Table of Contents

1. Data Intake Report

Name: Hate Speech detection using Transformers
Report date: 10 March 2023
Internship Batch:LISUM17
Version:<1.0>
Data intake by:Surbhi Sharma
Data intake reviewer:<intern who reviewed the report>
Data storage location: https://github.com/SurbhiS19/Hate_Speech_Transformers

Tabular data details:

Train Data

Total number of observations	31962
Total number of files	1
Total number of features	2
Base format of the file	.csv
Size of the data	3.1 MB

Test Data

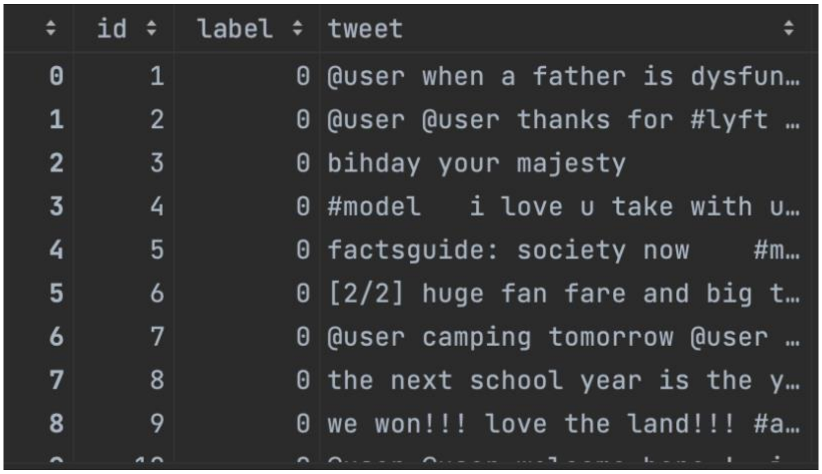
Total number of observations	17197
Total number of files	1
Total number of features	1
Base format of the file	.csv
Size of the data	1.6 MB

2. Problem Description and business understanding

The term hate speech is understood as any type of verbal, written or behavioural communication that attacks or uses derogatory or discriminatory language against a person or group based on what they are, in other words, based on their religion, ethnicity, nationality, race, colour, ancestry, sex or another identity factor. With the social media usage at its peak, identifying such tweets have become extremely crucial for every social media platform. Hate speech on social media can have harmful effect on a person/community. Hate speech is often used to spread hate and bigotry. It can also be used to intimidate and threaten people. It can make people feel isolated, anxious, and scared. It can also lead to hate crimes. Hate speech can also damage relationships between different groups of people. Detection of hate speech is important because it can help prevent these harmful effects.

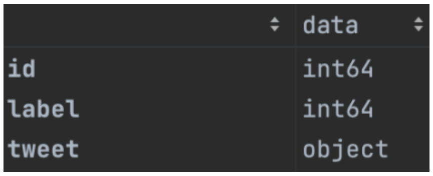
In this problem, we will create a hate speech detection model with Machine Learning and Python. Hate Speech Detection is generally a task of sentiment classification. So for training, a model that can classify hate speech from a certain piece of text can be achieved by training it on a data that is generally used to classify sentiments. So for the task of hate speech detection model, we will use the Twitter tweets to identify tweets containing Hate speech.

3. **Dataset**:
   Data has 3 columns: id(id of the tweet), label(=0 if not a hate speech and =1 if it is a hate speech) and tweet(the string containing the tweet. Following is a snapshot of first few rows of the dataset.



