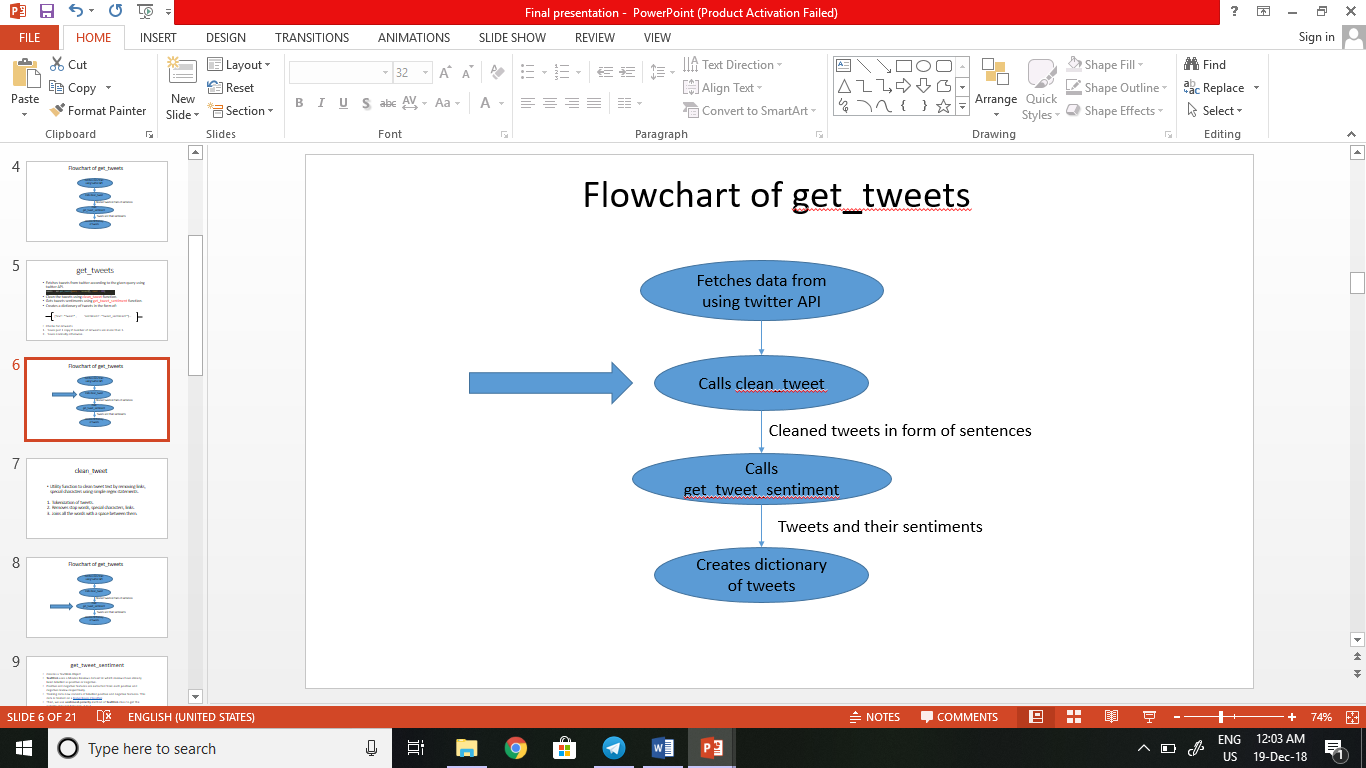
1. Saves just 1 copy if number of retweets are more than 1.
2. Saves normally otherwise.



**2.3.2 clean\_tweet**

Utility function to clean tweet text by removing links, special characters using simple regex statements.

1. Tokenization of tweets.
2. Removes stop words, special characters, links.
3. Joins all the words with a space between them.



**2.3.3 get\_tweet\_sentiment**

Creates a TextBlob Object. TextBlob is actually a high level library built over top of [NLTK](http://www.nltk.org/) library.

analysis = TextBlob(self.clean\_tweet(tweet))

First we call **clean\_tweet**method, then as we pass **tweet** to create a **TextBlob** object, following processing is done over text by textblob library:

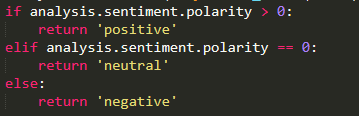
* Tokenize the tweet i.e., split words from body of text.
* Remove stopwords from the tokens.
* Do POS (part of speech) tagging of the tokens and select only significant features/tokens like adjectives, adverbs, etc.
* Pass the tokens to a **sentiment classifier**which classifies the tweet sentiment as positive, negative or neutral by assigning it a polarity between -1.0 to 1.0 .

Here is how **sentiment classifier** is created:

* **TextBlob** uses a Movies Reviews dataset in which reviews have already been labelled as positive or negative.
* Positive and negative features are extracted from each positive and negative review respectively.
* Training data now consists of labelled positive and negative features. This data is trained on a [Naive Bayes Classifier](https://en.wikipedia.org/wiki/Naive_Bayes_classifier).

Then, we use **sentiment.polarity** method of **TextBlob** class to get the polarity of tweet between -1 to 1.

