	text	labels
0	@realDonaldTrump This is one of the worst time...	0
1	How about the crowd in Oval in today's #AUSvIN...	1
2	@skroskz @shossy2 @JoeBiden Biden & his so...	0
3	#etsy shop: Benedict Donald so called presiden...	1
4	@realDonaldTrump Good build a wall around Arka...	0
...	...	...
5261	@ICC should allow ms dhoni to keep glove. It i...	1
5262	Trump on avoiding movie pirating: 'of course y...	1
5263	I noticed recently Jamie Oliver's restaurants ...	1
5264	#TeamIndia geared up is okay. What's on the GL...	0
5265	Is this the same piece of paper McCarthy used ...	1

HTSPC- Hate Speech Classification

AI CROWD CHALLENGE

RISHAB(2019201050) |
SMAI Assignment |
April 30th, 2020

ABOUT THE CHALLENGE:

There is tonnes of hate speech being posted everyday on social media by different users. Facebook runs its hate speech detection algorithm and actively removes content which is hateful. The objective is to build a machine learning model that classifies a piece of text as hate speech or not.

- The problem statement is that we are given a piece of text, which we need to classify into hate speech or not hate speech. It is a binary classification problem with labels "HOF"(0) denoting hate speech and "NOT"(1) denoting non hateful sentences. Numbers in brackets denote the class label.

ABOUT THE DATSET

The training data provided was of twitter tweets. There were 5266 data points each associated with a label of 0 or 1.

Lets see the dataset:

	text	labels
0	@realDonaldTrump This is one of the worst time...	0
1	How about the crowd in Oval in today's #AUSvIN...	1
2	@skroskz @shossy2 @JoeBiden Biden & his so...	0
3	#etsy shop: Benedict Donald so called presiden...	1
4	@realDonaldTrump Good build a wall around Arka...	0
...	...	...
5261	@ICC should allow ms dhoni to keep glove. It i...	1
5262	Trump on avoiding movie pirating: 'of course y...	1
5263	I noticed recently Jamie Oliver's restaurants ...	1
5264	#TeamIndia geared up is okay. What's on the GL...	0
5265	Is this the same piece of paper McCarthy used ...	1

Analysis:

The data provided was raw and was not preprocessed.

It required some preprocessing before further feeding it to the model.

The data contained URL'S ,usernames,emojis etc.

The steps which I followed for preprocessing were:

1. Removal of hashtags and mentions.
2. Removal of digits and punctuations
3. Removal of URL's
4. Conversion to lower case.
5. Removal of emoji.
6. Removal of excess whitespaces.
7. Lemmatization (converting words to base word and removal of stopwords)
8. Joining back the preprocessed words.

Lets see the step by step conversion of a training sample.

```
original text:
According to The Times and the Daily Mail, Brussels believe that whoever wins the Conservative leadership race will agree to
an extension to the UK's departure from the EU..  https://t.co/Y4JFZcXprQ    #BorisJohnsonShouldNotBePM   #BollocksToBoris
#BollockstoBrexit https://t.co/Lg1MksA2eG
============================
after removing url
According to The Times and the Daily Mail, Brussels believe that whoever wins the Conservative leadership race will agree to
an extension to the UK's departure from the EU.. BorisJohnsonShouldNotBePM BollocksToBoris BollockstoBrexit
=========================
after removing digits and punctuations :
According to The Times and the Daily Mail  Brussels believe that whoever wins the Conservative leadership race will agree to
an extension to the UK s departure from the EU   BorisJohnsonShouldNotBePM BollocksToBoris BollockstoBrexit
=========================
after removing http like strings
According to The Times and the Daily Mail  Brussels believe that whoever wins the Conservative leadership race will agree to
an extension to the UK s departure from the EU   BorisJohnsonShouldNotBePM BollocksToBoris BollockstoBrexit
======================================
after changing into lower case:
according to the times and the daily mail  brussels believe that whoever wins the conservative leadership race will agree to
an extension to the uk s departure from the eu   borisjohnsonshouldnotbepm bollockstoboris bollockstobrexit
======================================
after removing emojis:
according to the times and the daily mail  brussels believe that whoever wins the conservative leadership race will agree to
an extension to the uk s departure from the eu   borisjohnsonshouldnotbepm bollockstoboris bollockstobrexit
======================================
after removing extra white spaces:
['according', 'to', 'the', 'times', 'and', 'the', 'daily', 'mail', 'brussels', 'believe', 'that', 'whoever', 'wins', 'the',
'conservative', 'leadership', 'race', 'will', 'agree', 'to', 'an', 'extension', 'to', 'the', 'uk', 's', 'departure', 'from',
'the', 'eu', 'borisjohnsonshouldnotbepm', 'bollockstoboris', 'bollockstobrexit']
======================================
after lemitization:
['according', 'times', 'daily', 'mail', 'brussels', 'believe', 'whoever', 'wins', 'conservative', 'leadership', 'race', 'agr
ee', 'extension', 'uk', 'departure', 'eu', 'borisjohnsonshouldnotbepm', 'bollockstoboris', 'bollockstobrexit']
======================================
final text :
according times daily mail brussels believe whoever wins conservative leadership race agree extension uk departure eu borisj
ohnsonshouldnotbepm bollockstoboris bollockstobrexit
```

