# Classifying Toxic Comments:

Using machine learning to find the bad ones

Dovid Burns

5/6/18

Springboard Capstone Project

## 1. Define the Problem

Toxic comments posted in public forums online are common, and they are so corrosive that they rapidly shut down otherwise engaging discussions. Many online platforms that allow for user comments such as Facebook, YouTube, Twitter, Wikipedia, Yelp and Instagram have difficulties ensuring that conversations are taking place in an appropriate way. This project will build a classifier using a dataset containing comments from Wikipedia's talk page edits to classify them as toxic or benign. The model can then be used in many platforms to automatically detect—and possibly remove—these comments before they offend participants or deter users from engaging in communications. Additionally, the model can be used as a tool to identify and track inappropriate users, allowing platform administrators to issue serious warnings or altogether ban participants who chronically post toxic comments.

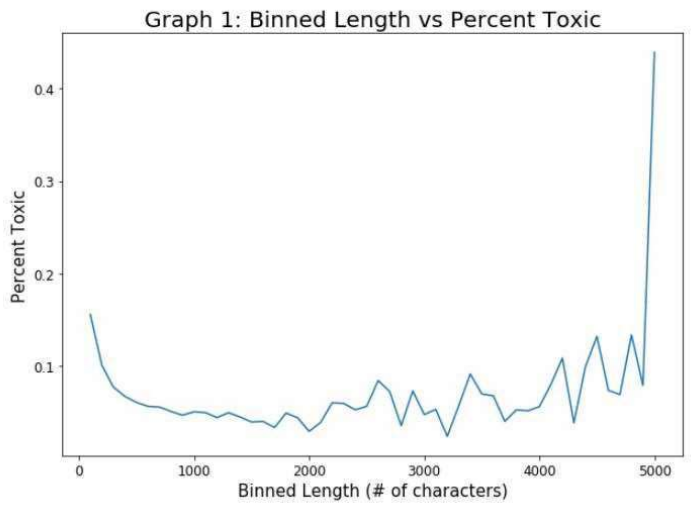
## 2. Identify the Client

The client who created this dataset is the Conversation AI team. They are a research initiative founded by Jigsaw and Google who work on making tools to help improve online conversation. The area of interest for their analysis is the study of negative online behaviors, one of which is toxic comments. These remarks are defined as being rude, disrespectful or other corrosive forms of language that shut down public online discussions. Conversion AI currently employs systems working to identify toxic language, but these models still make many errors. This project will seek to correct these errors in order to create a stronger model for the Conversation AI team.

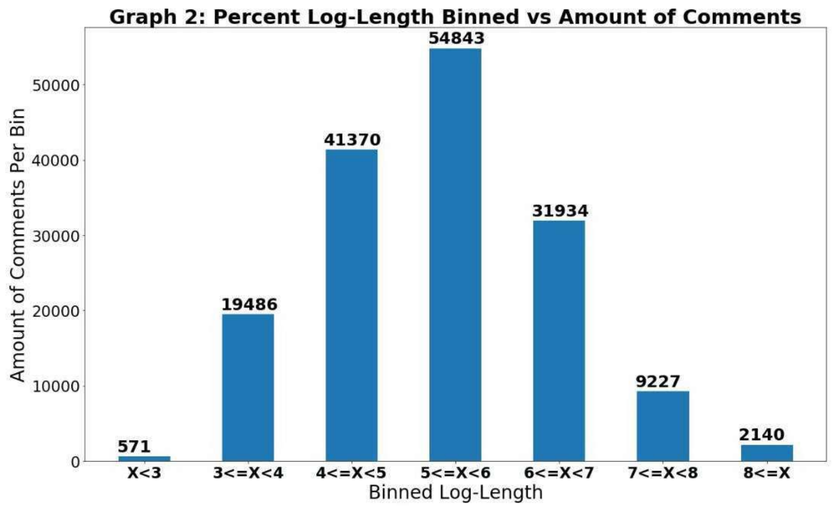
## . Describe the Data  et

This data set contains 15  ,5  1 comments taken from Wikipedia talk pages.  f these public comments, 15,    are tagged as toxic. These have been flagged manually by human raters.  ach comment provided is between 6 and 5,    characters in length. The data set to be used is publicly hosted at https //www.kaggle.com/c/jigsaw toxic comment classification challenge.