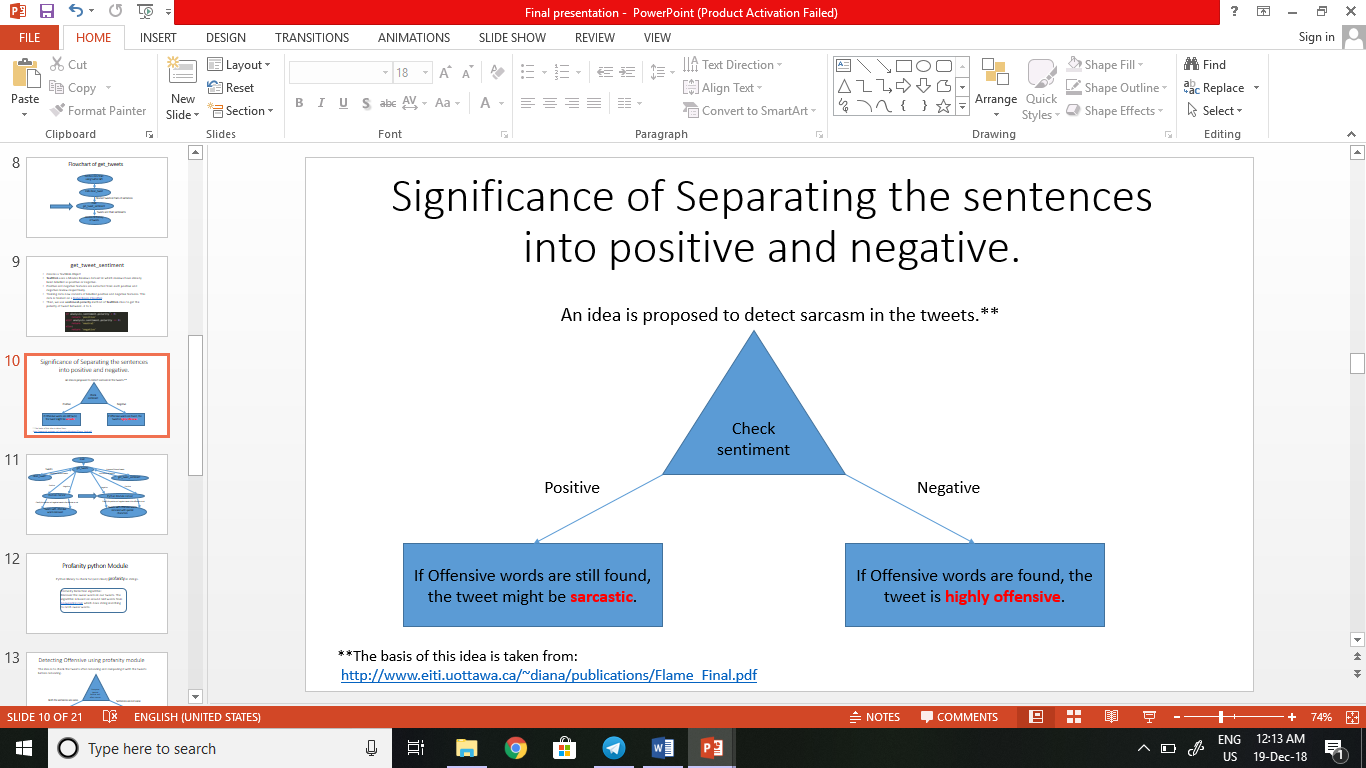
Then, we classify polarity as:



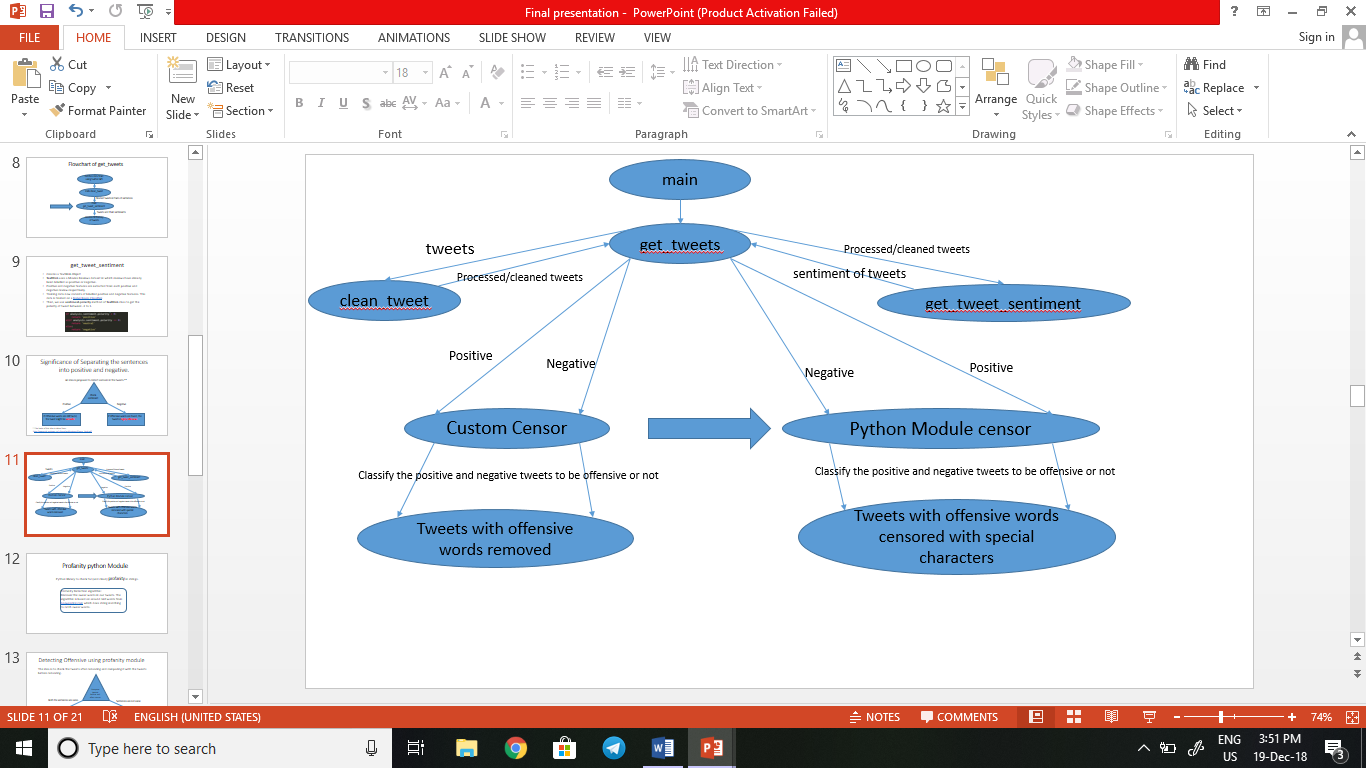
Finally, parsed tweets are returned.

**2.3.4 Significance of Separating the sentences into positive and negative.**

An idea is proposed to detect sarcasm in the tweets.



**2.3.5 Profanity python Module**



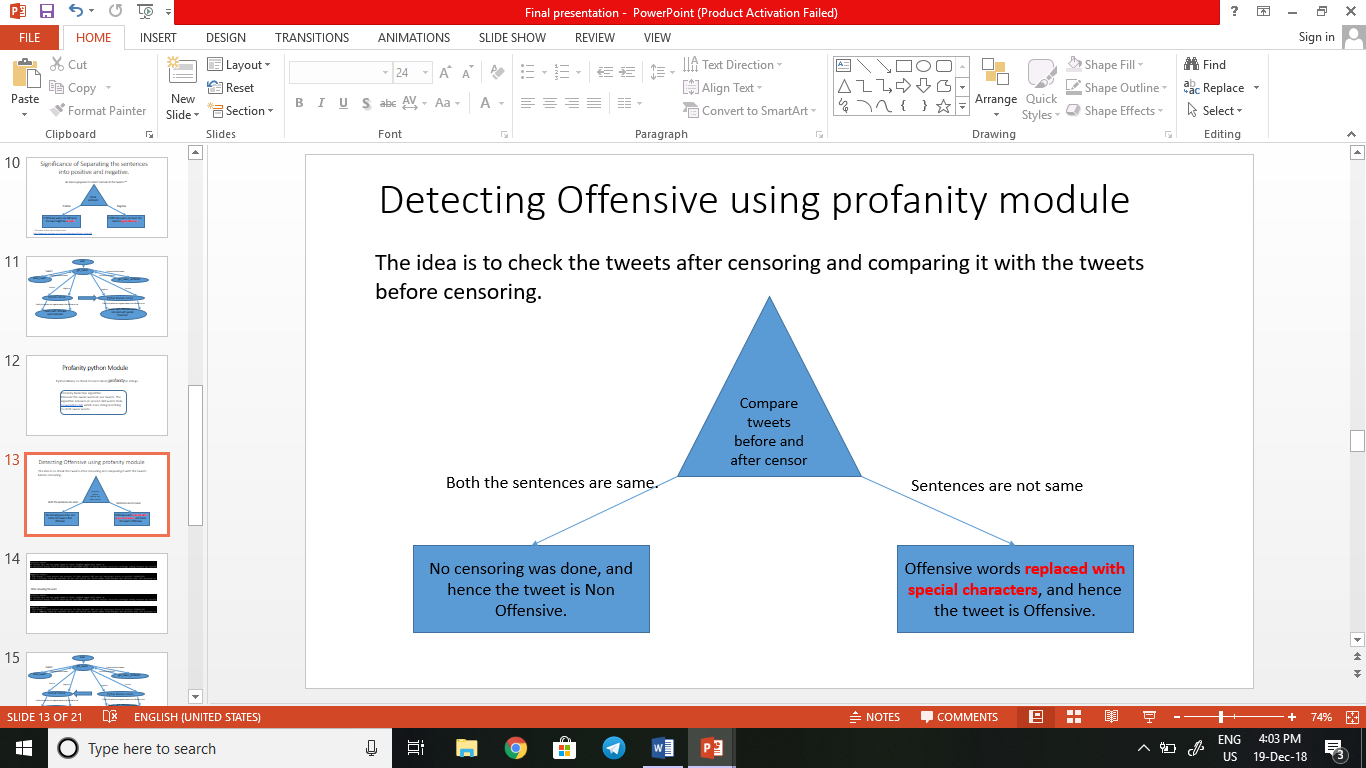
Python library to check for (and clean) profanity in strings.