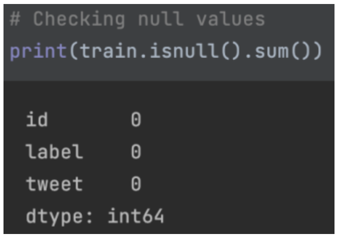
	id	label	tweet
0	1	0	@user when a father is dysfun…
1	2	0	@user @user thanks for #lyft …
2	3	0	bihday your majesty
3	4	0	#model i love u take with u…
4	5	0	factsguide: society now #m…
5	6	0	[2/2] huge fan fare and big t…
6	7	0	@user camping tomorrow @user …
7	8	0	the next school year is the y…
8	9	0	we won!!! love the land!!! #a…

4. **Datatypes**
   The datatype of id and label is int64 (integer). The data type of tweet is string.



	data
id	int64
label	int64
tweet	object

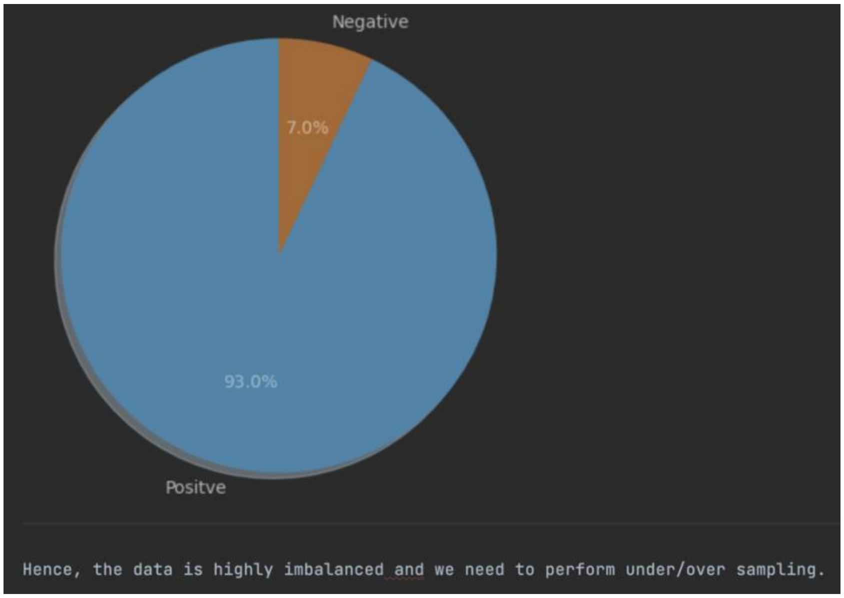
5. **Data Problems**
   - Null values: This dataset has no null values.



```
# Checking null values
print(train.isnull().sum())

 id       0
 label    0
 tweet    0
 dtype: int64
```

   - Test data csv file does not have label column. Hence, we cannot use it for further analysis.
   - Data imbalance: There is a heavy imbalance in the data as shown in the figure below. There are only 7% hate tweets as compared to 93% positive(non-hate) tweets. Hence when sampling the data for model building we need to apply under/over sampling technique.



Hence, the data is highly imbalanced and we need to perform under/over sampling.

- Special characters in tweets: As shown in the first figure showing the first few rows of the data set, several tweets have special characters such as @,#, urls etc. They need to be cleaned using regex.

