Unintended Bias in Online Comment Toxicity Classification

Jae Y. Sohn, Sally Hahn

[1] Princeton University, Department of Economics [2] Princeton University, Department of Mathematics

jaes@Princeton.edu, yhahn@Princeton.edu

ABSTRACT

- There is a need for a fair and scaleable moderation system that can more readily and efficiently (compared to human peer-reviews) evaluate comments on the online forum.
- The key here is to learn with enough certainty given unintended bias in such a model - nuances and the detection of context-based trigger words such as “gay” and “homosexual” may easily mislead prediction mechanisms.
- Furthermore, it is important to keep in mind the distinction between what is really considered “toxic”, and merely disagreeable comments.
- The aim of this project is to explore different models that can begin to fulfill this moderation role and accurately classify purely toxic comments while regarding potential biases. Perhaps in the future, civil conversation can take place with minimal harassment, leaving room for more organic dialogue and worthwhile exchange of ideas.

DATA DESCRIPTION AND ENGINEERING

Google Jigsaw Dataset Example Entry

> Continue to stand strong LGBT community. Yes, indeed, you’ll overcome and you have.

The above comment has the following label scores:

Toxicity Score: All 0.0
Identity Labels: homosexual_gay_or_lesbian: 0.8, bisexual: 0.6, transgender: 0.3 (all others 0.0)

What does the entire dataset look like?

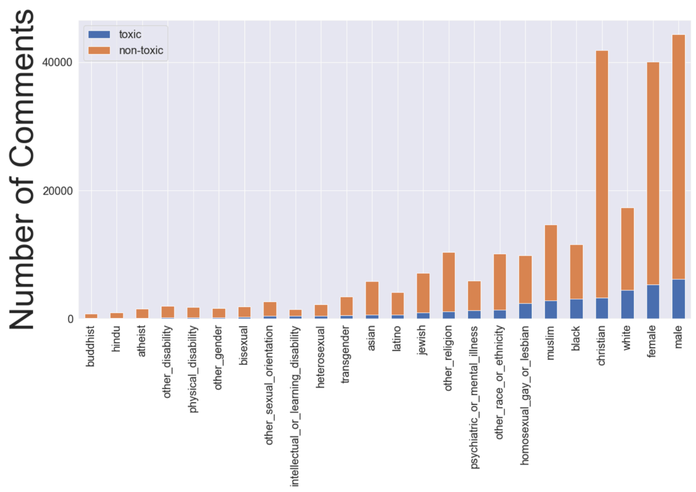


Figure 1 shows that gender identity labels male and female, and race identity labels white and black are extremely prevalent in toxic comments. Overall, the number of comments including the identities are also shown by the stacked barchart including the non-toxic context; in this sense, male, female, and christian are the most commonly tagged identities.

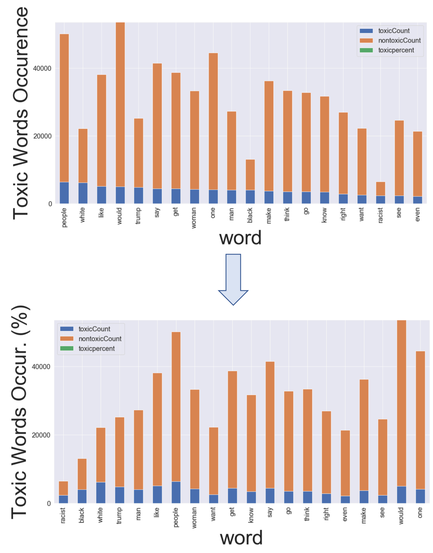
Cleaning up our dataset:

- **Train/Test Split:** We used sklearn tools to randomly split the data set into 80% train and 20% test sets. Then, the column `toxicity' was extracted to be target value represented by y_train, y_test, respectively. Since we are attempting to predict toxicity based on comments, all other columns except `comments' were dropped in X_test. Hence, the challenge lied in matching the occurrences vector of the two X datasets, which were of different dimensions.
- **Missing Values**; Not all comments in X_train were evaluated by peers and the rows corresponding to these comments had missing values, and thus these rows were dropped. Dropping some data allowed us to not only delete NaN values, but also reduce the size of the data set and thus significantly decrease running time.
- **Bag of Words:** We used the Bag of Words representation to vectorize the comments. Punctuation was removed and computer languages such as <href or \n were removed. Stop words were removed from the list of tokens, words were lemmatized, then tokenized.
- **Converting Float64 Values:** Standard classification-based sklearn models require target values to be in categories or discrete values. Because our identity and toxicity values are in fractions denoted in float64 data type, we had to normalize onto a workable scale by changing all values to Boolean outcomes.

