TOXICITY OF WIKIPEDIA COMMENTS

-Disclaimer: the dataset for this project contains text that may be considered profane, vulgar, or offensive.

Wikipedia is a multilingual online encyclopedia based on open collaboration through a model of content edited by web-based applications like web browsers. It is the largest and most popular general reference work on the World Wide Web.

Open collaboration has its own difficulties since some authors insert contents that are toxic, which means they have improper language or are reflecting hatred towards a group of people etc. Moreover, the software that powers Wikipedia provides certain tools allowing anyone to review changes made by others, drawing criticism from opposing groups, which in some cases lead to vulgar language and vandalism.

Editors can enforce these rules by deleting or modifying non-compliant material. However, it is more practical to have computer programs to perform anti-toxic duties, blocking or deleting entries of such properties.

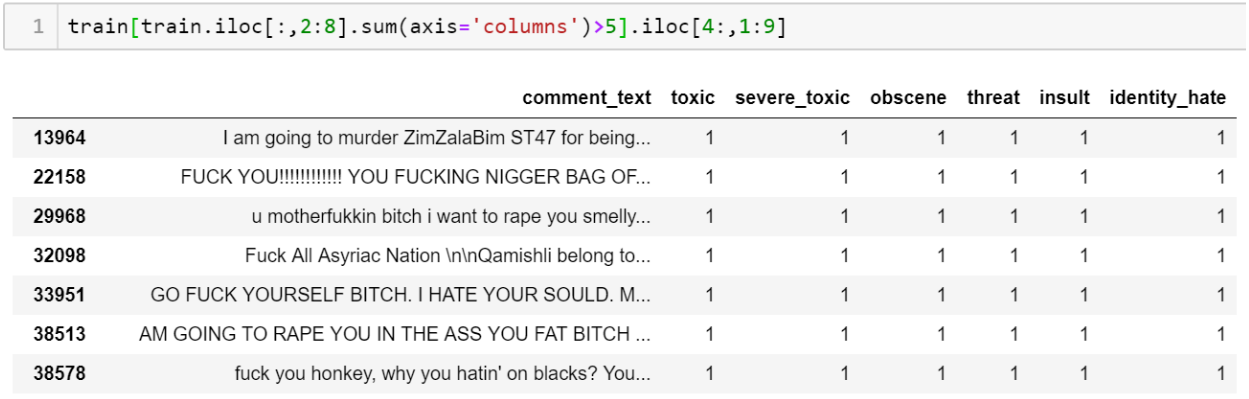
A model, utilizing neural networks (NN) , preferably Recursive NN, that's capable of detecting different types of toxicity like threats, obscenity, insults, and identity-based hate will be built and used to block improper comments.

Dataset:

Dataset consists of comments entered by authors word wide about various topics. In this project the aim is to build a multi-headed model that's capable of detecting different types of toxicity like threats, obscenity, insults, and identity-based hate. A successful model will hopefully help online discussion become more productive and respectful.

The number of observations in the dataset are more than 150 thousand, each including a document under column name 'comment_text', length of which ranges from just a couple of words to a few paragraph entry.

The nature of the label set, consisting of 6 columns brings challenge to this dataset because a comment may be both toxic and obscene, it may have elements of threat and identity hate - and even in some cases all six flags may be raised.



```
train[train.iloc[:,2:8].sum(axis='columns')>5].iloc[4:,1:9]
```

	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
13964	I am going to murder ZimZalaBim ST47 for being...	1	1	1	1	1	1
22158	FUCK YOU!!!!!!!!!!!! YOU FUCKING NIGGER BAG OF...	1	1	1	1	1	1
29968	u motherfukkin bitch i want to rape you smelly...	1	1	1	1	1	1
32098	Fuck All Asyriac Nation \n\nQamishli belong to...	1	1	1	1	1	1
33951	GO FUCK YOURSELF BITCH. I HATE YOUR SOULD. M...	1	1	1	1	1	1
38513	AM GOING TO RAPE YOU IN THE ASS YOU FAT BITCH ...	1	1	1	1	1	1
38578	fuck you honkey, why you hatin' on blacks? You...	1	1	1	1	1	1

This is a classification problem, but it comes with its slight variations since a comment can be labelled with more than one toxicity flag. Any specific comment may be represented with more than one label.

The aforementioned problem can be solved with different approaches, depending on the model used to solve the problem. For a baseline Multinomial Naive Bayes, model has to be run for each class label. However, on the other hand, with utilization of a neural network, output layer nodes can be adjusted to represent all class labels at once. Both methods were used for this study.

Data Exploration:

The 159’571 documents were joined into a corpus for analysis. The number of words in the corpus were 7’153’449 and number of unique words were 210’337.

Most common 50 words, after text preprocessing were:

```
[('article', 8850), ('page', 7312), ('wikipedia', 5399), ('edit', 4893), ('talk', 4702), ('use', 4317), ('like', 3591),
('think', 3022), ('know', 2995), ('source', 2930), ('good', 2810), ('time', 2404), ('add', 2316), ('people', 2236),
('user', 2234), ('want', 1937), ('block', 1911), ('need', 1891), ('image', 1823), ('find', 1746), ('delete', 1746),
('link', 1728), ('look', 1715), ('work', 1714), ('remove', 1711), ('thank', 1627), ('information', 1602), ('write',
1582), ('fuck', 1541), ('change', 1512), ('way', 1504), ('little', 1489), ('comment', 1481), ('editor', 1457),
('thing', 1446), ('section', 1441), ('list', 1415), ('point', 1409), ('deletion', 1406), ('fact', 1384), ('try', 1378),
('help', 1377), ('thanks', 1375), ('read', 1360), ('doe', 1342), ('question', 1336), ('new', 1317), ('mean',
1316), ('wp', 1312), ('right', 1253)]
```

Data Cleaning / Text Pre-processing:

Text pre-processing is a very important stage of data cleaning and the methods chosen, their order and detail level has significant impact on the quality of the classifier. Since the corpus consists of more than 7 millions of words, pre-processing is time consuming and sometimes computationally expensive.

First of all, URL addresses were removed by Python built-in library Regex. Then accented characters were normalized into ASCII characters, which really comes into play when dealing with non-English alphabets. This also prevents some of the flagged words to go unnoticed, since sometimes collaborators try to cloak swear words by making minor alterations, such as changing an “e” to “é”.

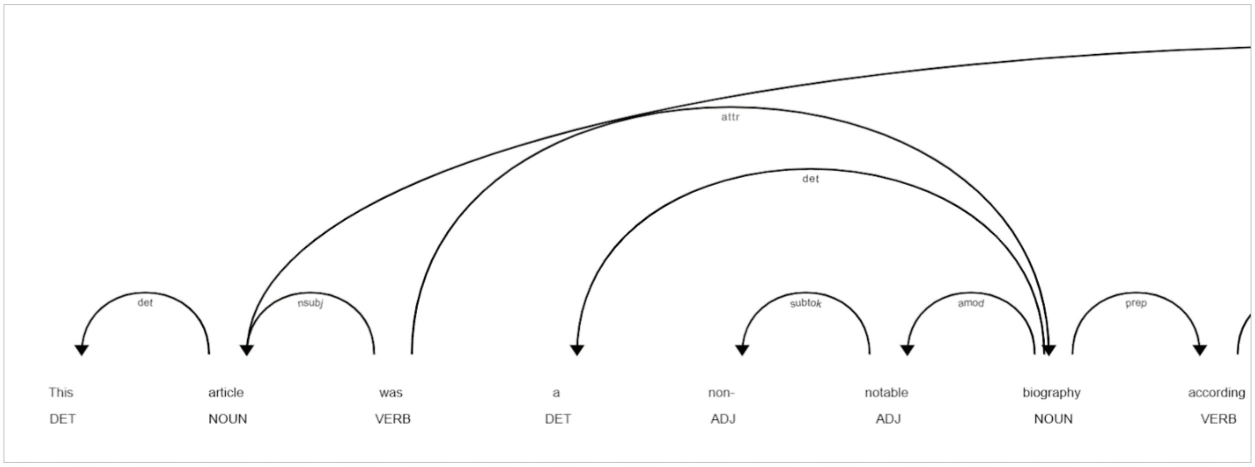
Next, contractions are expanded via a dictionary. Contractions are shortened versions of words or syllables. These shortened versions of existing words or phrases are created by removing specific letters and sounds. Examples would be, do not to don’t and I would to I’d. Converting each contraction to its expanded, original form often helps with text standardization.

Special characters including non-English alphabet and symbols were regarded as noise and removed with regular expressions code.