Profanity Detection algorithm:

Discover the swear words in our tweets. The algorithm is based on around 340 words from [noswearing.com](https://www.noswearing.com/) which does string matching to catch swear words.

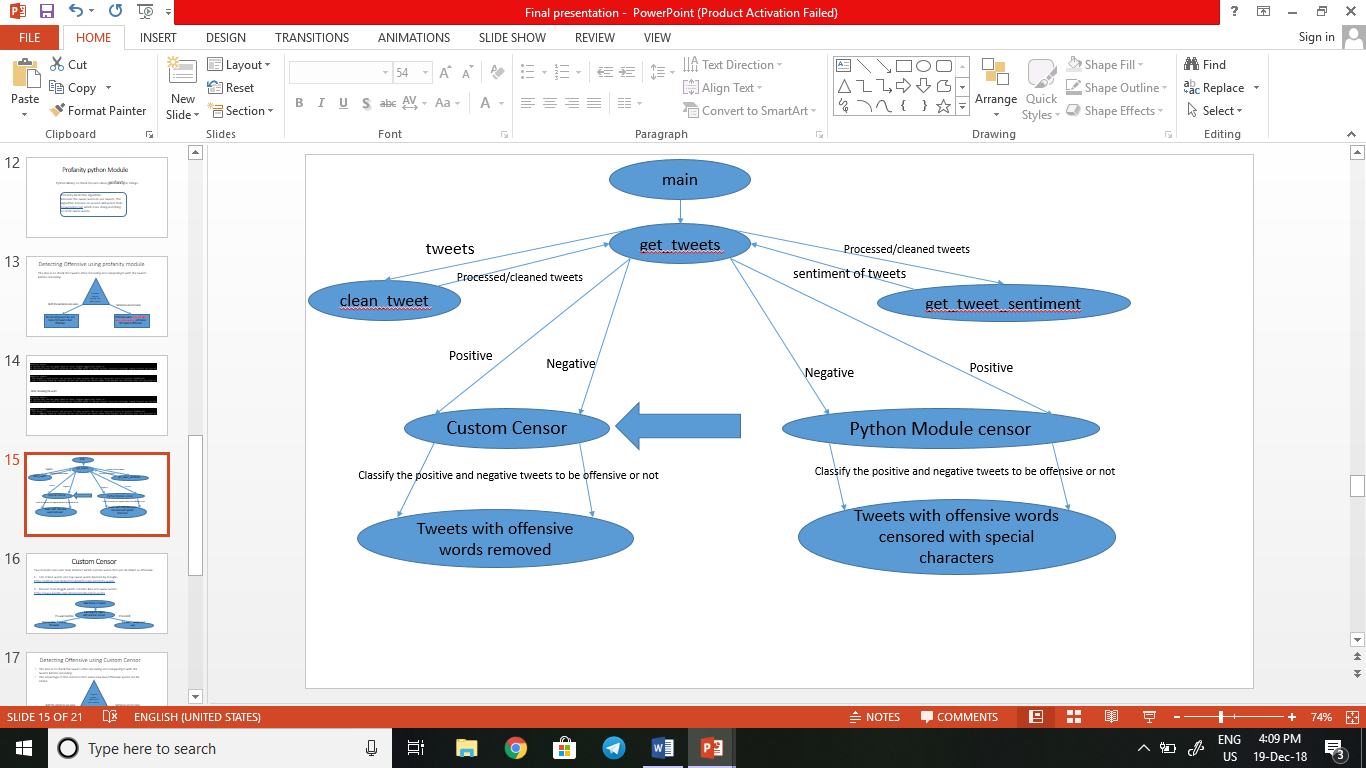
**2.3.6 Detecting Offensive using profanity module**

The idea is to check the tweets after censoring and comparing it with the tweets before censoring.



There are some sentences that the profanity module doesn’t detect as offensive. To improve that working, we introduce our custom censor.

