
ML Course Project Yahoo Troll

— Shreyansh Rai & Saket Gurjar —

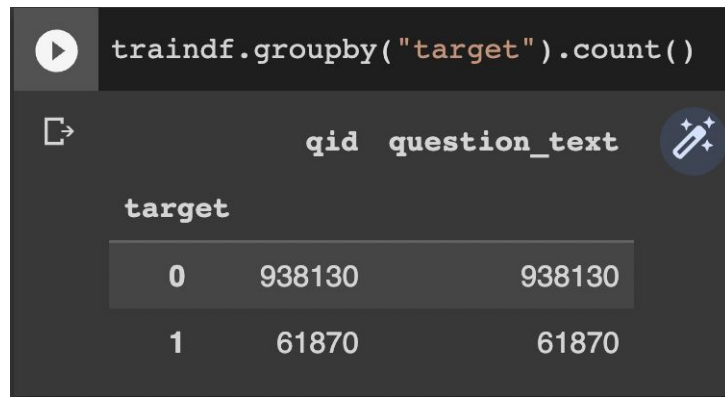
NLP : Basics covered and what was learned -

- Basics of NLP including earlier models based on probability to predict upcoming words, and traditional n-gram probability models.
- Towards Data Science and Medium articles for learning the basics preprocessing and cleaning that is required before you can use the data and apply a model to it.
- Tools to perform preprocessing tasks like NLTK stopwords removal, lemmatization and the reasoning behind that.
- Word to vector mappings were explored. GLoVE was used for experimentation

Experiments on unsampled Raw data :

We observed a heavy bias in the data set with over 93% of the labels being “Not Troll”. As a result we decided to sample down the data to balance it out later.

But initial testing on the unsampled data is as shown below



A screenshot of a Jupyter Notebook cell. The code `traindf.groupby("target").count()` is executed, resulting in a table with two columns: `qid` and `question_text`. The table is indexed by the `target` variable, showing counts for target values 0 and 1.

| | qid | question_text |
|--------|--------|---------------|
| target | | |
| 0 | 938130 | 938130 |
| 1 | 61870 | 61870 |

Preprocessing :

Before jumping into models a small note on the preprocessing for the unsampled and sampled data :

- Case changed to lowercase
- Data transformed to only letters ie. numbers and special symbols were removed.
- Words were lemmatised

Model 1 : GLoVe embeddings and a Basic Neural Network

There was not much hope for a regular dense neural network to do well on such a heavily biased data set with just GLoVe embeddings.

There is no way to gain context with just a NN and GLoVe. Either a Bidirectional LSTM or Transformer model would have been more useful or BERT Embeddings with a Dense NN would have done the job better.

So to some extent this gives an idea as to why this model did not work out.

(PS: We will try to make an ensemble model using NN values in later submissions)

Results :

```
Epoch 20/20  
3125/3125 [=====] - 19s 6ms/step - loss: 0.2324 - accuracy: 0.9382  
<keras.callbacks.History at 0x7f1c06f841d0>
```

The training accuracy is fixed at 93.8 % which is essentially the percentage of data that is labelled 0


```
6250/6250 [=====] - 13s 2ms/step - loss: 0.2329 - accuracy: 0.9380  
[0.23293673992156982, 0.9379645586013794]
```

The testing accuracy is fixed at 93.8%

Model 2: Multinomial Naive Bayes

```
[58] from sklearn import metrics  
      predicted = MNB.predict(x_testnb)  
      accuracy_score = metrics.accuracy_score(predicted, y_testnb)
```

 accuracy_score

 0.944332

This is the accuracy we got on a 20% test split on the training Dataframe

```
True positive % : 39.26934744723424  
True negatives % : 98.07766932330378
```

It was noticed here that 40% of all the true samples were predicted correctly.

And 98% of all 0 labels were classified correctly.

This incited a will to balance the data set so that the probabilities that are going to be captured are more accurate. (Meaning that the words that identify troll comments are not diminished in terms of probability by the overwhelming barrage of 0 label data)

Under-Sampling for removing bias

To deal with the label bias, majority data points were dropped such that there are equal number of data points for both labels.

| | qid | question_text |
|--------|-------|---------------|
| target | | |
| 0 | 61870 | 61870 |
| 1 | 61870 | 61870 |

Model 1 : GLoVe embeddings on NN, with sampled data

Upon under-sampling, the following results were observed :

Training accuracy converged to 50.6%.

```
Epoch 20/20  
387/387 [=====] - 3s 8ms/step - loss: 0.6908 - accuracy: 0.5063  
<keras.callbacks.History at 0x7f18c2920850>
```

Testing accuracy came out to be 51%.

```
774/774 [=====] - 3s 3ms/step - loss: 0.6908 - accuracy: 0.5101  
[0.6907732486724854, 0.5101436376571655]
```

Model 2 : Multinomial Naive Bayes on sampled data

```
[68] from sklearn import metrics
      predicted = MNB.predict(x_testnb)
      accuracy_score = metrics.accuracy_score(predicted, y_testnb)

[69] accuracy_score

0.8504929691288184
```

85% accuracy was achieved using this model.

```
True positive % : 91.41820067409904
True negatives % : 78.71283936286838
```

This has a much better chance to predict a 'Troll' data point

Checkpoint - 2, 10th Dec

Pre - Processing

- Tweaks made to the pre - processing
- Stemming and lemmatization
- Tried removing words that were in another language or spelled incorrectly on a subset of data to see if that approach would help

Vectorization

- TF - IDF vectorizer was used parameters tuned and we settled at 1 - 4 ngram range.
- Min-df capped to 3

NB Features

- Since most people would tend not to be trolls and they would emulate a normal distribution a paper suggested that we should try using gaussian features
- So we converted the features to gaussian

Grid search and cross validation

- Stratified K-fold Cross-Validation used.
- Ditched random sampling and decided an optimal split for training the data.

Changing the threshold of logistic regression

Since we iterated over multiple values of the threshold for sigmoid to decide if a certain value of it makes our logistic regression better suited for prediction.

We got the value 0.67 as the threshold here.

Checkpoint - 3, 17th Dec

Count Vectorizer and other models

Using count vectorizer made our accuracy drop and so we ditched that idea.

We tried using our Neural net model with glove and word2vec embeddings but that yielded a result worse than the logistic regression model that we had built. As a result that model was also scrapped.

Ensemble

We tried making an ensemble with a Naive bayes and Logistic regression, two of our best models, and tried to take these 2 models' output and fed it into a Random forest classifier as input and tried using that as out final prediction but that gave similar results as the logistic regression classifier since the LR seemed to be the dominant feature and the Naive Bayes classifier was not able to predict many statements correctly that the LR has misclassified. Thus making that avenue fruitless. Perhaps more tuning would have made it yield a better result but we did not have enough time to pursue that given the last checkpoint was only a week ago.