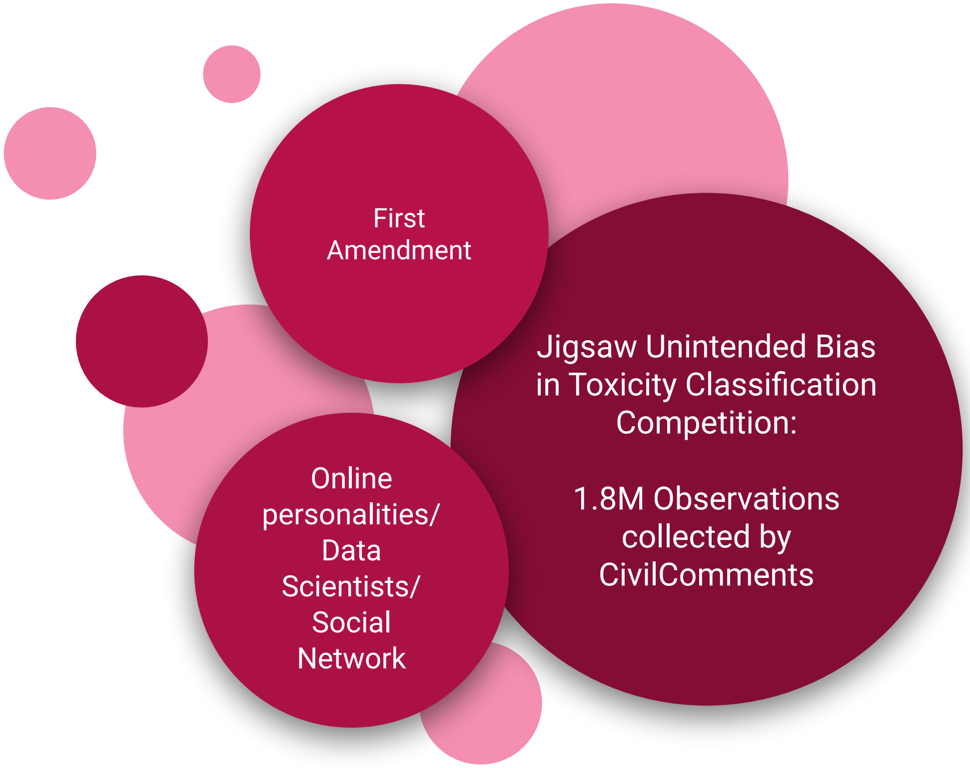
No Hate All Love

Loren Hinkson
Andrea Koch
Natasha Mathur

Goal:

More specifically identify hate speech



We had a large amount of data.