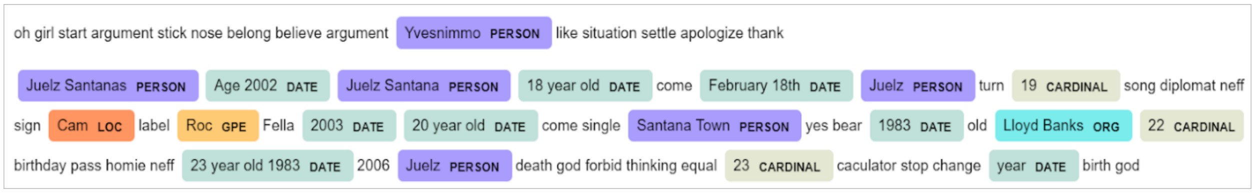
Next step in text preprocessing was lemmatization, where we remove word affixes to get to the base form of a word, which is always lexicographically correct. Lemmatization is computationally cumbersome compared to stemming, but it was preferred over stemming since it is more precise.

Later stop words were removed together with extra white spaces and text was converted to lowercase.

Spacy, a free open-source library for Natural Language Processing, was used in this project since it is one of the most sophisticated libraries and has great documentation. Spacy features NER, POS tagging, dependency parsing, word vectors.

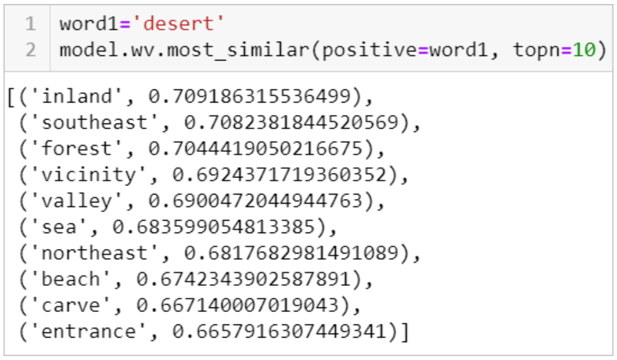


Two pre trained statistical English language models were used. The first model was ‘en_core_web_lg’, which supports NER and POS features, the other one was 'en_vectors_web_lg'. The latter model does not support NER or POS, but has 1070971 unique vectors (300 dimensions) trained with GloVe.



oh girl start argument stick nose belong believe argument Yvesnimmo PERSON like situation settle apologize thank
Juelz Santanas PERSON Age 2002 DATE Juelz Santana PERSON 18 year old DATE come February 18th DATE Juelz PERSON turn 19 CARDINAL song diplomat neff
sign Cam LOC label Roc GPE Fella 2003 DATE 20 year old DATE come single Santana Town PERSON yes bear 1983 DATE old Lloyd Banks ORG 22 CARDINAL
birthday pass homie neff 23 year old 1983 DATE 2006 Juelz PERSON death god forbid thinking equal 23 CARDINAL caculator stop change year DATE birth god

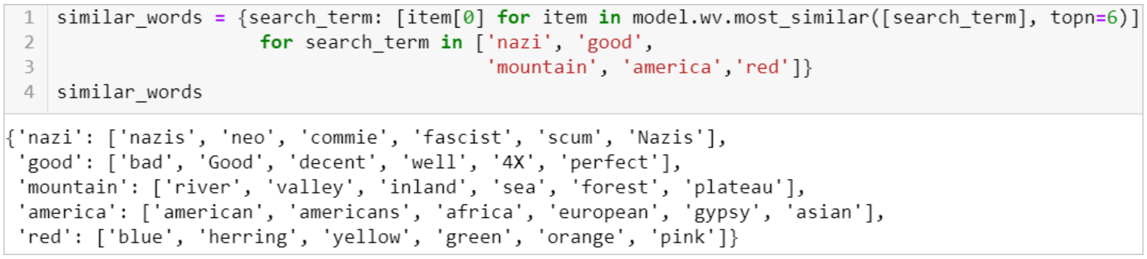
Non English comments:
This is an area of improvement since there are many comments that are not English. There are some libraries that can detect non-English words, however, this problem was not addressed at this time and left as provisional room for improvement. This also warrants a revisit to the filtering stage if the language uses a non-Latin alphabet, since such letters are filtered out.
Similarities:



```
word1='desert'
model.wv.most_similar(positive=word1, topn=10)
```

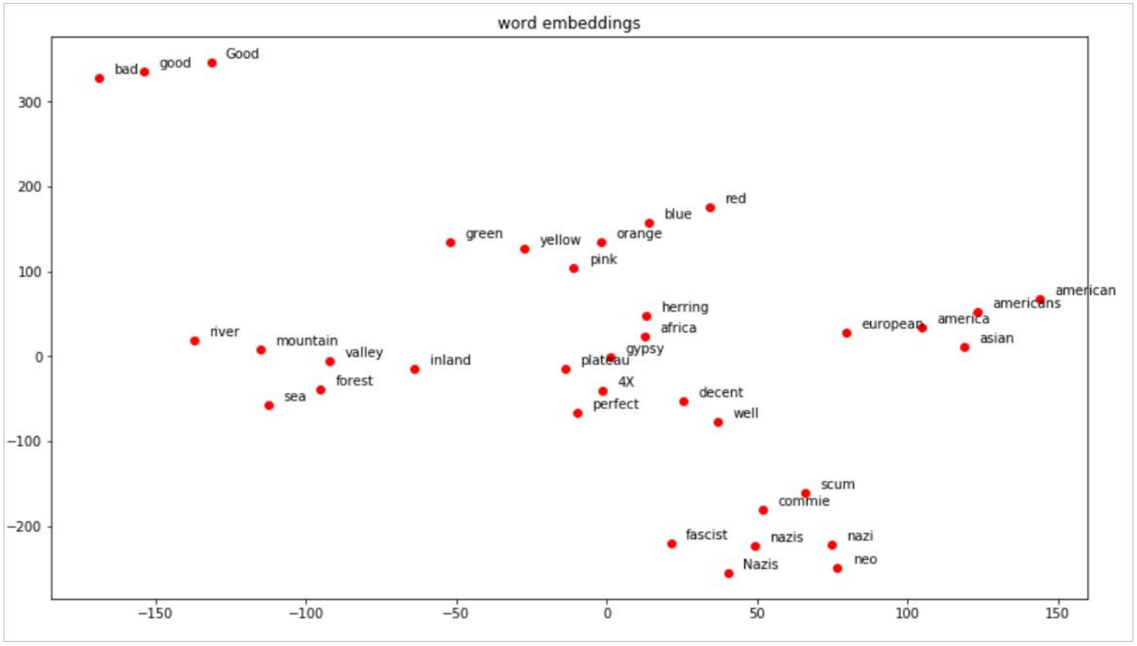
```
[('inland', 0.709186315536499),
 ('southeast', 0.7082381844520569),
 ('forest', 0.7044419050216675),
 ('vicinity', 0.6924371719360352),
 ('valley', 0.6900472044944763),
 ('sea', 0.683599054813385),
 ('northeast', 0.6817682981491089),
 ('beach', 0.6742343902587891),
 ('carve', 0.667140007019043),
 ('entrance', 0.6657916307449341)]
```

The pre-trained vectors of spacy did not work well for similarities since language of comment texts is not daily used language. After training own word2Vec vectors with Gensim, the similarity results were quite good. On the most similar words to selected word can be seen. For visualization of similarities, dimension reduction is required. T-distributed Stochastic Neighbor Embedding (t-SNE) algorithm for visualization was used for embedding high-dimensional data for visualization in a low-dimensional space of two dimensions. t-SNE models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.



```
similar_words = {search_term: [item[0] for item in model.wv.most_similar([search_term], topn=6)]
                    for search_term in ['nazi', 'good',
                                         'mountain', 'america','red']}
similar_words
```

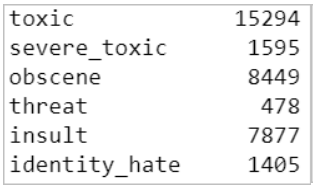
```
{'nazi': ['nazis', 'neo', 'commie', 'fascist', 'scum', 'Nazis'],
 'good': ['bad', 'Good', 'decent', 'well', '4X', 'perfect'],
 'mountain': ['river', 'valley', 'inland', 'sea', 'forest', 'plateau'],
 'america': ['american', 'americans', 'africa', 'european', 'gypsy', 'asian'],
 'red': ['blue', 'herring', 'yellow', 'green', 'orange', 'pink']}
```



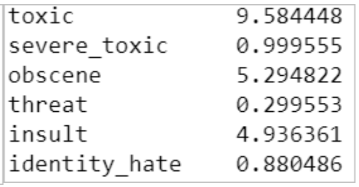
Baseline Models:

After text pre-processing; Multinomial Naive Bayes, Logistic regression and Support Vector Classification models were used to classify each comment for each of the six classes.