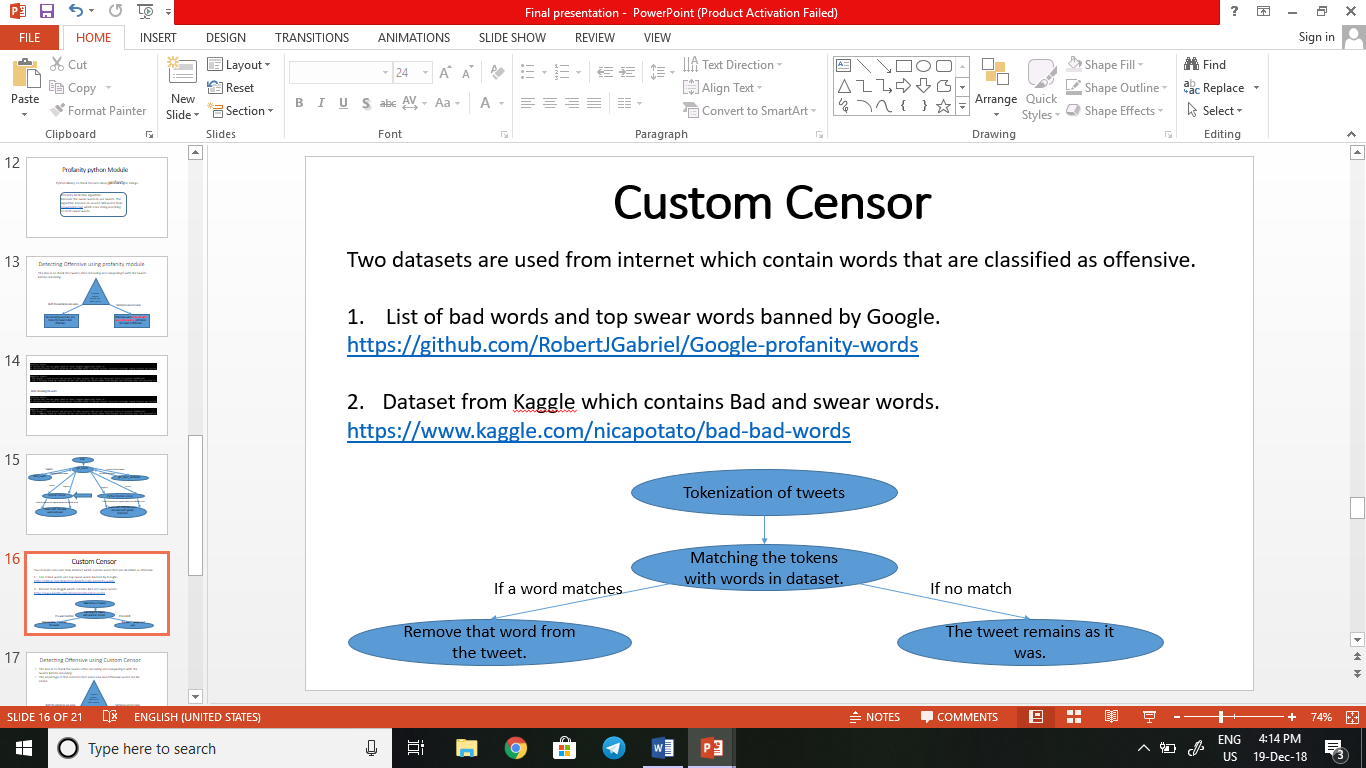
**v2.3.7 Custom Censor**



Two datasets are used from internet which contain words that are classified as offensive.

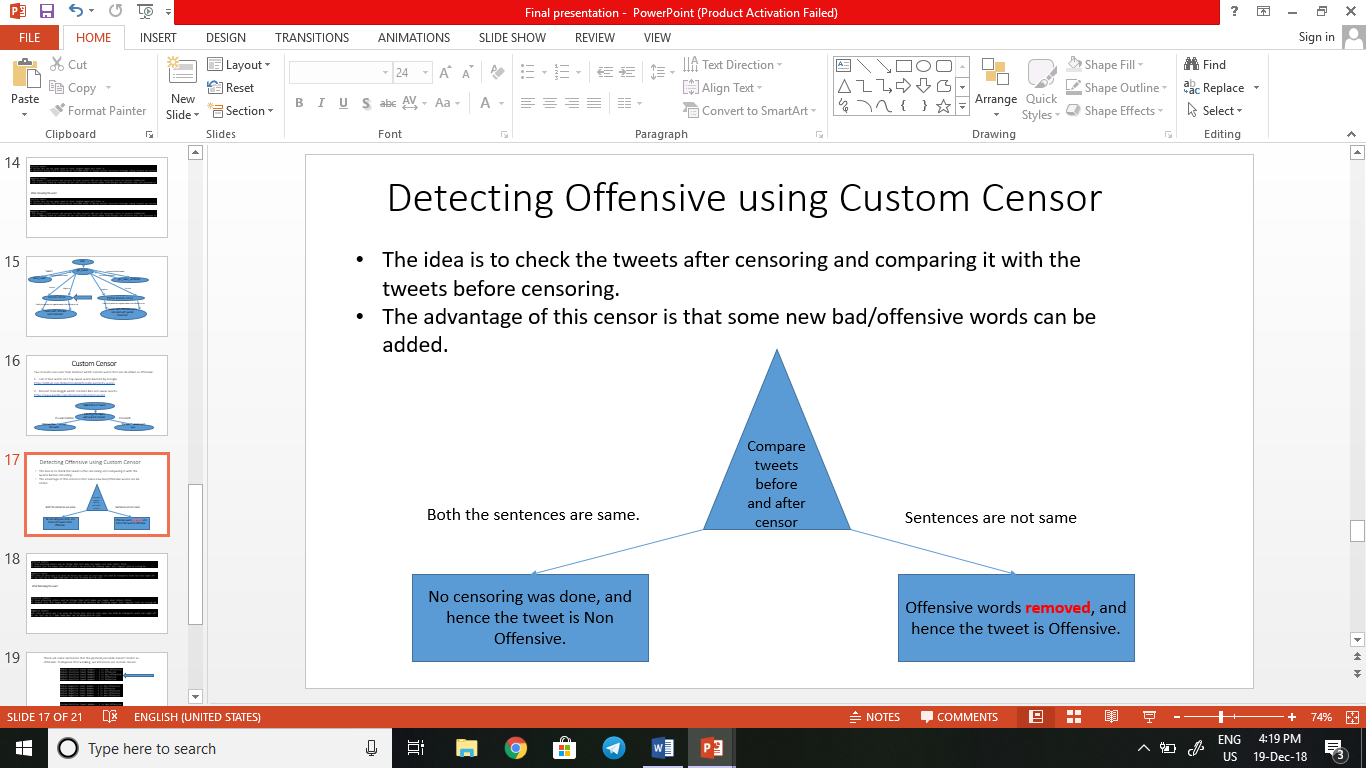
1. List of bad words and top swear words banned by Google.