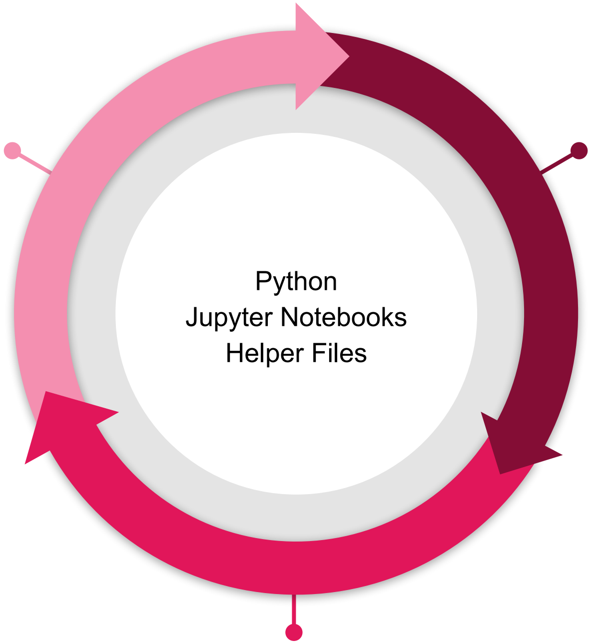
Data Management

STORE

AWS S3 Buckets
Google Cloud Storage

BUILD

AWS SageMaker (EC2)
RCC & GoogleCoLab



PICKLE

Save different stages of the process

Data Exploration

Data Exploration

1.8 Million Comments

Vary greatly in length

Identity matters

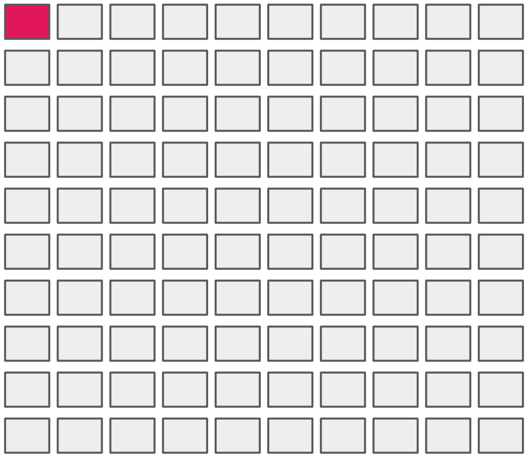
Comments associated with an identity more likely to have a high toxicity score

Imbalanced data set

Only 6% of comments are actually toxic

Bias in the toxicity scores

Human factors to creating scores and framing of the public policy issue; based on social norms



Preprocessing & Features

Preprocessing & Feature Generation

01	**Cleaned**	• Lowercasing • Punctuation removed • Strings and lists
02	**Stop Words**	• Removed some but not all • Left in some stop words such as “can’t” and “won’t” because they have negative sentiment
03	**Stemming**	• Porter • Lancaster
04	**Embedding Weights**	• Initially started with zero weights and let model learn organically • In later steps, applied pre-trained embedding weights from GloVe
05	**Word Embeddings**	• Create a numerical representation of each word • Based on cleaned text with no stemming and stop words removed

Preprocessing & Feature Generation

01

Original Comment

We are now .5 or a half point behind the US. Close the gap and the problem will start to resolve itself. Its called monetary economics.

02

Cleaned, no stem, w/o stop words

we 5 half point behind us close gap problem start resolve its called monetary economics