The biggest problem for approaching a dataset like this is class imbalance, which gives incorrect results. Accuracy terms higher than 0.995 were achieved, however, when we look at precision and/or recall we can see that actual results are not promising.



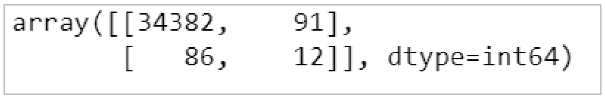
```
toxic            15294
severe_toxic      1595
obscene           8449
threat             478
insult            7877
identity_hate     1405
```



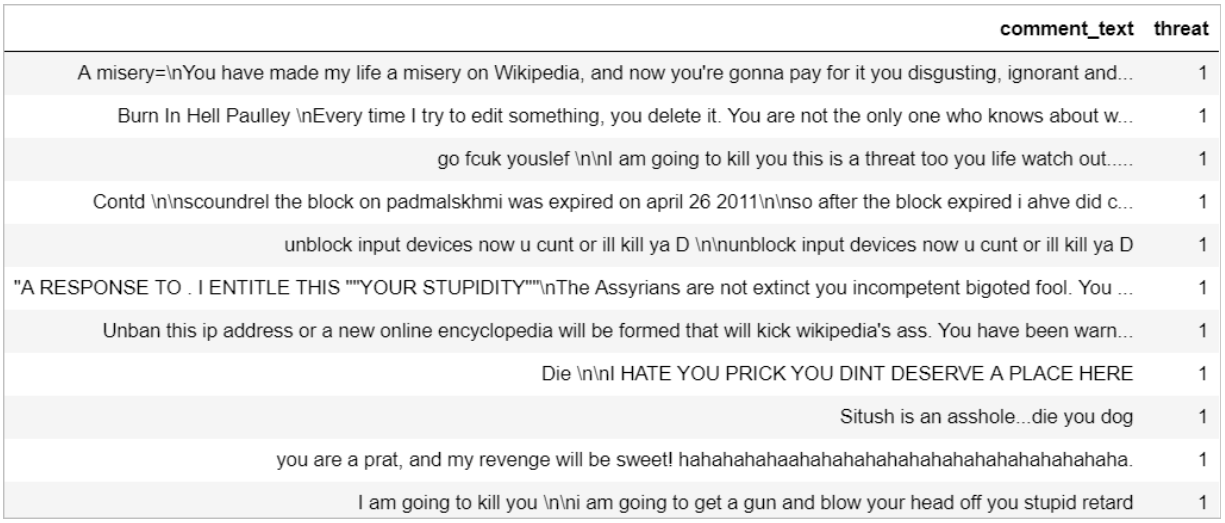
```
toxic            9.584448
severe_toxic     0.999555
obscene          5.294822
threat           0.299553
insult           4.936361
identity_hate    0.880486
```

Distribution of classes as depicted on charts on the left, are for the dataset size of 159 thousand. On the far left are number of “True” values for each feature, and to its right are the relating percentages of each variable. Weight imbalance was reduced by utilizing class weight and sample weight features of scikit learn.

Each model must be run 6 times, one for each class label. The validation accuracy score, 0.9948, looks interestingly high for “threat” label, but when we look at the confusion matrix we can see that the score is not what it appears like. The confusion matrix below clearly shows the “inaccuracy” of the accuracy calculation, and points out why other score functions such as precision, recall and auc should be used. Below are false negative results for threat.

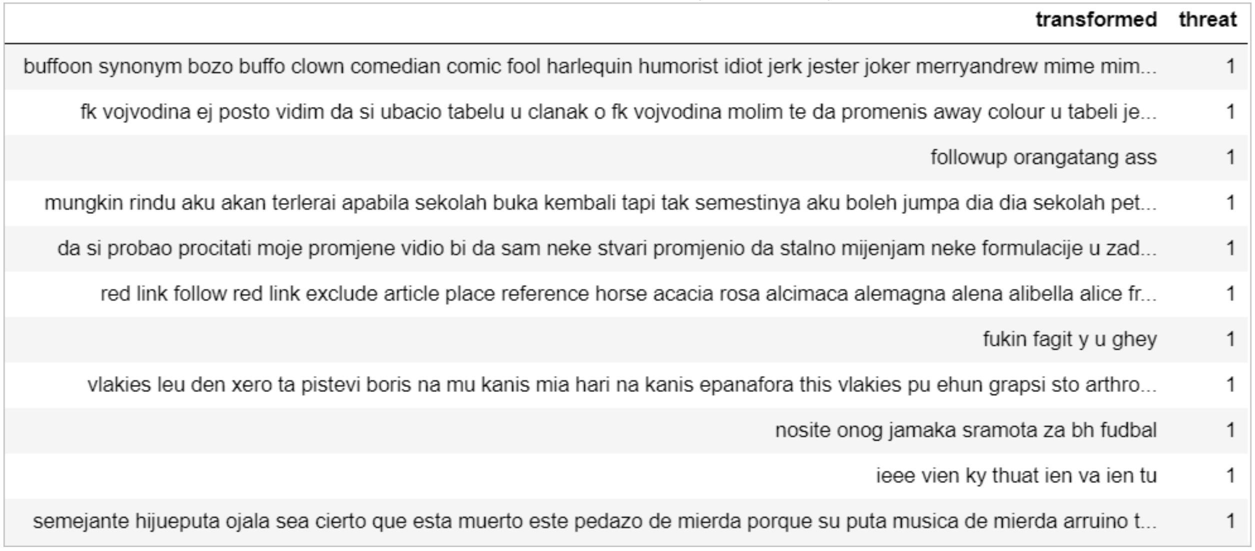


```
array([[34382,    91],
       [   86,    12]], dtype=int64)
```



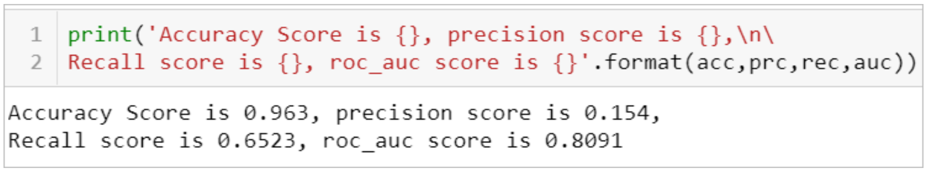
comment_text	threat
A misery=\nYou have made my life a misery on Wikipedia, and now you're gonna pay for it you disgusting, ignorant and...	1
Burn In Hell Paulley \nEvery time I try to edit something, you delete it. You are not the only one who knows about w...	1
go fcuk youslef \n\nI am going to kill you this is a threat too you life watch out.....	1
Contd \n\nscoundrel the block on padmalskhmi was expired on april 26 2011\n\nso after the block expired i ahve did c...	1
unblock input devices now u cunt or ill kill ya D \n\nunblock input devices now u cunt or ill kill ya D	1
"A RESPONSE TO . I ENTITLE THIS ""YOUR STUPIDITY""\nThe Assyrians are not extinct you incompetent bigoted fool. You ...	1
Unban this ip address or a new online encyclopedia will be formed that will kick wikipedia's ass. You have been warn...	1
Die \n\nI HATE YOU PRICK YOU DINT DESERVE A PLACE HERE	1
Sitush is an asshole...die you dog	1
you are a prat, and my revenge will be sweet! hahahahahaahahahahahahahahahahahahahaha.	1
I am going to kill you \n\ni am going to get a gun and blow your head off you stupid retard	1

Out of 98 threats of a 34571 validation comments, only 12 were picked out and labelled as threat.



transformed	threat
buffoon synonym bozo buffo clown comedian comic fool harlequin humorist idiot jerk jester joker merryandrew mime mim...	1
fk vojvodina ej posto vidim da si ubacio tabelu u clanak o fk vojvodina molim te da promenis away colour u tabeli je...	1
followup orangatang ass	1
mungkin rindu aku akan terlerai apabila sekolah buka kembali tapi tak semestinya aku boleh jumpa dia dia sekolah pet...	1
da si probao procitati moje promjene vidio bi da sam neke stvari promjenio da stalno mijenjam neke formulacije u zad...	1
red link follow red link exclude article place reference horse acacia rosa alcimaca alemagna alena alibella alice fr...	1
fukin fagit y u ghey	1
vlakies leu den xero ta pistevi boris na mu kanis mia hari na kanis epanafora this vlakies pu ehun grapsi sto arthro...	1
nosite onog jamaka sramota za bh fudbal	1
ieee vien ky thuat ien va ien tu	1
semejante hijueputa ojala sea cierto que esta muerto este pedazo de mierda porque su puta musica de mierda arruino t...	1

Above are the test dataset results. The text predicted as a member of threat class, but the contents are not really posing a threat. Please note that especially foreign languages have been marked as a threat, but almost all are false positives. The results of MNB classifier after weight balancing for “identity hate” feature is shown as below.