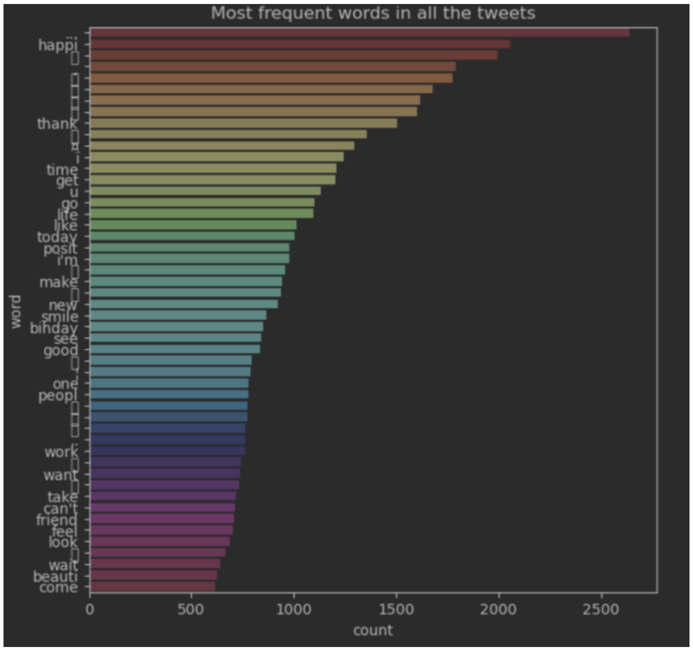
6. Data Transformations

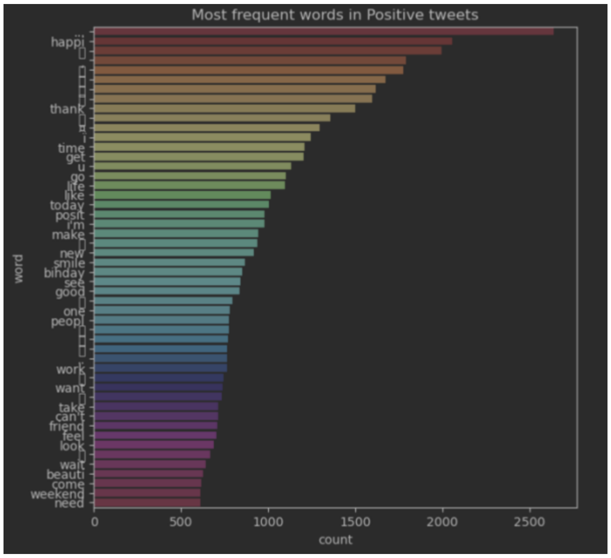
The ids will not be used for model building and labels have only 0 and 1 values. Hence these two columns need not to be transformed. However, the tweets need to be preprocessed. The data needs to be cleaned by removing urls or any special characters which doesn't make sense.

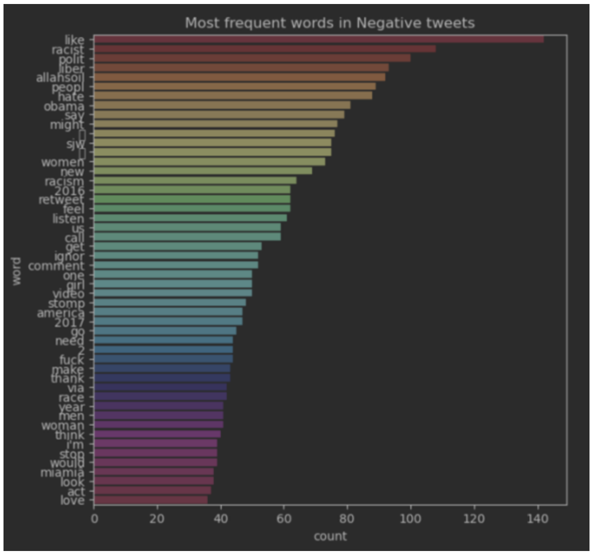
Data preprocessing is one of the critical steps in any machine learning project. It includes cleaning and formatting the data before feeding it into any ML algorithm. For NLP, following task needs to be performed for pre-processing:

- Tokenizing the string
- Lowercasing
- Removing stop words and punctuations
- Stemming

We will then make a dictionary with word counts, with most frequest words in positive tweets, negative tweets and across all the tweets as follows.







7. Model Development

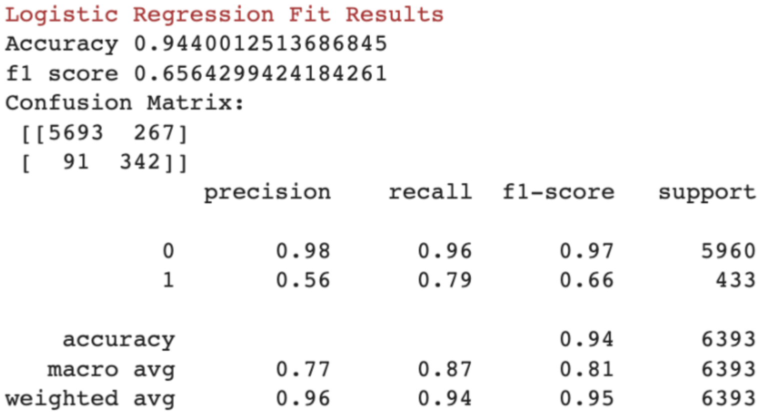
Since the data is imbalanced, oversampling has been performed from the negative tweets class to obtain balanced dataset. Firstly, data has been split into train(80%) and test sets(20%) and then oversampling has been performed for the training dataset.