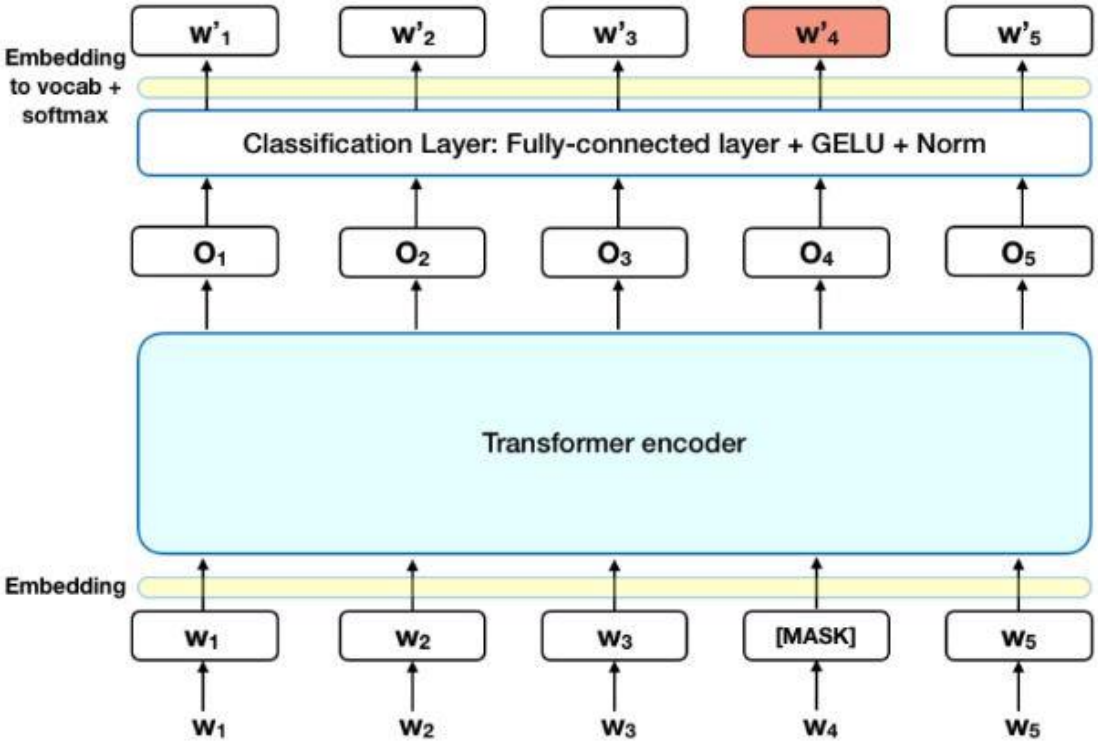
The Plan of Attack:

Since it is a text classification problem, there can be many ways to solve this.

The different approaches which I tried to solve this problem are:

1. **Use Sci-Kit learn Vectorizers and Classifiers** – Sklearn provides several vectorizers like TF-IDF Vectorizer and Count Vectorizer. After the above transformations we fed the data to these vectorizers and set the max_features parameter to 7000.However such high dimensional data was not needed so I applied PCA to preserve data with keeping the 99% variance. I saw that the dimensionality was reduced to 559.After reducing the data it was fed to various classifiers. The classifiers I used were Support Vector Machines with different Kernels like – rbf,linear,polynomial kernel of degree 9.Random Forest and other ensemble methods like stacking classifiers with all the above classifiers as the estimators ,the accuracy observed was 68.9 %.Logistic Regression was also used with an observed accuracy of 69.6%.While using Count Vectorizer since it uses probabilities I tried with Naïve Bayes and Gradient Boosting Classifier,but again the accuracy didn't rise above 69.5%.Since these results were not up to the mark I tried my hand with Neural Networks.
2. **BERT-** It is a state of the art model for Natural Language Processing.BERT stands for Bidirectional Encoder Representations from Transformers. BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary. As opposed to directional models, which read the text input

sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).Lets see how BERT works internally in a diagram.