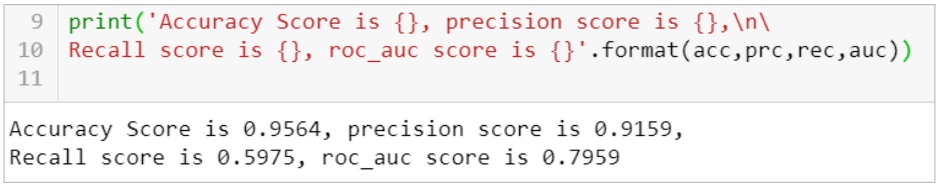


```
print('Accuracy Score is {}, precision score is {},\n\
Recall score is {}, roc_auc score is {}'.format(acc,prc,rec,auc))
```

```
Accuracy Score is 0.963, precision score is 0.154,
Recall score is 0.6523, roc_auc score is 0.8091
```

Logistic Regression:

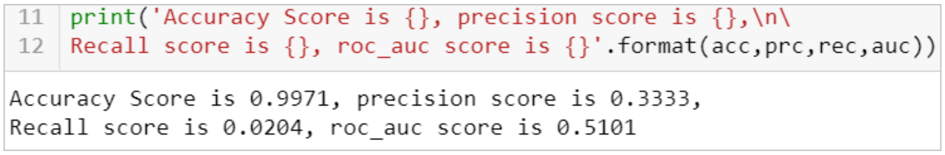
The results of Logistic Regression model for “toxic” and “threat” are shown below respectively. Please note that the model is good at explaining toxic feature, but not the threat.



```
print('Accuracy Score is {}, precision score is {},\n\
Recall score is {}, roc_auc score is {}'.format(acc,prc,rec,auc))

```

```
Accuracy Score is 0.9564, precision score is 0.9159,
Recall score is 0.5975, roc_auc score is 0.7959
```



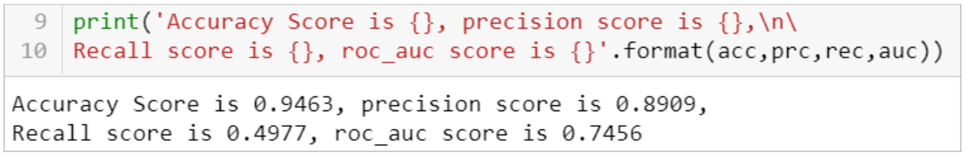
```
print('Accuracy Score is {}, precision score is {},\n\
Recall score is {}, roc_auc score is {}'.format(acc,prc,rec,auc))
```

```
Accuracy Score is 0.9971, precision score is 0.3333,
Recall score is 0.0204, roc_auc score is 0.5101
```

Support Vector Classification:

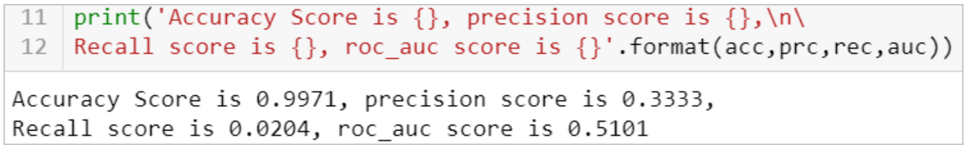
Final baseline model for classification used in this project is SVC. In order to reduce time required, number of features of TF-IDF were reduced using Singular Value Decomposition.

Again, the results of SVC model for “toxic” and “threat” are shown below respectively and the model acts similar to logistic regression in that it is good at explaining toxic feature, but not the threat.



```
print('Accuracy Score is {}, precision score is {},\n\
Recall score is {}, roc_auc score is {}'.format(acc,prc,rec,auc))
```

```
Accuracy Score is 0.9463, precision score is 0.8909,
Recall score is 0.4977, roc_auc score is 0.7456
```



```
print('Accuracy Score is {}, precision score is {},\n\
Recall score is {}, roc_auc score is {}'.format(acc,prc,rec,auc))
```

```
Accuracy Score is 0.9971, precision score is 0.3333,
Recall score is 0.0204, roc_auc score is 0.5101
```

Baseline Models Conclusion:

A 150 thousand long dataset comprising of comments entered about various topics has been preprocessed with a goal of building a multi-headed model that’s capable of detecting different types of toxicity like threats, obscenity, insults, and identity-based hate. After intensive text pre-processing, and compensating for heavy class imbalance, various baseline models were employed in attempt to correctly label each feature. While models successfully explained “toxic, severe toxic, obscene”, they failed to generate promising results for variables such as “threat”.

In the following chapter, recurrent neuro-network models will be used to fill this gap and provide a better solution to the problem.

Neural Network Models

In the previous chapters of this project, various models including Naïve-Bayes, Logistic regression, SVC were utilized to reach a solution. These models proved effective for some of the label sets, however, especially for threat and identity hate columns, they weren't successful.

This is because bag of words model or n-gram would not suffice. In order to be able to label a comment text as an identity hate, it must have a nation, a religious group, a sect etc. as the subject in the sentence and in an adjective clause. In other words, having a toxic word and a POS with a specific group in a bag of words is not enough to label as identity hate: the toxicity must be directed towards that specific entity or group of people.

This is also true for the threat label. Simply put, having "kill" and "you" in a sentence alone does not constitute a threat. "Kill" must be the verb in the clause and "you" must be the subject.

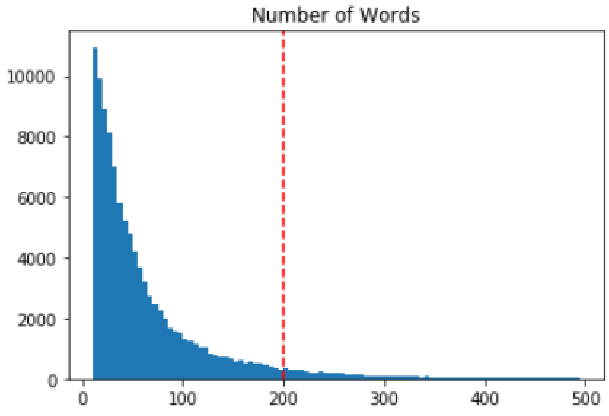
Text Preprocessing:

Keras provides a text-preprocessing library. For this study initially only tokenizer and padding features were utilized. Later on, the steps used previously for non-neural networks were deployed to provide a benchmark between two disciplines. The results with pre-processed text were better.

Text preparation included removal of URL addresses by Regex and normalization of accented characters. Next contractions were expanded via a dictionary for text standardization.

Special characters including non-English alphabet and symbols were regarded as noise and removed with regular expressions code.