03

Porter Stemming

we 5 half point behind us close gap problem start resolv it call monetari econom

04

Lancaster Stemming

we 5 half point behind us clos gap problem start resolv it cal monet econom

Model Development

Round 0:
Preliminary Models

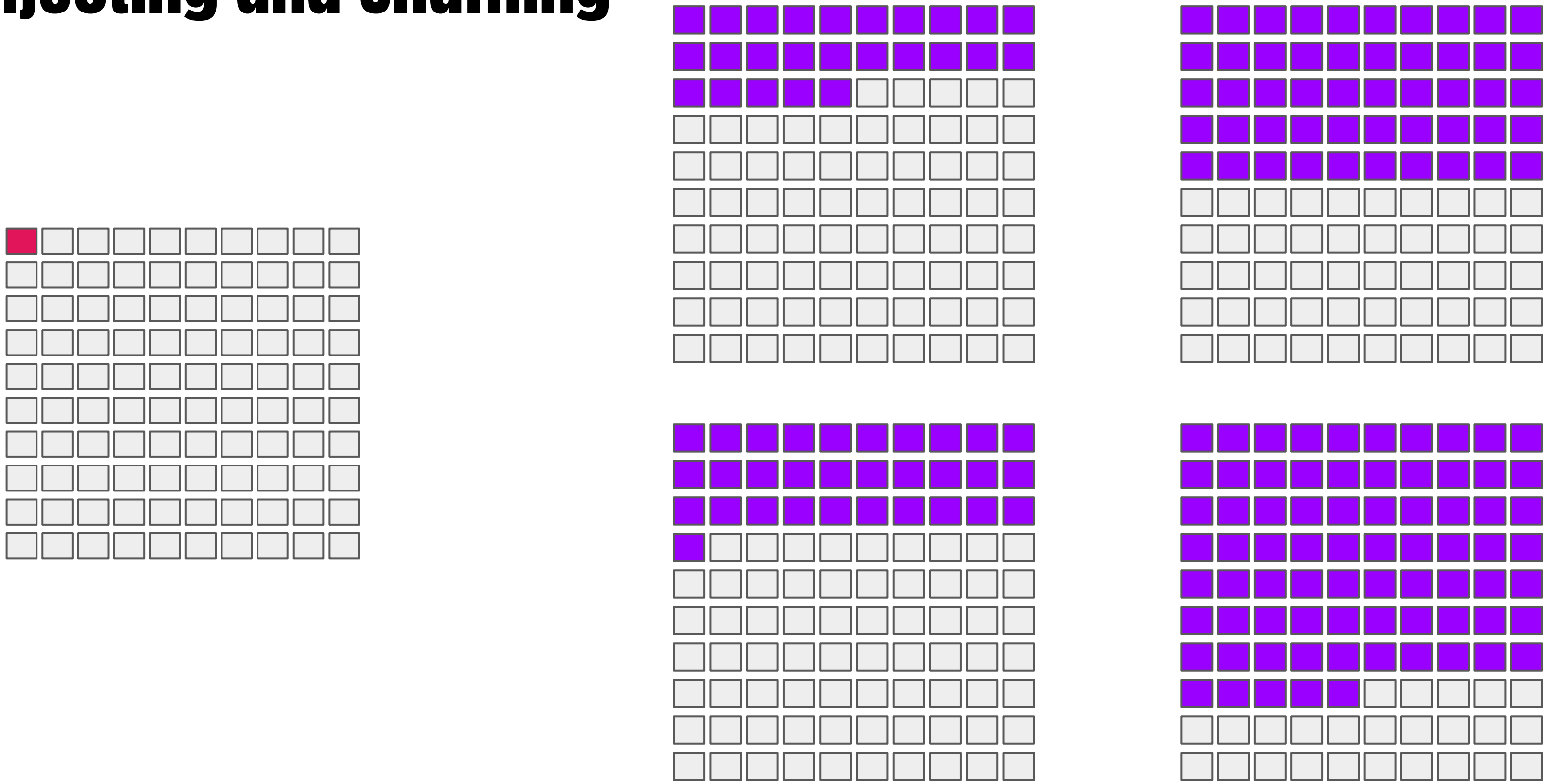
First Models

1. Ran a quick Naive Bayes, Support Vector Machines, and Logistic Regression
2. Got seemingly high accuracy scores
3. Main takeaway: **Everything was labeled non-toxic**

“Look at me!”

-the Toxic Comments

Injecting and Shuffling



Metrics

Metrics

- We initially evaluate the models based on precision and F1 score, but realized that that did not capture what we were actually interested in.

- More concerned about intervening on toxic comments
  - False negatives rather than false positives
  - Accuracy → Recall

- False positives could be addressed through a secondary model (will go into that more later) or human review

Round 1: Naive Bayes

Naive Bayes Models

- Trained on a weighted data set and tested on both weighted and normal data
- Consistently found that unstemmed comments contributed to models that performed better
- Found that models performed noticeably better on weighted testing sets - illustrating that the disparities in the data sets impaired model performance
  - Highly probabilistic

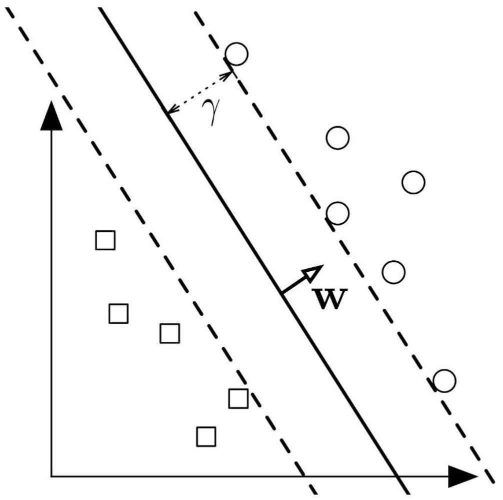
Naive Bayes Models

	Naive Bayes
Training Dataset	
Parameter tuning	~257K observations (50% toxic, 50% nontoxic comments)
Model Selected	
Overall accuracy	75.951%
Overall recall	83.750%
Target Recall	83.750%
Non-target Recall	75.465%
Strong Identity Recall	87.616%
Obscenity Recall	83.213%
Insults Recall	85.480%
Threats Recall	85.472%
Target Precision	100%