The BERT loss function takes into consideration only the prediction of the masked values and ignores the prediction of the non-masked words. As a consequence, the model converges slower than directional models, a characteristic which is offset by its increased context awareness.

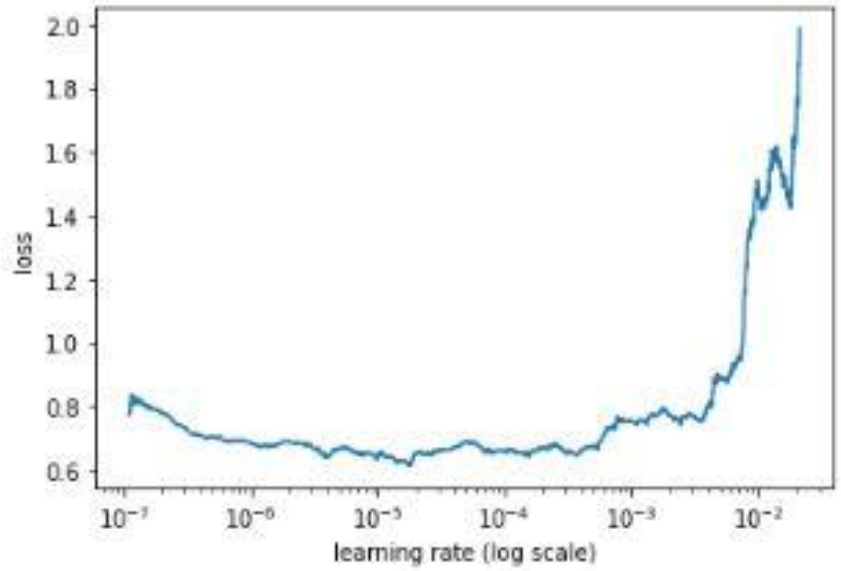
BERT works fairly well even when number of training data is less.

So in my case also, BERT proved to be the best way to classify data. Gave 75% and above accuracy with an appreciable F1-score.

The library used was ktrain which has inbuilt support for BERT for data transformation.

The accuracy increased after preprocessing was done on the data which are stated above.

I also plotted how BERT behaves with a change in learning rate.



KTrain provides an excellent interface to use with BERT. It has a function called autofit which which there is a parameter called ‘early stopping’ which when set to 1 stops the model before it starts to overfit and does validation by changing various parameters inside automatically while training. A snapshot how early stopping was used in my model is shown below.

```
1 learner.autofit(2e-5,4,early_stopping=1)
```

```
begin training using triangular learning rate policy with max lr of 2e-0
5...
Train on 4739 samples, validate on 527 samples
Epoch 1/4
4739/4739 [==============================] - 502s 106ms/sample - loss: 0.55
51 - accuracy: 0.7122 - val_loss: 0.6183 - val_accuracy: 0.6926
Epoch 2/4
4734/4739 [============================>.] - ETA: 0s - loss: 0.4579 - accur
acy: 0.7818Restoring model weights from the end of the best epoch.
4739/4739 [==============================] - 502s 106ms/sample - loss: 0.45
77 - accuracy: 0.7818 - val_loss: 0.6997 - val_accuracy: 0.6907
Epoch 00002: early stopping
Weights from best epoch have been loaded into model.
```

Results

After using BERT and using it on test data.It performed well in the challenge with an F1 score of 0.887 and Accuracy of 81.7 %.

BERT proved to be quite effective for this task.

ABOUT THE CODE:

PYTHON VERSION USED 3.7

TRAIN FILE AND TEST FILE PATH ARE HARDCODED IN MACROS 'TRAIN_DATA_FILE' AND 'TEST_DATA_FILE' as 'training_data.csv' and 'test_final.csv' RESPECTIVELY.

THE CODE IS DIRECTLY DOWNLOADED AS PY FROM A IPYNB SO IT IS NOT BE USED DIRECTLY

p.s USE THE FUNCTIONS REQUIRED BY TAKING THE SNIPPETS FROM THE PY FILE.