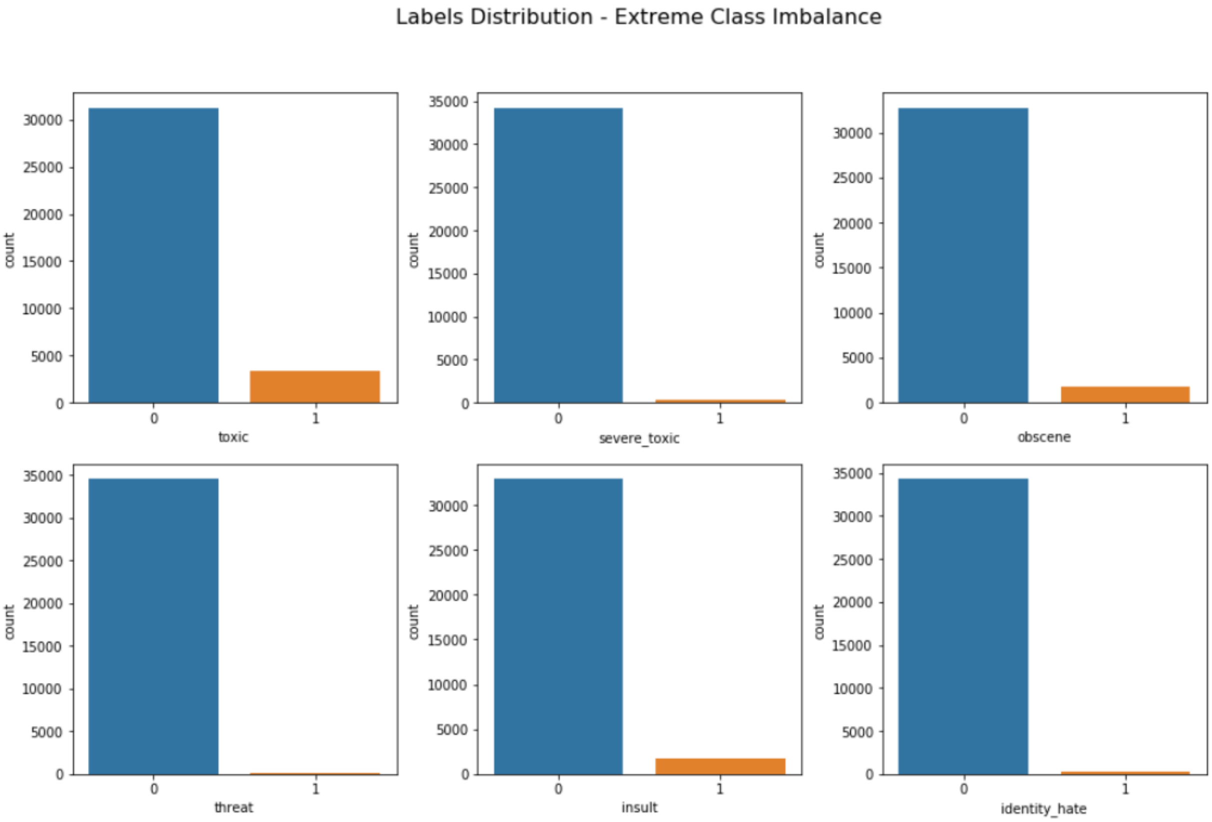
Next step in text preprocessing was lemmatization, where word affixes were removed to get to the base form of a word, which is always lexicographically correct. However, this time, stop words were not removed since this would also cause loss of subject pronouns, object pronouns and possessive adjectives. This is vital for the model to resolve identity hate and threat labels since it adds richness to the text and picks up the nuance.



After these steps, Keras text preprocessing tools were used extensively to tokenize and transform preprocessed comment texts into a sequence of integers. The histogram below shows the distribution of the number of words used for each comment. When creating sequences with Keras preprocessing, therefore, sequence length of 200 was selected, and comments having less tokens were padded.

The dataset EDA with count plot method reveals extreme imbalance in each label distribution such that a model predicting "zeroes only" would have achieved higher than %99.7 accuracy for threat.

This is regarded as "Accuracy Paradox for Predictive Analytics" in that, Predictive Models with a given level of Accuracy may have greater Predictive Power than Models with higher Accuracy. In the same prospect, one would easily fall into the lure of tracking a better ROC AUC score through the models, which would provide meaningless faulty results. Since we don't have roughly equal numbers of observations for each class we cannot rely on ROC curves, which would provide an optimistic picture of the model on datasets with a class imbalance.



Precision-Recall (PR) curves, have been cited as an alternative to ROC curves for tasks with a large skew in the class distribution, since they can provide an accurate prediction of future classification performance due to the fact that they evaluate the fraction of true positives among positive predictions.

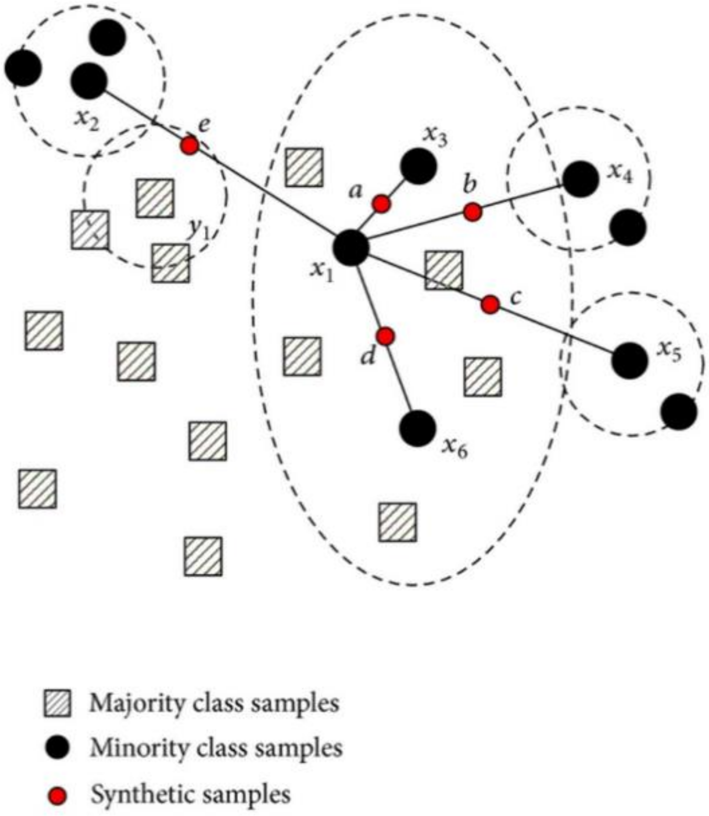
While it is important to decide which metric to use before building a model for performance metering, it is imperative to remedy the class imbalance. Techniques used for this project were synthetic sampling together with class weights.

The use of under sampling is not recommended since it will cause data loss, therefore RandomOverSampler, SMOTE (Synthetic Minority Over-sampling Technique) and ADASYN (Adaptive Synthetic Sampling Approach for Imbalanced Learning) were used to generate synthetic data. SMOTE finds the n-nearest neighbors in the minority class for each of the samples in the class and interpolates between the neighbors and generates random points on the lines. ADASYN is an improved version of SMOTE which adds random small values to the points, making them more realistic and scattered.

Therefore, ADASYN can adaptively generate synthetic data samples for the minority class to reduce the bias introduced by the imbalanced data distribution. The necessary modules are readily available from imbalanced learn and class weight was imported from Sci-kit learn library.

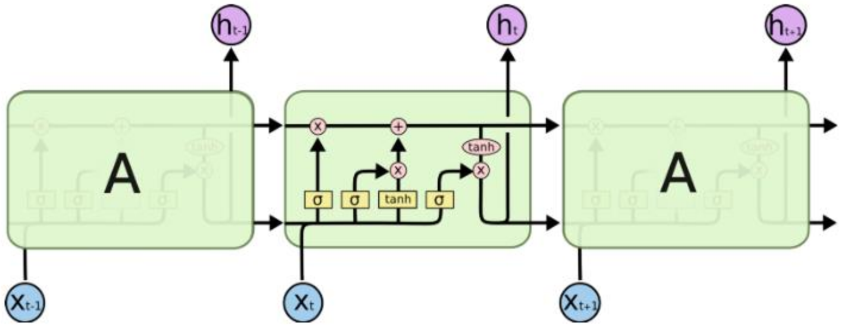
There are some caveats about synthetic sampling. Once the oversampling threshold is exceeded, the model becomes highly prone to overfitting. On the other hand, if enough oversampling is not used, it underperforms, in most cases returns only zeroes. Several models had to be trained to find exact parameters for each oversampling technique.

Another drawback with imbalanced learn synthetic oversampling library is that it does not support multi label classes, rendering it impossible to oversample the whole label set. For this project, label sets were individually fit on over samplers.



Modeling

After preparing the dataset, various models were built and their performances were compared including some feed forward models but mostly Recurrent Neural Networks. Despite computational burden and lack of a hyperparameter tuning and a perfect grid search optimization tool, LSTMs, GRUs, Bidirectional LSTMs, Bidirectional GRUs, Convolutional Neural Networks, and Bidirectional LSTM with Attention models with a broad range of options and parameters have been tested. Feed forward models comprising of dense layers converged easily and produced high ROC AUC scores, however they returned zeroes for entire “threat” and “identity hate” columns. Text classification task is generally considered to be difficult, the main reason for this is long term dependencies found in a text.