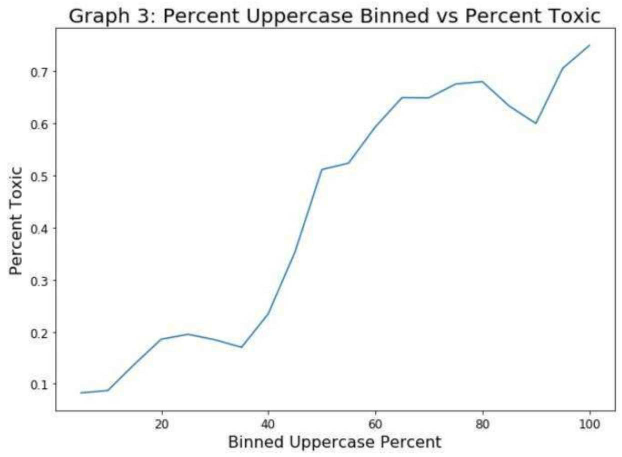
To create predictive variables, the data was binned by length with each bin increasing in length by 1   characters  see Graph 1 .



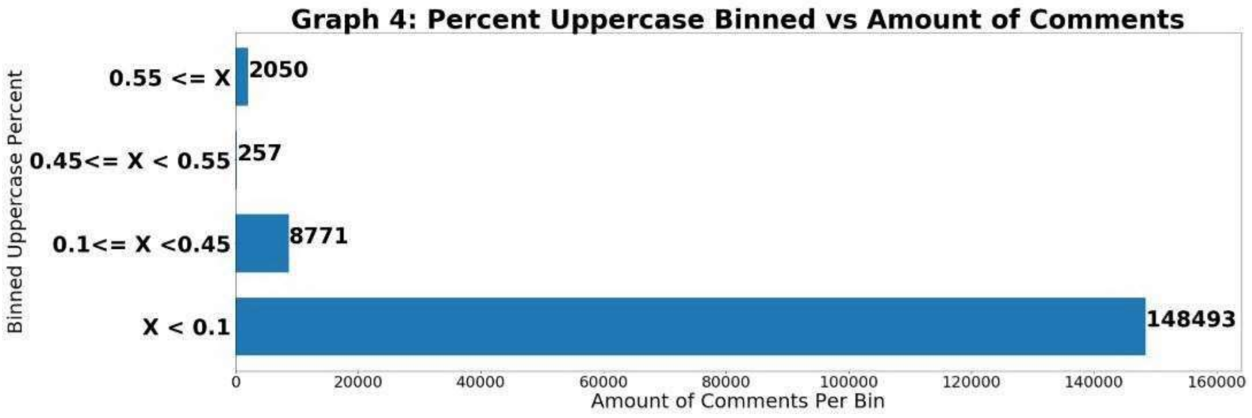
The resulting data was then analy  ed for trends in the rate of toxic comments. Since there was such a wide range of comment length, we took log length as well to analy  e and bin accordingly. We found seven distinct bins for analysis  see Graph   below .



The final variable created was the percent of uppercased letters in the comment. This was binned starting at 5 , increasing by 5 each time to see the trends in percent of toxic comments per group see Graph .



We established four bins of percent uppercase as a useful variable see Graph below .



The text in the comments re uired significant pre processing before it was useful for analysis. The first steps were to remove all of the new line characters, make the words lowercase, and remove apostrophes. After regex was used to filter out any non words from the comments, a clean string of words remained. We then vectori ed these words using Count ectori er to prepare them for the model. Additionally, the words were stemmed using a SnowballStemmer to simplify similar words for the model.