1. Dataset from Kaggle which contains Bad and swear words.



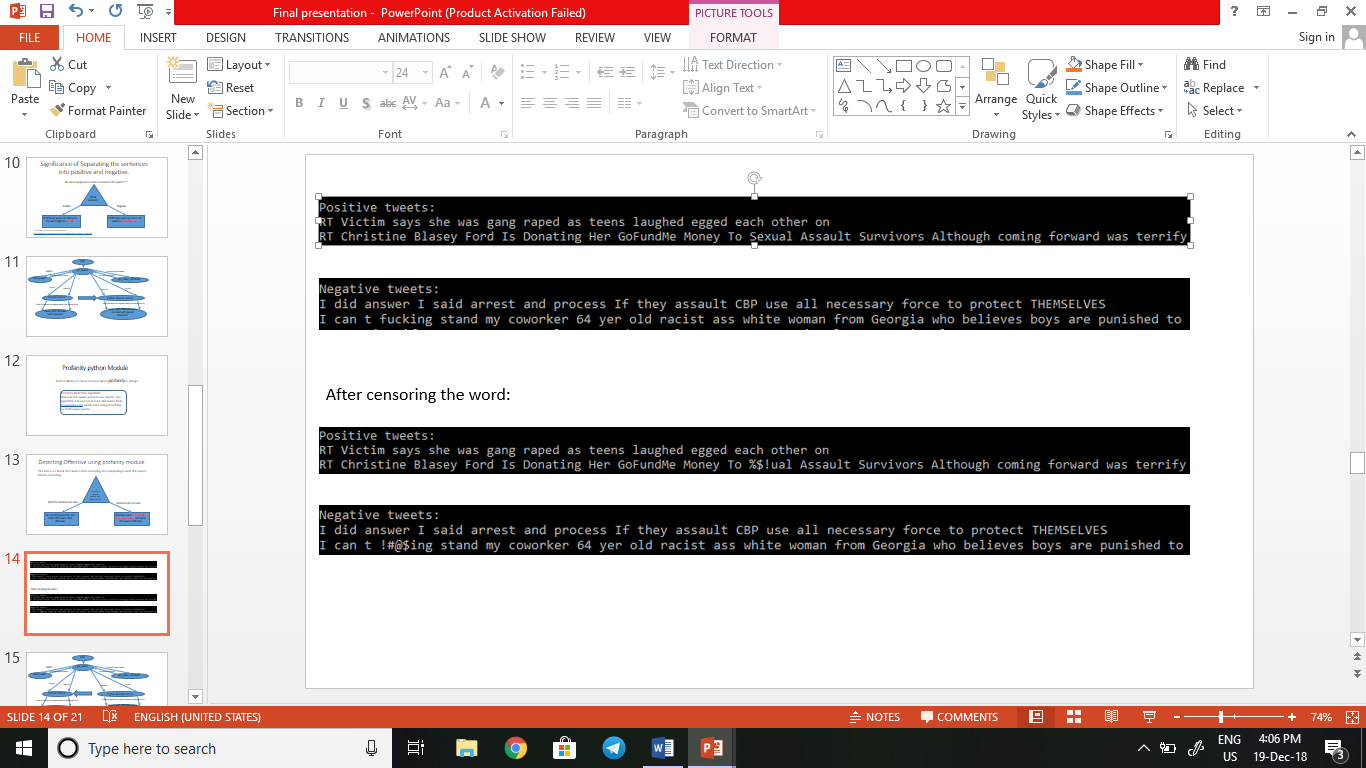
**2.3.8 Detecting Offensive using Custom Censor**