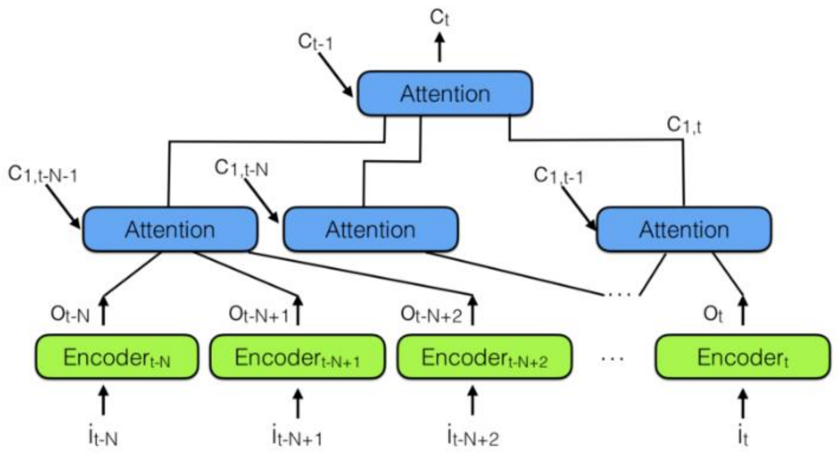


LSTM, from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Unlike feed-forward neural networks activation outputs from neurons propagate in both directions in Recurrent Neural Networks creating, a loop in the architecture which acts as a “memory state” of the neurons. This state allows the neurons an ability to remember what have been learned. However, plain recursive neural networks do not have gates, thus cannot learn and tune the parameters of the earlier layers. LSTMs, networks that have been designed to remember or forget the leanings, have been designed to address this problem.

As a value adder, bidirectional LSTMs improve model skill over regular LSTMs by stacking outputs of LSTM layer applied forward and backward on input vectors.

However, even though LSTM cells are designed to capture long terms dependencies, they are prone to performance decay as the sequence length exceeds about 30. With the sequence length of 200, introduction of an additional layer, "Attention with context" was necessary. Attention, as the name suggests, provides a mechanism where output can 'attend to' (focus on) certain input time step for an input sequence of arbitrary length. A better way to look into the past is to use attention modules to summarize all past encoded vectors into a context vector Ct.



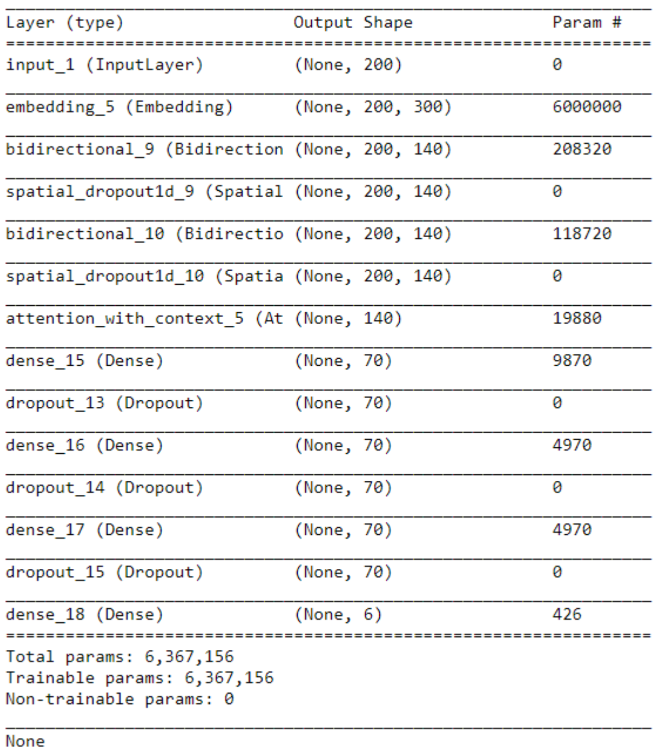
Attention with context, from: https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0

In the hierarchical neural attention encoder, multiple layers of attention can look at a small portion of recent past, about 100 vectors, while layers above can look at 100 of these attention modules, effectively integrating the information of 100 x 100 vectors. This extends the ability of the hierarchical neural attention encoder to 10,000 past vectors.

Another feature that is worth mentioning is the activation. The model had a tendency to lead to dead nodes, therefore "Leaky ReLu" was used, which proved to be more effective.

Layers

Layer 1 - The Embedding layer is defined as the first hidden layer of a network. Rather than using a traditional bag-of-words model encoding scheme with large sparse vectors, for this project, word embeddings, thus dense vector representations were used. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. The embedding layer used was defined with a vocabulary of 20'000 a vector space of 300 dimensions in which words will be embedded, and input documents that have 200 words each (sequences). Maximum lengths are suitably chosen from the distribution of sentence length of the dataset used, and it would vary on a different dataset.



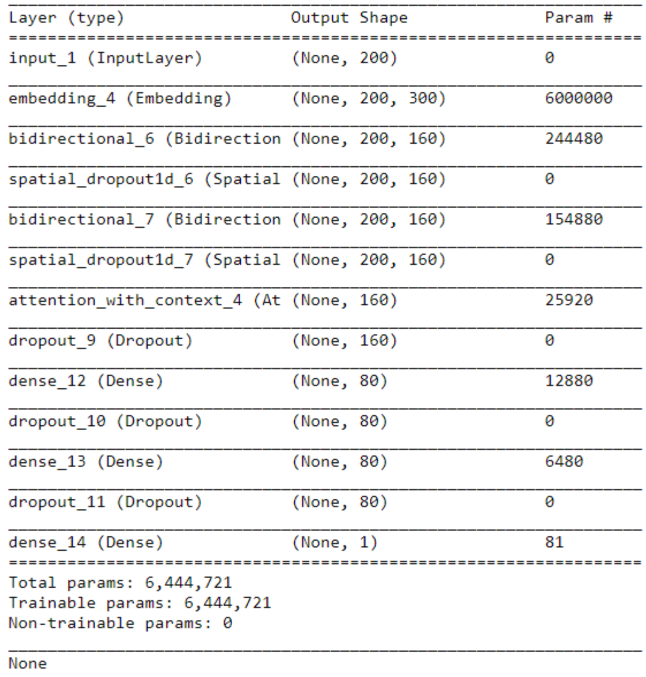
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         (None, 200)               0         
_________________________________________________________________
embedding_5 (Embedding)      (None, 200, 300)          6000000   
_________________________________________________________________
bidirectional_9 (Bidirection (None, 200, 140)          208320    
_________________________________________________________________
spatial_dropout1d_9 (Spatial (None, 200, 140)          0         
_________________________________________________________________
bidirectional_10 (Bidirectio (None, 200, 140)          118720    
_________________________________________________________________
spatial_dropout1d_10 (Spatia (None, 200, 140)          0         
_________________________________________________________________
attention_with_context_5 (At (None, 140)               19880     
_________________________________________________________________
dense_15 (Dense)             (None, 70)                9870      
_________________________________________________________________
dropout_13 (Dropout)         (None, 70)                0         
_________________________________________________________________
dense_16 (Dense)             (None, 70)                4970      
_________________________________________________________________
dropout_14 (Dropout)         (None, 70)                0         
_________________________________________________________________
dense_17 (Dense)             (None, 70)                4970      
_________________________________________________________________
dropout_15 (Dropout)         (None, 70)                0         
_________________________________________________________________
dense_18 (Dense)             (None, 6)                 426       
=================================================================
Total params: 6,367,156
Trainable params: 6,367,156
Non-trainable params: 0
_________________________________________________________________
None
```

Layer 2/3 - is a Bidirectional LSTM layer on both sides which outputs vectors of length 100. Bidirectional GRU was also tried, but LSTM was working better. Due to the computational weight of the LSTM layer, for this project CudNNLSTM implementation was used. This is a fast LSTM implementation with CuDNN, and it can only be run on GPU, with the TensorFlow backend. While it lacks the vital arguments such as dropout and recurrent dropout, this layer was preferred due to its speed during hyperparameter tuning. Although unconventional, spatial dropout was preferred over conventional dropout, as it proved to be more effective.

Layer 4/5 - is another Bidirectional LSTM layer featuring similar traits as of the previous counterpart.

Layer 6 - is the “Attentionwithcontext” layer.

Remaining Layers – The rest of the layers are dense layers with diminishing dropout percentages. Please note that the output layer has 6 nodes, 1 representing each label. At this stage only class weights were incorporated. The next step will be to oversample underperforming individual label columns and concatenate them to this master model.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         (None, 200)               0         
_________________________________________________________________
embedding_4 (Embedding)      (None, 200, 300)          6000000   
_________________________________________________________________
bidirectional_6 (Bidirection (None, 200, 160)          244480    
_________________________________________________________________
spatial_dropout1d_6 (Spatial (None, 200, 160)          0         
_________________________________________________________________
bidirectional_7 (Bidirection (None, 200, 160)          154880    
_________________________________________________________________
spatial_dropout1d_7 (Spatial (None, 200, 160)          0         
_________________________________________________________________
attention_with_context_4 (At (None, 160)               25920     
_________________________________________________________________
dropout_9 (Dropout)          (None, 160)               0         
_________________________________________________________________
dense_12 (Dense)             (None, 80)                12880     
_________________________________________________________________
dropout_10 (Dropout)         (None, 80)                0         
_________________________________________________________________
dense_13 (Dense)             (None, 80)                6480      
_________________________________________________________________
dropout_11 (Dropout)         (None, 80)                0         
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 81        
=================================================================
Total params: 6,444,721
Trainable params: 6,444,721
Non-trainable params: 0
_________________________________________________________________
None
```