## .  ther Potential Data

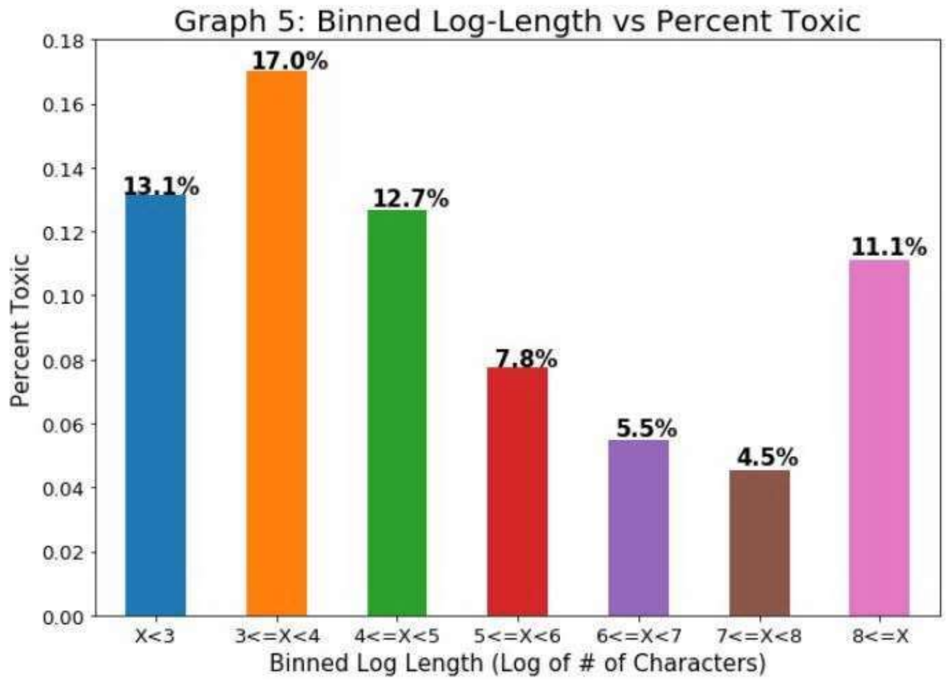
Because our client is interested in improving online communication in general—not exclusively comments on Wikipedia—there are other features and data that would strengthen our ability to predict toxic comments across multiple platforms. For example, if our data set was broader, including comments from YouTube, Facebook or  uora, the model built would most likely generali e better outside of Wikipedia. It might also be useful to analy e the amount of time that elapses between the initial view of a post and the submission of a response. Perhaps users who spend more time considering their responses are less likely to post a toxic comment. Also, long term tracking of users across multiple platforms would inform the model of toxic comments made by users over time and across the web. This information could very likely be a useful predictor of future toxic comments. Additionally, the length of time in which the user has been a member of the community in which they are posting toxic comments could be a significant prediction variable. Another interesting feature that could be created would be to examine the percent of words that were spelled incorrectly in a comment. Perhaps this feature would prove to have predictive value in our model.

 astly, it is very important to note that every one of these comments were labeled by humans as being toxic or benign. This introduces many possible issues into the model, the first of which is the subjective nature of humans identifying comments as toxic or offensive. This means that there will be some inherent margin of error in whatever model is created from this data. Another problem is the likelihood of human error, meaning analysts will erroneously flag comments on occasion. To avoid these issues, a reasonable solution would be to have multiple people scoring each comment. Should disagreement arise between analysts, the comments would then be reviewed by yet another person or executive team for a final decision on the comments. Additionally, it is essential to note that a clear set of guidelines must be implemented by the people labeling the comments. Training of analysts would have to be effective and extremely consistent.