Following model have been trained for the given problem:

- Logistic Regression
- Support Vector Machines
- Naïve Bayes

- Random Forest
- Transformers

a. Logistic Regression results:

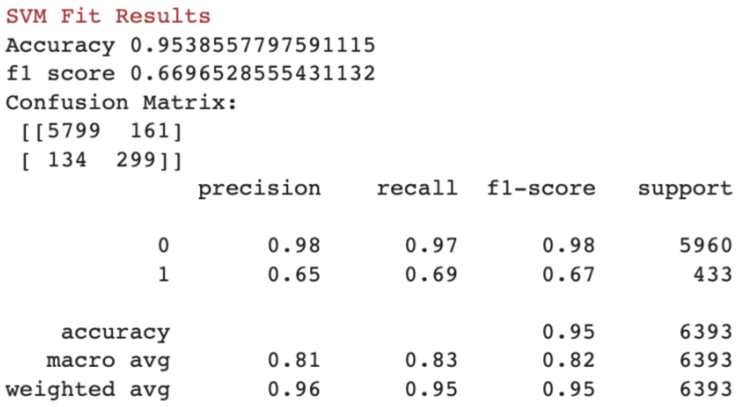


```
Logistic Regression Fit Results
Accuracy 0.9440012513686845
f1 score 0.6564299424184261
Confusion Matrix:
 [[5693  267]
 [  91  342]]
              precision    recall  f1-score   support

           0       0.98      0.96      0.97      5960
           1       0.56      0.79      0.66       433

    accuracy                           0.94      6393
   macro avg       0.77      0.87      0.81      6393
weighted avg       0.96      0.94      0.95      6393
```

b. SVM results:

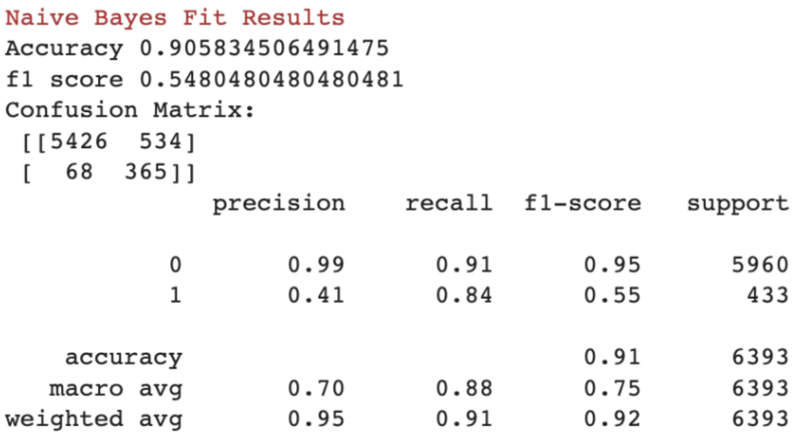


```
SVM Fit Results
Accuracy 0.9538557797591115
f1 score 0.6696528555431132
Confusion Matrix:
 [[5799  161]
 [ 134  299]]
              precision    recall  f1-score   support

           0       0.98      0.97      0.98      5960
           1       0.65      0.69      0.67       433

    accuracy                           0.95      6393
   macro avg       0.81      0.83      0.82      6393
weighted avg       0.96      0.95      0.95      6393
```

c. Naïve Bayes:

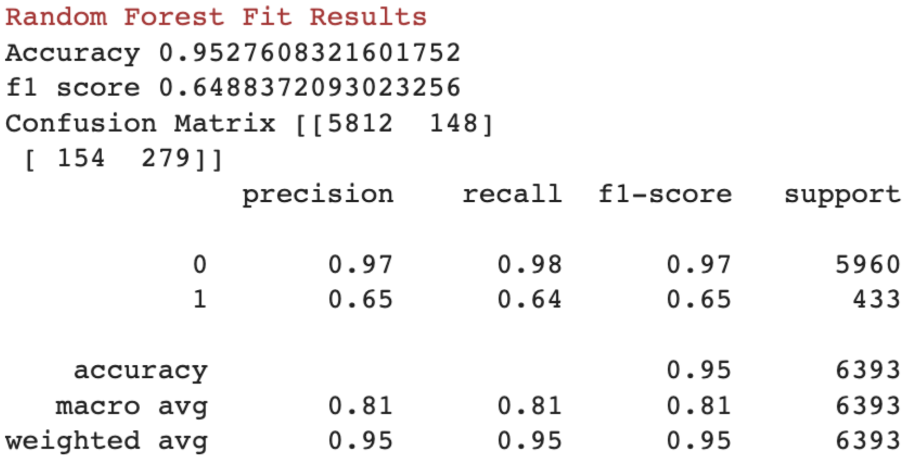


```
Naive Bayes Fit Results
Accuracy 0.905834506491475
f1 score 0.5480480480480481
Confusion Matrix:
 [[5426  534]
 [  68  365]]
              precision    recall  f1-score   support

           0       0.99      0.91      0.95      5960
           1       0.41      0.84      0.55       433

    accuracy                           0.91      6393
   macro avg       0.70      0.88      0.75      6393
weighted avg       0.95      0.91      0.92      6393
```

d. Random Forest



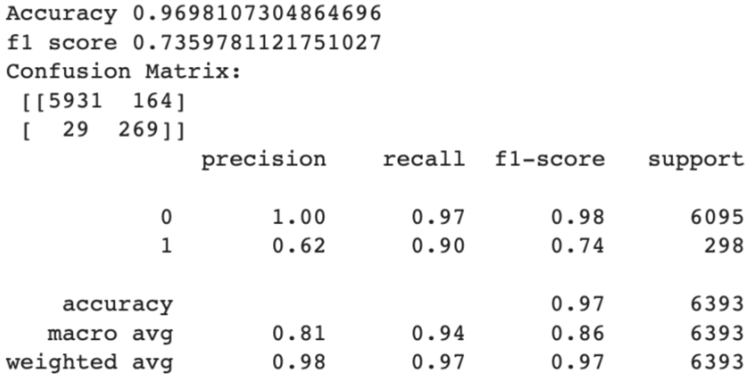
```
Random Forest Fit Results
Accuracy 0.9527608321601752
f1 score 0.6488372093023256
Confusion Matrix [[5812  148]
 [ 154  279]]
             precision    recall  f1-score   support

           0       0.97      0.98      0.97      5960
           1       0.65      0.64      0.65       433

    accuracy                           0.95      6393
   macro avg       0.81      0.81      0.81      6393
weighted avg       0.95      0.95      0.95      6393
```

e. Transformers:

A pretrained model ‘DistilBertForSequenceClassification’ has been retrained for the sentiment analysis. The data is tokenized using distilbert-base-uncased tokenizer. Following result has been obtained for the transformer model fitted for twitter data:



```
Accuracy 0.9698107304864696
f1 score 0.7359781121751027
Confusion Matrix:
 [[5931  164]
 [  29  269]]
              precision    recall  f1-score   support

           0       1.00      0.97      0.98      6095
           1       0.62      0.90      0.74       298

    accuracy                           0.97      6393
   macro avg       0.81      0.94      0.86      6393
weighted avg       0.98      0.97      0.97      6393
```

8. Final Recommendations:

The following results show that transformers provide the highest accuracy as well as f1 score for this use case.

Model	LR	SVM	NB	RF	Transformers	
Accuracy	0.94	0.95	0.90	0.95	0.97	
F1 score	0.65	0.66	0.54	0.65	0.74	