Threat Model:

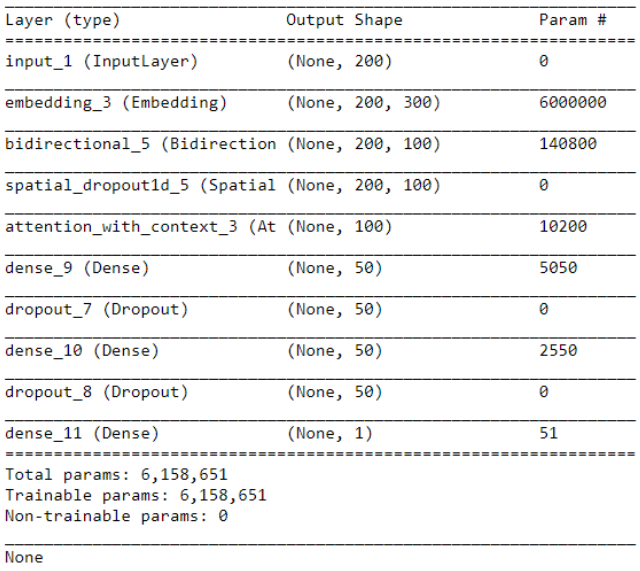
Due to the heavy imbalance, this class had to be oversampled, and a separate network was built on the oversampled dataset.

This time, the model had less hidden layers and the number of nodes on each layer has a been increased as well as the batch size, where an 8 fold increase produced better results.

Again attention with context layer was used.

The predictions results were concatenated to the master model, and significant improvement was achieved through oversampling.

Identity Hate Model:



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         (None, 200)               0         
_________________________________________________________________
embedding_3 (Embedding)      (None, 200, 300)          6000000   
_________________________________________________________________
bidirectional_5 (Bidirection (None, 200, 100)          140800    
_________________________________________________________________
spatial_dropout1d_5 (Spatial (None, 200, 100)          0         
_________________________________________________________________
attention_with_context_3 (At (None, 100)               10200     
_________________________________________________________________
dense_9 (Dense)              (None, 50)                5050      
_________________________________________________________________
dropout_7 (Dropout)          (None, 50)                0         
_________________________________________________________________
dense_10 (Dense)             (None, 50)                2550      
_________________________________________________________________
dropout_8 (Dropout)          (None, 50)                0         
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 51        
=================================================================
Total params: 6,158,651
Trainable params: 6,158,651
Non-trainable params: 0
_________________________________________________________________
None
```

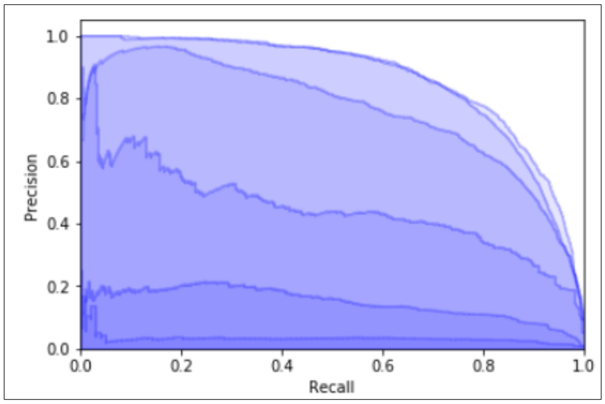
Another problematic target column was identity hate, and the master model had difficulty to learn and predict the correct labels due to extreme imbalance. Once again ADASYN technique was used to oversample together with class weights. This time a simpler model with less layers and nodes was used. Bidirectional LSTM, together with an Attention with context layer was used.

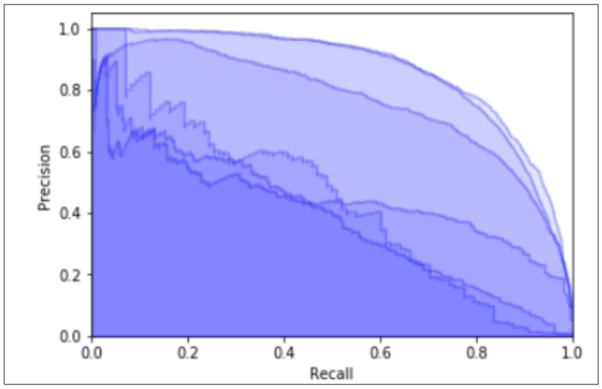
A smaller batch size was observed to produce better results.

Again, the prediction results were concatenated to the master model, and significant improvement was achieved through oversampling.

Performance

Datasets with excessive imbalance need to be processed and evaluated carefully. As previously stated above, using ROC AUC as a metric would be quite misleading and euphoric due to base accuracy of the classes. Therefore, for this project, main focus was on precision recall curve and the area pertaining to it. Since this dataset is multi-label, metrics were calculated for each class and weighted average was used to increase precision.

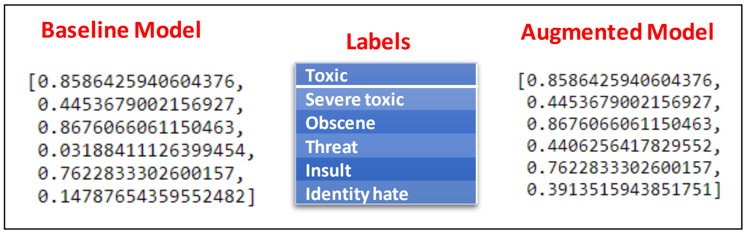




The master model performed good for “toxic”, “insult” and “obscene” however did underperform on “threat” and “identity hate”. The nature of this discrepancy was explained previously and to remedy this problem, ADASYN was introduced, and individual models were built for problematic labels.

The results showed significant improvement, as can be seen on the precision / recall graphs on the left side, with the baseline model appearing on top and augmented model appearing below.

The mean of the area that falls under these curves increased a significantly by %20. The dramatic increase of precision-recall score for the identity and threat labels can be observed below.



Baseline Model

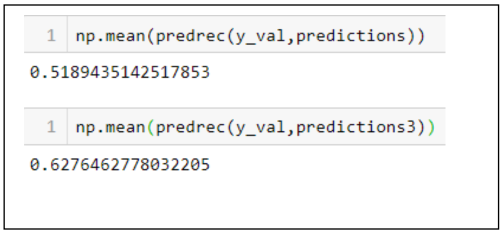
```
[0.8586425940604376,
 0.4453679002156927,
 0.8676066061150463,
 0.03188411126399454,
 0.7622833302600157,
 0.14787654359552482]
```

Labels

Toxic
Severe toxic
Obscene
Threat
Insult
Identity hate

Augmented Model

```
[0.8586425940604376,
 0.4453679002156927,
 0.8676066061150463,
 0.4406256417829552,
 0.7622833302600157,
 0.3913515943851751]
```



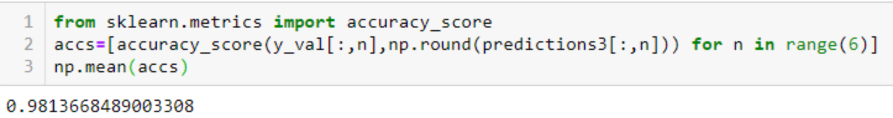
```
np.mean(predrec(y_val,predictions))
```

```
0.5189435142517853
```

```
np.mean(predrec(y_val,predictions3))
```

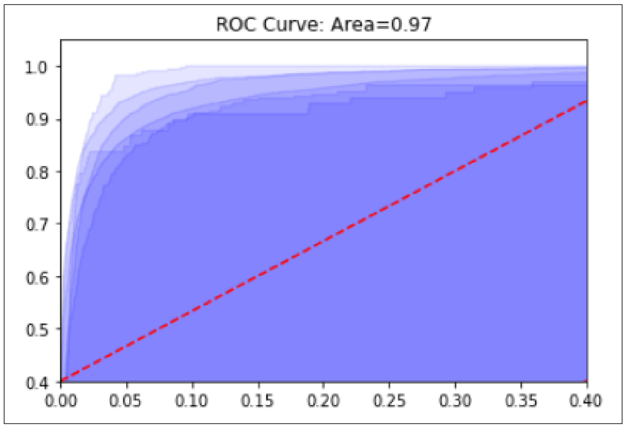
```
0.6276462778032205
```

For imbalanced datasets accuracy is another metric to be used with caution. The accuracy for the trained model for this project was 0.98.