## . Initial   indings

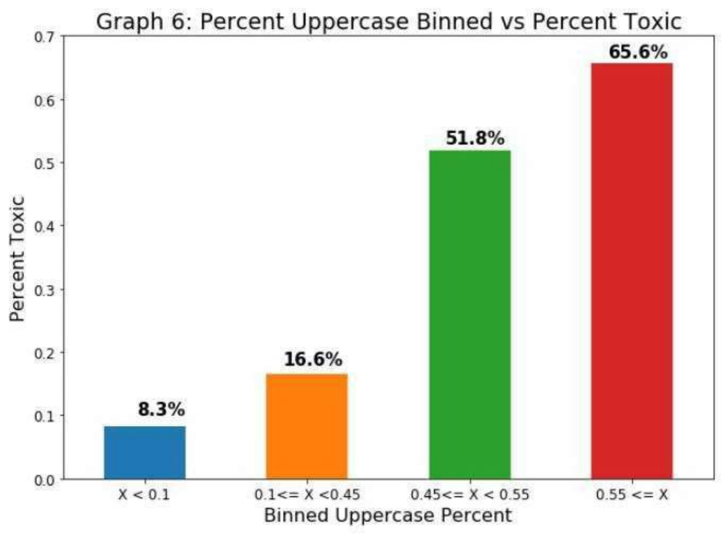
***Disclaimer: the dataset contains text that may be considered profane, vulgar, or offensive.***

The created variables percent of uppercase characters and log length of each comment were both strong indicators of comment toxicity. In order to verify that these were statistically valid variables to use, a chi s  uared contingency test from the scipy.stats packages was used to find a p  value of less than   .    1 for both variables. The rates of toxic comments per Binned   og   ength and Binned Percent Uppercase are demonstrated in Graphs 5 and 6 below.



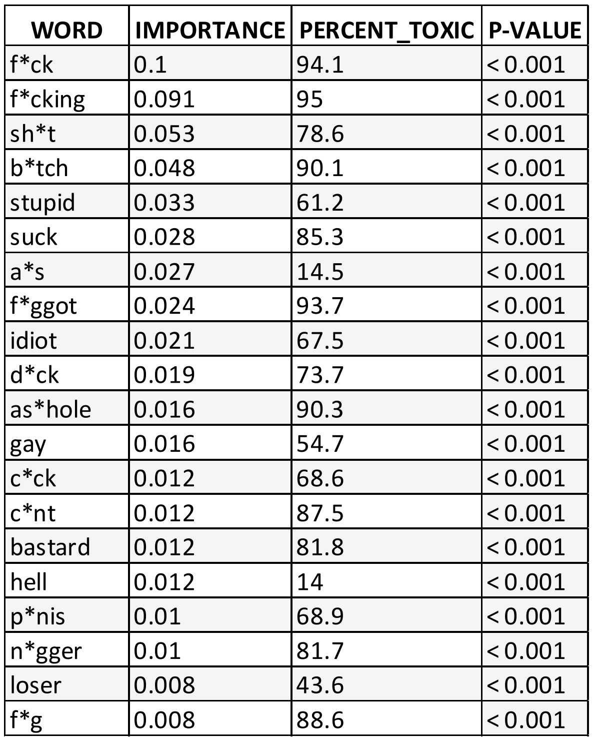
As indicated above in Graph 5, the overall trend for length is that comment length and likelihood of toxicity are negatively correlated. This makes sense  people generally write offensive comments in brief, not typically taking the time or effort to compose toxic essays. This trend fails at the beginning and end of our data, meaning very short comments do not have the highest fre  uency of toxic posts, while extremely long comments show an increased tendency to be toxic.

The trend in percent uppercase is very clear as well  the percent of characters that are capitali  ed is positively correlated with likelihood of comment toxicity. This is both a strong statistical indicator as well as a logical communication pattern. Capitali  ation typically expresses increased vocal volume, which is consistent with communicating anger verbally. People seeking to express toxic ideas would likewise wish to communicate their anger, even in the context of online discourse.



While a similar analysis cannot be performed for the vectori  ed words, the    andom Forest Classifier gives the most important features in the model. When we ran a model with only the vectori  ed words as features, we found that several words strongly predicted toxic comments  see Table 1  . We then created a subset of the dataset to find groups of comments that contain each of these words.    ach word specific subset was then analy  ed to find the percent of that subset that were tagged as toxic comments  P     C     T  T     IC Table 1  .    astly, we did a chi s  uared test to find the statistical significance of comments containing each of these words verses percent toxic  P    A  U   Table1  . All the p  values from these chi  s  uared tests were less than   .    1, confirming that these words are statistically significant indicators in determining toxicity. The words have been censored in Table 1 to reduce abrasiveness.

