PROJECT PROPOSAL
MACHINE LEARNING (DS-GA 1003)

Jason Phang (zp489) Kenil Tanna (kyt237) Mihir Ujjwal Rana (mur214)

1 Overview

We will tackle the Toxic Comment Classification Challenge on Kaggle. This a multi-label classification problem where the task is to categorize Wikipedia discussion comments into different classes of toxicity (toxic, severe_toxic, obscene, threat, insult, identity_hate).

2 Data [code]

The data we will be using will be the training data from the competition. It will be split into training (60%), validation (20%), and test (20%) sets, since we don't have the labels for the test set given in the competition. The column names, their respective data types, along with an example row, are given below:

	comment_id	**comment_text**	**toxic**	**severe_toxic**	**obscene**	**threat**	**insult**	**identity_hate**
dtype	object	object	integer	integer	integer	integer	integer	integer
example	003217c3eb469ba9	Hi! I am back again!\nLast warning...	1	0	0	1	0	0

where the last 6 columns are binary depending on whether the comment is classified as that label (1) or not (0).

The total number of rows are 159571. There are no missing values. For each of the labels, the value counts are given below:

	toxic	**severe_toxic**	**obscene**	**threat**	**insult**	**identity_hate**
0	144277	157976	151122	159093	151694	158166
1	15294	1595	8449	478	7877	1405

The statistics for the length (number of characters) of comments are summarized below:

	mean	**std**	**min**	**25%**	**50%**	**75%**	**max**
comment_length	394	590	6	96	205	435	5000

3 Methodology

3.1 Preprocessing

Please note that many of our models require the comments to be pre-tokenized. We follow a standard NLP text preprocessing pipeline of tokenization and sanitization (e.g. removing of stop words, rare words, lower-casing, etc.) using spaCy. In the list of stop-words, negating words (e.g. *not*, *don't*, etc.) would not be included as is popular in sentiment-analysis tasks.

3.2 Models [code]

We propose the following models in order of increasing complexity. Unless otherwise stated, each model is trained separately for each category of toxic comments (using skmultilearn package as shown in [2]). Please note, for the baseline models, we have not done stratified splits on the target labels, have not removed stop words, did not lower-case the comments, and used *max_features*=1000 (and default settings besides this).

1. [Baseline] Bernoulli Naive-Bayes with Count Vectorizer
2. [Baseline] Gaussian Naive-Bayes with Count Vectorizer
3. [Baseline] Gaussian Naive-Bayes with TF-IDF Vectorizer
4. Logistic Regression with TF-IDF Vectorizer and l-1 penalty for sparsity
5. Logistic Regression with TF-IDF Vectorizer and Gradient Tree Boosting
6. fastText [3], an augmented bag-of-words text-classification model
7. LSTM-RNN [1], with shared RNN weights and toxicity-specific fully-connected layers

4 Evaluation

4.1 Evaluation Metric

Following the evaluation metric stipulated in the Kaggle competition, we will evaluate the proposed models based on the **mean column-wise ROC-AUC** for each category; this makes sense because the labels are highly imbalanced (as is evident above), and using a metric such as accuracy would not be reflective of the true performance. Thus, we will be taking the average ROC-AUC across each category of toxic comments. More specific category-based evaluation and error analysis may be performed if performance across classes differs starkly between models. Lastly, performance on longer and ambiguous sentences will also be evaluated.

4.2 Baseline Results [code]

The ROC-AUC values for each model (individual and mean) are shown below:

	toxic	**severe_toxic**	**obscene**	**threat**	**insult**	**identity_hate**	**mean**
BernoulliNB (CountVectorizer)	0.8016	0.8981	0.8301	0.8594	0.8191	0.8318	**0.8400**
GaussianNB (CountVectorizer)	0.7560	0.7483	0.7300	0.7580	0.7150	0.7165	**0.7373**
GaussianNB (TfIdfVectorizer)	0.8359	0.8219	0.8405	0.7805	0.8190	0.7735	**0.8119**
Majority class (all zeros)	0.5	0.5	0.5	0.5	0.5	0.5	**0.5**

5 Timeline

- [Week of Mar 22] Proposal Due
- [Week of Mar 29] Analysis of Baselines, deep data-exploration, initial feature engineering
- [Week of Apr 5] Advanced NLP feature engineering, experimenting with and analyzing non-baseline, non-NN models
- [Week of Apr 12] Refining non-baseline, non-NN models, experimenting and analyzing with NN-based models
- [Week of Apr 19] 2nd meeting with Advisor, refining NN-based models
- [Week of Apr 26] Collation, cleaning, and summarizing of results
- [Week of May 3] 3rd meeting with Advisor, writing final Report
- [Week of May 10] Project Due

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[2] Shubham Jain et al. *Solving Multi-Label Classification problems (Case studies included)*. Aug. 2017. URL: https://www.analyticsvidhya.com/blog/2017/08/introduction-to-multi-label-classification/.

[3] Armand Joulin et al. “Bag of Tricks for Efficient Text Classification”. In: *arXiv preprint arXiv:1607.01759* (2016).