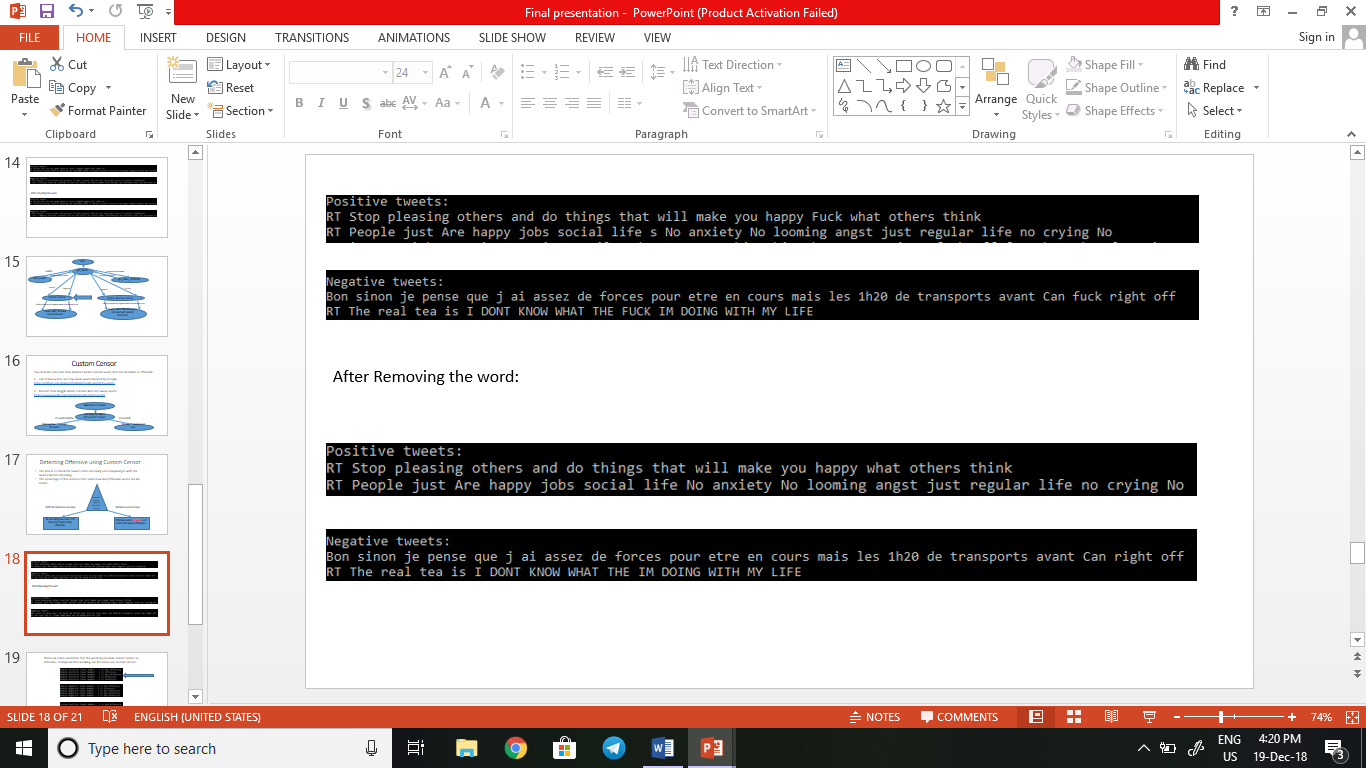
* The idea is to check the tweets after censoring and comparing it with the tweets before censoring.
* The advantage of this censor is that new bad/offensive words can be added to the dataset.



1. **Experimental Results**

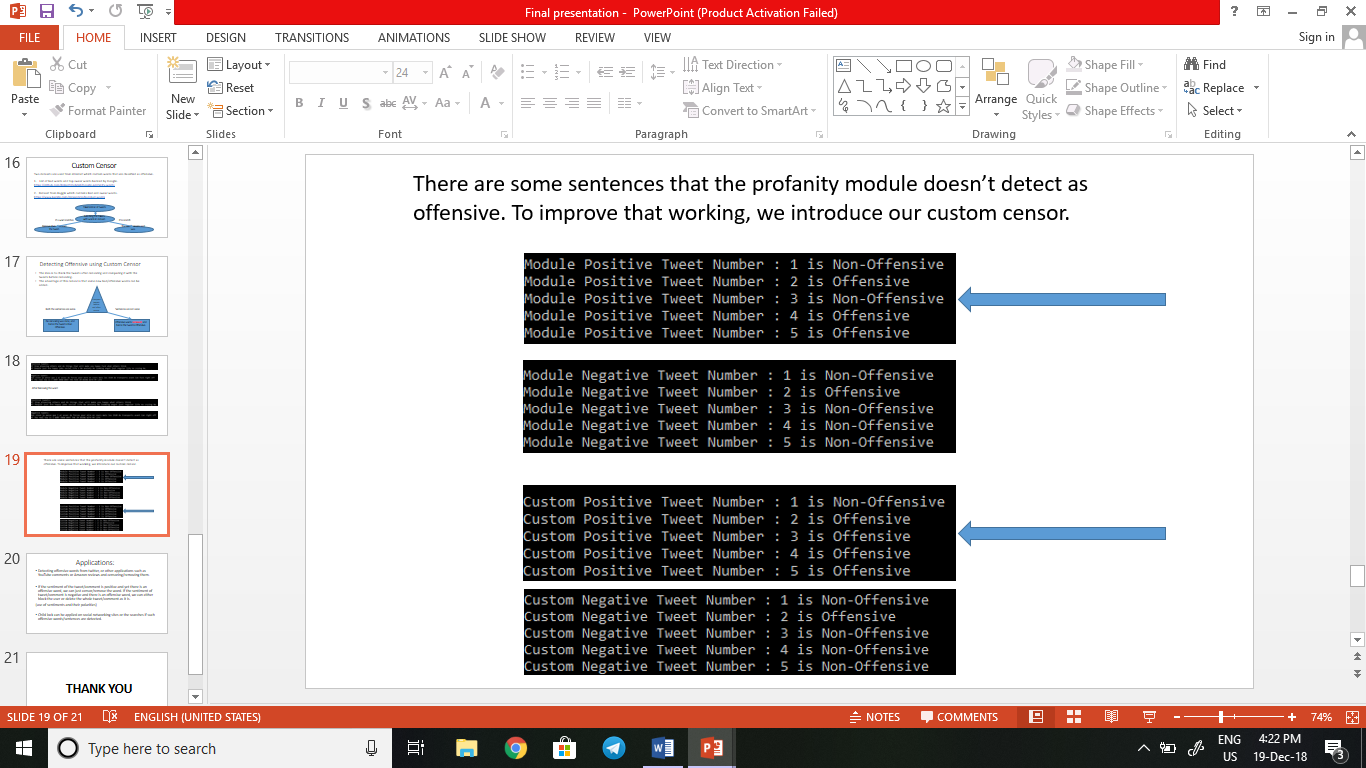
**3.1 Results from Profanity python Module**