*able : andom orest mportant eatures*



| WORD | IMPORTANCE | PERCENT_TOXIC | P-VALUE |
|---|---|---|---|
| f*ck | 0.1 | 94.1 | < 0.001 |
| f*cking | 0.091 | 95 | < 0.001 |
| sh*t | 0.053 | 78.6 | < 0.001 |
| b*tch | 0.048 | 90.1 | < 0.001 |
| stupid | 0.033 | 61.2 | < 0.001 |
| suck | 0.028 | 85.3 | < 0.001 |
| a*s | 0.027 | 14.5 | < 0.001 |
| f*ggot | 0.024 | 93.7 | < 0.001 |
| idiot | 0.021 | 67.5 | < 0.001 |
| d*ck | 0.019 | 73.7 | < 0.001 |
| as*hole | 0.016 | 90.3 | < 0.001 |
| gay | 0.016 | 54.7 | < 0.001 |
| c*ck | 0.012 | 68.6 | < 0.001 |
| c*nt | 0.012 | 87.5 | < 0.001 |
| bastard | 0.012 | 81.8 | < 0.001 |
| hell | 0.012 | 14 | < 0.001 |
| p*nis | 0.01 | 68.9 | < 0.001 |
| n*gger | 0.01 | 81.7 | < 0.001 |
| loser | 0.008 | 43.6 | < 0.001 |
| f*g | 0.008 | 88.6 | < 0.001 |

## . Machine earning

### a. Process er ie

The machine learning process has many stages. We first pre processed the data to be usable in the models and then created three different models using a Multinomial a ve Bayes, andom Forest and AdaBoost classifiers. We created C curves for each model to demonstrate effectiveness and picked the best model based on the highest AUC score on the test data.

### b. Data Pre Processing

After vectori ing the words per comment and the binned variables, the final step was to combine these in a way that a machine learning model could learn from them. We first ran a get dummies function on both binned variables, the og length and percent capitali ed. We dropped the first dummy column to prevent variable correlation. The features that are created by count vectori er are stored in a sparse matrix with elements in Compressed Sparse Column format. We used the hstack method from the scipy.sparse library to combine our dummy variables with the vectori ed words. This completed the data pre processing stage. We then split the data into training and testing groups that allowed for simulated model scoring on new data. To make sure we had the optimal hyper parameters, we ran a cross validated grid search for the count vectori er method on both the a ve Bayes and andom Forest Classifiers. nce we had the best parameters for the count vectori er, we did a similar cross validated grid search on the a ve Bayes and andom Forest Classifiers.

