Project 2

Jigsaw Unintended Bias in Toxicity Classification

By

Fahim Mahmood

Email: fahimmahmood@iut-dhaka.edu

Abstract: The task was to classify text as toxic comment i.e. containing racial, religious, homophobic slurs etc. or as neutral comment. The dataset contained around 1.8 million comments out of which only 3% (approx.) comments were labelled as toxic. I used XGBoost, Logistic Regressor, Linear SVM Classifiers, Random Forest, RNN (LSTM) and BERT for text classification. I obtained highest score (**80%**) using Random Forest model. I expected RNN (LSTM) and BERT to obtain accuracy close to 90% but my implementation wasn't optimal resulting in 64% accuracy and 50% for LSTM and BERT respectively.

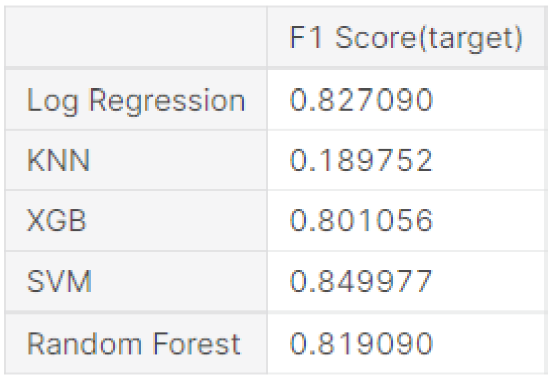
Methodology: I worked on 3 different notebook to minimize execution time. The methodology that I followed are as follows:

- Jigsaw Toxic Comment Detection.ipynb was initial notebook. I performed EDA only in this notebook to understand 'comment_text'. I dropped all other additional subtype attributes. The important ones were the comments and its corresponding labels. Since there was huge imbalance in training dataset (3% toxic & 97% non-toxic comment), I used a subset of the training dataset comprising of 30% toxic comments and 70% non-toxic comments. I compared the F1 score of XGBoost, Logistic Regression, Linear SVM and Random Forest, K Nearest Neighbor models. I opted for the model which gave good score for my cross-validation dataset. Before training, I changed the label/target of training dataset to binary value (0 or 1) as the task is similar to binary text classification. If the target value for a comment is greater than 0.7, it is 1 (toxic) otherwise 0 (neutral).

- In RNN (LSTM).ipynb, I have done similar preprocessing. Only difference is, I labelled comments with target value greater than 0.5 as 1 (toxic comment). RNN architecture in my notebook is simple LSTM (Long Short Term Memory).

- In BERT with ktrain.ipynb, I used **ktrain** library for implementing BERT. For beginners, it is quite hard to implement BERT. ktrain is a high level API which wraps **Keras** and **Tensorflow** to enable beginners implement complex models like BERT. Since ktrain has to be installed and this competition doesn't allow notebook to use internet, I had to use Google Colab and import the submission file to kaggle notebook using BERT Submission.ipynb.

Result Analysis: LSTM and BERT are state-of-art models which if implemented correctly can yield accuracy over 90%. Since I am a beginner in NLP domain and never worked with text classification, my implementation of LSTM & BERT wasn't optimal. Hence, my accuracy was low.



	F1 Score(target)
Log Regression	0.827090
KNN	0.189752
XGB	0.801056
SVM	0.849977
Random Forest	0.819090

We can see that K-Nearest Neighbor model performs poorly. I chose Random Forest even though SVM has highest score because I have tried using XGB, Random Forest, Logistic Regression and found out that the accuracy on test set doesn't vary that significantly.

Conclusion: All highest scorer in Kaggle leaderboard used some variation of LSTM like bi-directional LSTM and BERT. I think it is quite impossible to improve my accuracy much by using Random Forest or Logistic Regression. So, I intend to gain deeper insight on LSTM & BERT to improve my accuracy further.