```
from sklearn.metrics import accuracy_score
accs=[accuracy_score(y_val[:,n],np.round(predictions3[:,n])) for n in range(6)]
np.mean(accs)
```

```
0.9813668489003308
```



ROC AUC score and ROC curves should be ignored, however, it is worthy to note that, augmenting the model with oversampled data did not have an impact on the ROC AUC score, while it increased the precision recall.

ROC curves of the all six label sets can be observed on the left.

ROC AUC score, with weighted average was a little bit better than 0.97.

Conclusion, Problem Areas, and Future Work

The project aimed to correctly identify and label comment texts written by internet users on Wikipedia. The dataset is multi-label, and highly imbalanced, and many times mislabeled. Initially, baseline models such as Naïve-Bayes, Logistic Regression, SVC were utilized, however these models did not perform well and failed to learn dependencies in explaining label sets “threat” and “identity hate”.

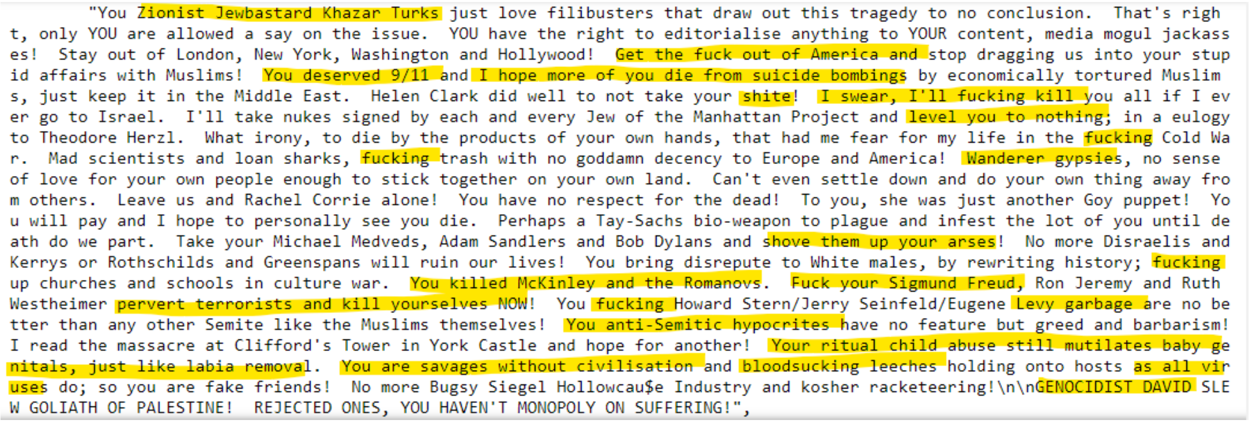
Following baseline models, various neural networks were trained after Keras text-preprocessing. At this time no other preprocessing methods were used, however, in later stages of the project, Spacy based text processing including lemmatization was made, with and without removing the stop words. Of all three instances, best performance was achieved with Spacy preprocessing without removing the stop words, followed by Keras text-preprocessing.

Due to extreme class imbalance, both class weights and synthetic oversampling methods were used when building single label models. Of all oversampling techniques, ADASYN performed the best, and thus all singular models were built with ADASYN. Over sampler was trained after padded sequences were created and was fit on model for each single label column that had high class imbalance.

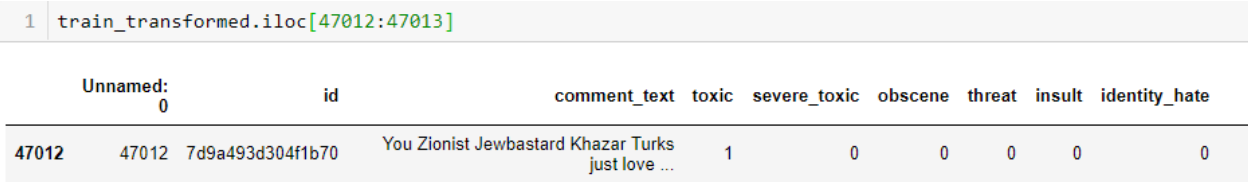
Neural networks evaluated for this project include feed forward dense, CNN, LSTM, GRU, BiDirectional LSTM and BiDirectional GRUs. Best results were achieved with dual BiDirectional LSTMs followed by an

Attention with context layer. Despite lack of some important arguments, CudNNLSTM, a GPU-accelerated version for deep neural networks was used since it provided highly tuned implementations for standard routines, allowing models to run many times faster.

The biggest problem ran into during this project was mislabeled instances. The model can only work effectively if the labels it is introduced are precise, and would produce erroneous results when fed with arbitrary labels. Unfortunately, during EDA analysis of this dataset, many comment texts were found to be mislabeled. Below is a single comment text. Please see the original row and its corresponding labels as presented below. This comment is extremely vulgar and inappropriate, however, when being labeled, it was only regarded to be toxic.



```
        "You Zionist Jewbastard Khazar Turks just love filibusters that draw out this tragedy to no conclusion.  That's righ
t, only YOU are allowed a say on the issue.  YOU have the right to editorialise anything to YOUR content, media mogul jackass
es!  Stay out of London, New York, Washington and Hollywood!  Get the fuck out of America and stop dragging us into your stup
id affairs with Muslims!  You deserved 9/11 and I hope more of you die from suicide bombings by economically tortured Muslim
s, just keep it in the Middle East.  Helen Clark did well to not take your shite!  I swear, I'll fucking kill you all if I ev
er go to Israel.  I'll take nukes signed by each and every Jew of the Manhattan Project and level you to nothing; in a eulogy
to Theodore Herzl.  What irony, to die by the products of your own hands, that had me fear for my life in the fucking Cold Wa
r.  Mad scientists and loan sharks, fucking trash with no goddamn decency to Europe and America!  Wanderer gypsies, no sense
of love for your own people enough to stick together on your own land.  Can't even settle down and do your own thing away fro
m others.  Leave us and Rachel Corrie alone!  You have no respect for the dead!  To you, she was just another Goy puppet!  Yo
u will pay and I hope to personally see you die.  Perhaps a Tay-Sachs bio-weapon to plague and infest the lot of you until de
ath do we part.  Take your Michael Medveds, Adam Sandlers and Bob Dylans and shove them up your arses!  No more Disraelis and
Kerrys or Rothschilds and Greenspans will ruin our lives!  You bring disrepute to White males, by rewriting history; fucking
up churches and schools in culture war.  You killed McKinley and the Romanovs.  Fuck your Sigmund Freud, Ron Jeremy and Ruth
Westheimer pervert terrorists and kill yourselves NOW!  You fucking Howard Stern/Jerry Seinfeld/Eugene Levy garbage are no be
tter than any other Semite like the Muslims themselves!  You anti-Semitic hypocrites have no feature but greed and barbarism!
I read the massacre at Clifford's Tower in York Castle and hope for another!  Your ritual child abuse still mutilates baby ge
nitals, just like labia removal.  You are savages without civilisation and bloodsucking leeches holding onto hosts as all vir
uses do; so you are fake friends!  No more Bugsy Siegel Hollowcau$e Industry and kosher racketeering!\n\nGENOCIDIST DAVID SLE
W GOLIATH OF PALESTINE!  REJECTED ONES, YOU HAVEN'T MONOPOLY ON SUFFERING!",
```



```
train_transformed.iloc[47012:47013]
```

	Unnamed: 0	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
47012	47012	7d9a493d304f1b70	You Zionist Jewbastard Khazar Turks just love ...	1	0	0	0	0	0

It is quite open that this comment has all the elements of toxic, severe toxic, obscene, threat, insult and identity hate, and it is as profane as it can be. However, as witnessed above, it was flagged as toxic only. This is one of the many underrated comments that were run into when working on this project, and it is impossible to assess the ratio or the impact on the model before reading all comments.

Due to the imbalance in the dataset, ROC AUC score would produce meaningless results, and thus was not preferred. Instead, the model performance was measured by precision recall curve.

The model produced good results with BiDirectional LSTMs and Attention layer, and showed significant improvement with introduction of oversampling. Various activations, number of hidden layers, dropout percentages, batch sizes were tried to achieve these results, however there could still be room for improvement on the hyperparameter tuning.

For future work, planned area of improvement is using word embeddings to build a synonyms dictionary for data augmentation, which would be used to alleviate class imbalance.

Other are of planned future work includes using Bidirectional Encoder Representations from Transformers, or BERT, which will be handled as a separate project.