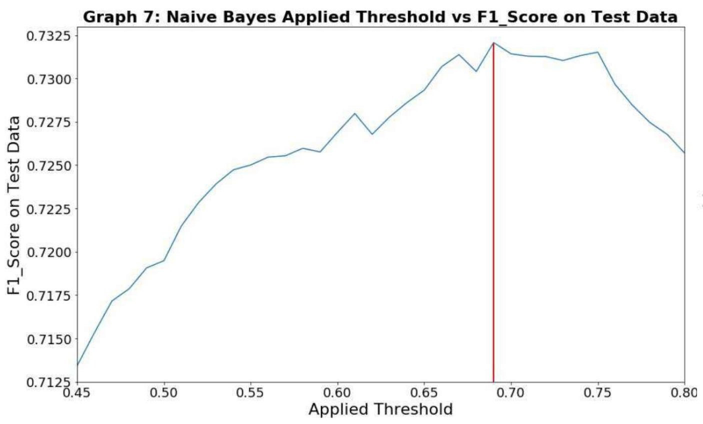
## c. a e ayes Classifier

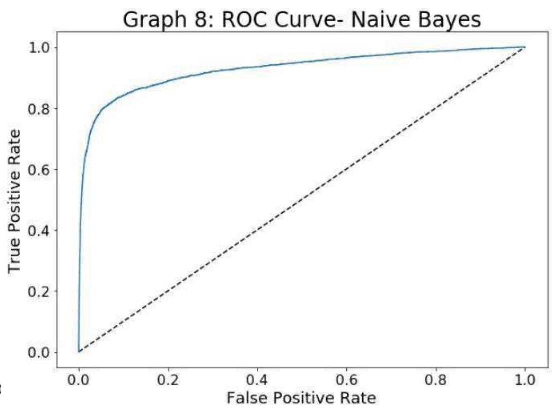
The optimal hyper parameters for count vectori er as determined by grid search with Multinomial a ve Bayes classifier optimi ing for roc auc was a min df of 15, a max df of . and no max features. Using these parameters, we also found that a value of Alpha 1 was optimal for the Multinomial a ve Bayes classifier. The following scores were obtained when we ran this algorithm on the stemmed data see Table

*able : Multinomial a ve ayes lassifier cores*

| **Metric** | **core** |
|---|---|
| Accuracy on Training Data | .6 |
| Accuracy on Test Data | . |
| F1 Score on Test Data | . |
| **C on Test Data** | **. 2** |

When we examined various thresholds besides the default 5 , we found that a threshold greater than .6 gave the highest F1 Score of . see Graph . The C Curve is illustrated in Graph 8 below.





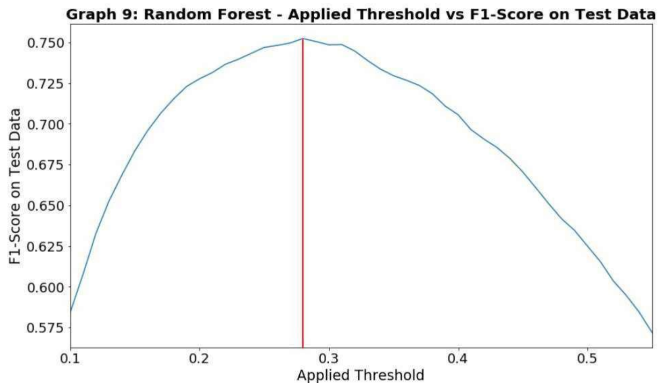
## d. andom orest Classifier

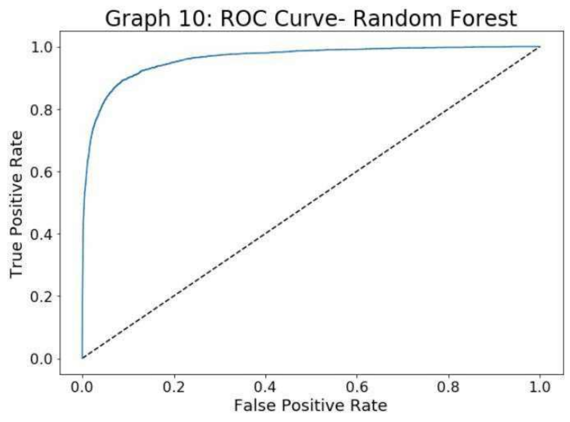
The optimal hyper parameters for count vectori er as determined by a grid search with a andom Forest Classifier optimi ing for roc auc was a min df of 1 , a max df of . and no max features. The grid search optimi ing roc auc found that a andom Forest with bootstrap False, criterion gini , max depth none, max features auto , min samples leaf 1 , and min samples split optimi ed the model. Using the above stated parameters on stemmed words, the model achieved the follow results see Table

***able : andom orest lassifier cores***

| **Metric** | **core** |
|---|---|
| Accuracy on Training Data | . |
| Accuracy on Test Data | . |
| F1 Score on Test Data | .6 5 |
| AUC on Test Data | . 61 |

When we analy ed different thresholds besides the default 5 , we determined that a threshold of greater than . 8 gave the highest F1 Score of . 5 . The C Curve is shown in Graph below.