**3.2Results from Custom Censor**

**3.3 Advantage of Custom Censor**

There are some sentences that the profanity module doesn’t detect as offensive but our custom censor does. Also, new bad/offensive words can be added to the dataset.



1. **Conclusion and Future Work**
   1. **Conclusion**

In this project, we were detecting offensive and sarcastic words/phrases from tweets which were being fetched live using twitter API.

An idea to detect sarcasm had been proposed and we were successfully able to remove/censor the offensive words from the sentences as needed. Profanity module did work but the classification models used in the profanity module did not give a high accuracy whereas the custom module (method) built, worked far better. Using sentimental analysis we were also able to detect the sentences which were highly offensive or lightly offensive.

**4.2 Applications and Future Work**

* Detecting offensive words from twitter, or other applications such as YouTube comments or Amazon reviews and censoring/removing them.
* If the sentiment of the tweet/comment is positive and yet there is an offensive word, we can just censor/remove the word. If the sentiment of tweet/comment is negative and there is an offensive word, we can either block the user or delete the whole tweet/comment as it is.

(use of sentiments and their polarities)

* Child lock can be applied on social networking sites or the searches if such offensive words/sentences are detected.

**References**

[1] MohdAmjad ,MohdZeeshan Ansari and NasimAlam,“An MLP based Approach of Hate Speech Detection on Twitter”, Department of Computer Engineering, JamiaMilliaIslamia, New Delhi, India.

[2] A.H. Razavi, D. Inkpen, S. Uritsky, and S. Matwin, "Offensive language detection using multi-level classification," Advances in Artificial Intelligence, vol. 6085/2010, pp. 16-27, 2010.

[3] <https://github.com/RobertJGabriel/Google-profanity-words>

[4]<https://www.kaggle.com/nicapotato/bad-bad-words>

[5] [https://en.wikipedia.org/wiki/Hate speech](https://en.wikipedia.org/wiki/Hate%20speech)

[6] <https://en.wikipedia.org/wiki/Profanity>

CONTENTS

* + 1. Introduction…………………………………………………………………………………...1
  1. About Offensive Language…………………………………………………………...1
  2. Need for Offensive Language Detection……………………………………………...2
  3. Challenges…………………………………………………………………………….3
  4. Offensiveness Content Filtering Methods in Social Media…………………………..4
  5. Programming Environment and Tools Used…………………………………………4
  6. Preprocessing and Natural Language Processing…………………………………….4
  7. **Sentiment Analysis…………………………………………………………………...5**
     1. Implementation…………………………………………………………….…………………5
  8. Steps Involved………………………………………………………………...………5
  9. Architecture………………………………………………………………….……….6
  10. Coding………………………………………………………………………….…….6
      1. get\_tweets…………………………………………………………………….…..7
      2. clean\_tweet……………………………………………………………………….8
      3. get\_tweet\_sentiment……………………………………………………………...9
      4. Significance of Separating the sentences into positive and negative……….……10
      5. Profanity python Module……………………………………………………......11
      6. Detecting Offensive using profanity module………………………………....…12
      7. Custom Censor…………………………………………………………………..13
      8. Detecting Offensive using Custom Censor……………………………..……….14

1. Experimental Results…………………………………………………………………….….15
   1. Results from Profanity python Module……………………………………….…….15
   2. Results from Custom Censor……………………………………………….……....16
   3. Advantage of Custom Censor………………………………………………………17

15

1. Conclusion and Future Work………………………………………………………………18
   1. Conclusion…………………………………………………………………………18
   2. Applications and Future Work…………………………………….....……………18

References………………………………………….……………………………………………………………………………….19