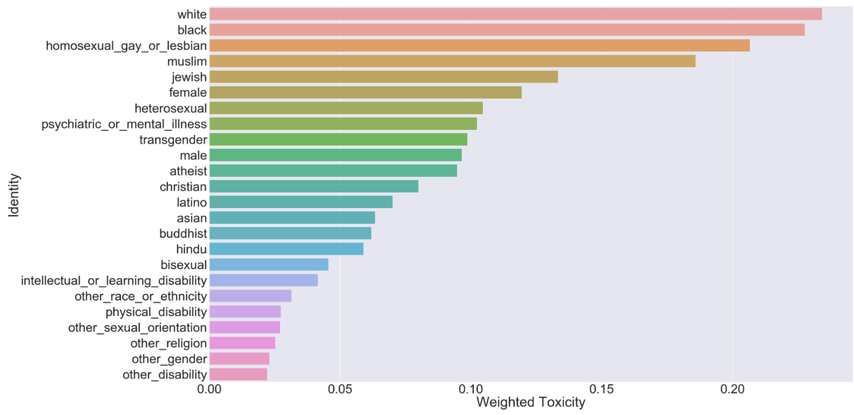
METHODOLOGY AND APPLIED MODELS

Results of Clean Up – Words Occurrence

Figure 2 displays the words in descending order, organized by the number of times they appear in toxic comments. Some words, such as *people, like* and *say*, clearly do not contain toxic elements. To combat this issue, we account for it in the Figure 3 by taking a percentage value instead. This methodology rules out words that are merely numerous in all comments and thus commonly found in toxic comments. In this new figure, the most commonly racist words comes out to be *racist, black, white, trump, man*, and so on



Results of Clean Up - Weighted Toxicity by Identity Label



Machine Learning Models Used

Bernoulli Naïve Bayes: Logical choice given Boolean nature of data set (after data engineering).
Logistic Regression: Binary classification problem, applied using *class_weight = ‘balanced’* in order to take into account the 10/90% toxic/non-toxic imbalance.
Support Vector Classification: Very popular methodology applied by researchers in natural language processing. We alsp applied *class_weight = ‘balanced’* in order to take into account the 10/90% toxic/non-toxic imbalance. Because our data set was of the order of tens of thousands. We used the linear SVC library.
Random Forest: Model is capable of handling imbalanced data; applied with hyperparameter *n_samples*=1000

Methodology

This project's goal is to calculate the toxicity levels of comments.
This required 2 separate steps:

1. **Predicting Identity Labels**: We needed to find a way to predict identities from comments, aka generate identity columns from only comments and append the results to X_test. (80-20 split again into X_test’)
2. **Predicting Toxicity Scores**: From this newly filled X_test, we have to be able to predict toxicity levels to test against y_test.

METRICS (Overall AUC and Generalized Mean Bias AUC)

For **Overall AUC,** we used scikit learn’s *roc_auc_score* library to calculate the ROC-AUC for the full evaluation set. This AUC_overall value is weighted with w_0 = 0.25 in the Final Metric.

$$w_0 \mathrm{AUC}_{overall}$$

For **Bias AUCs**, we took three different subgroups to more precisely evaluate bias:

Subgroup AUC: We restrict the data set to only the entries that mention the specific identity subgroup.

BPSN (Background Positive, Subgroup Negative) AUC: We restrict the data set to non-toxic entries that mention the identity and toxic entries that do not.

BNSP (Background Negative, Subgroup Positive) AUC: We restrict the data set to toxic entries that mention the identity and non-toxic entries that do not.

Taking these three Bias AUCs, we calculate the generalized mean:

$$M_p(m_s) = \left(\frac{1}{N} \sum_{s=1}^{N} m_s^p \right)^{\frac{1}{p}}$$

where:
M_p = the pth power-mean function
m_s = the bias metric m calculated for subgroup s
N = the number of identity subgroups.
In accordance with Borkan *et al.*’s methodology, we also applied p value of -5.

Combining the above AUCs, we get the Final Metric:

$$\text{score} = w_0 \mathrm{AUC}_{overall} + \sum_{a=1}^{A} w_a M_p(m_{s,a})$$

where:
A = number of submetrics (3)
$m_{s,a}$ = bias metric for identity subgroup s using submetric a
w_a, w_0 = are weights for the relative importance of each submetric;

RESULTS (Baseline w/o Identity, Project Approach w/ Identity)

W/o Identity	Train Accuracy	Test Accuracy	Overall AUC	Subgroup AUC	BPSN AUC	BNSP AUC	Final Score
Random Forest	0.8927	0.8943	0.5690	0.5554	0.5725	0.5524	0.5582
Linear SVC	0.8988	0.8983	0.5455	0.5326	0.5360	0.5411	0.5388
Logistic Reg	0.8989	0.8983	0.5569	0.5451	0.5433	0.5547	0.5500
W/ Identity	**Train Accuracy**	**Test Accuracy**	**Overall AUC**	**Subgroup AUC**	**BPSN AUC**	**BNSP AUC**	**Final Score**
Random Forest	0.9908	0.8849	0.5554	0.5310	0.5571	0.5300	0.5435
Linear SVC	0.7930	0.6645	0.6934	0.6527	0.5852	0.7592	**0.6726**
Logistic Reg	0.7818	0.6589	0.6934	0.6560	0.5904	0.7579	**0.6801**

*For references and citations please see paper submission.