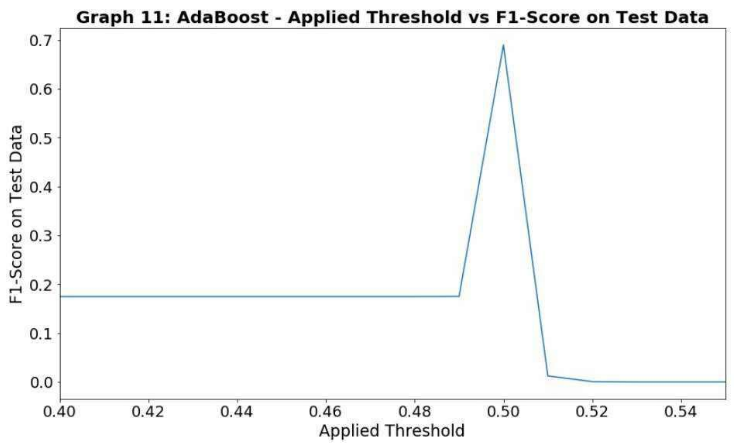
## e. da oost Classifier

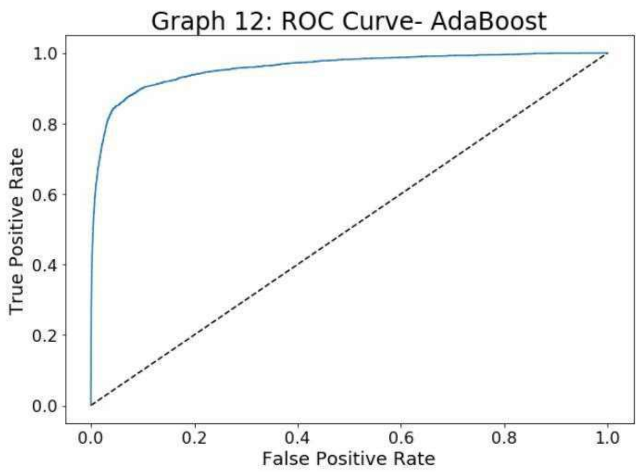
The optimal hyper parameters for count vectori er as determined by a grid search with a AdaBoost Classifier optimi ing for roc auc was a min df of , a max df of . and 1 , for max features. The grid search optimi ing roc auc found that a AdaBoost with a learning rate of .1 and 5, estimators optimi ed the model. Using the above stated parameters on stemmed words, the model achieved the follow results see Table

***able : da oost lassifier cores***

| **Metric** | **core** |
|---|---|
| Accuracy on Training Data | 5. |
| Accuracy on Test Data | 5. |
| F1 Score on Test Data | .6 |
| AUC on Test Data | . 5 |

When we analy ed different thresholds besides the default 5 , we determined that applying any threshold other than the default of .5 dramatically decreased the F1 Score see Graph 11 . The C Curve is shown in Graph 1 below.





## . Concl sion

All three models accurately predicted which comments were toxic. andom Forest had the best scores with an AUC of . 61 and a F1 Score of . 5 after applying a threshold. ur client can improve their current system of tagging comments using any of these models. Because our client is seeking to improve online dialog, we recommend that public platforms implement a system of blocked or flagged words based on the andom Forest model's top predictors. When the application blocks predictably toxic comments, public dialog will be able to maintain the level of decency necessary to sustain meaningful online discourse. This model could be implemented real time in its entirety to ensure most toxic comments are flagged as they are being made. Unfortunately, the program would likely erroneously flag some comments as toxic. These errors would need to be addressed in a timely manner by staff trained to manage the application and its errors. astly, it is important to note that a summary of all a user's comments can be run through this algorithm to see how many of them are toxic. A score could then be generated from the comment history for each user, which could be used to warn users and the platforms they fre uent of their history of making toxic comments. Further, those users with poor scores could be flagged for extra monitoring, or they could be outright banned from posting future comments. verall, this report shows that there is real room for machine learning to contribute in a meaningful way to maintaining the safety and appropriateness of online discourse.