Round 2:
Support Vector Machines

Support Vector Machine Models

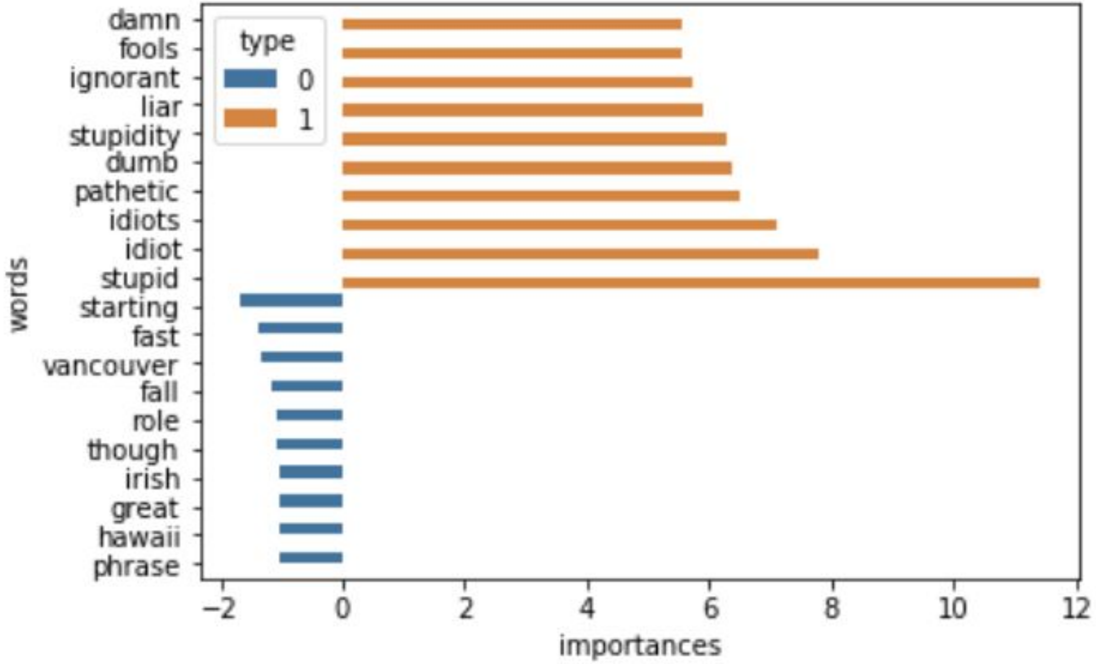
- SVM models do not create a probabilistic model
- SVM creates each class and makes sure that all values are a certain distance away from the boundary.
- This boundary isn't affected by new data, unlike a linear model, so we thought that it would better maintain the characteristics of the training model even when faced with the vastly different composition of the testing data.



Support Vector Machine Models

	SVM
Training Dataset	
Parameter tuning	~21K observations (25% toxic, 75% nontoxic comments)
Model Selected	
Overall accuracy	95.301%
Overall recall	59.100%
Target Recall	59.100%
Non-target Recall	97.217%
Strong Identity Recall	47.945%
Obscenity Recall	55.172%
Insults Recall	68.073%

What made a difference?



Let’s try more layers . . .

Round 3:
First Neural Net

Initial neural network

- 2 layer neural network
- Linear layers
- Improved when dropout applied, and with more epochs but . . . .
- Overwhelmingly classified all comments as non-toxic

Round 4:
Long Short-Term Memory

LSTM's Next Top Model:

Hypothesis

Larger datasets with a larger proportion of toxic comments to learn from would yield best results

Testing

Multiple iterations on increasingly large rates of toxic comments, varying dataset size and number of epochs

Results: Partially Correct

Our Best Model: 40% of training data available (576K comments) resampled to 60% toxicity

- Balance of Recall between classes
- Higher secondary metric performance

LSTM Model Results

	LSTM
Training Dataset	
Parameter tuning	~288K observations (75% toxic, 25% nontoxic comments) after a single epoch
Model Selected	
Overall accuracy	88.164%
Overall recall	88.140%
Target Recall	88.188%
Non-target Recall	88.000%
Target Precision	99.483%

First Iteration: Toy Model

- “Mini” dataset of 10K used in initial creation of LSTM model
- Batch size of 1 to start
- Very small number of hidden and embedding dimensions
- Manually created word-to-idx and idx-to-word for sentence sequences
- Tested other packages like **Keras**, but found **PyTorch** was more transparent for beginners

Toy Model Challenges

- Entailed a lot of experimentation and trial-and-error with dimensionality
- Efficiency: LONG run time (multiple hours for a single model on small dataset)
- Not scalable

Solution:

Batching, GPU usage

Second Iteration: Building for Scale

Implemented batching with **Torchtext** and Pytorch **torch.cuda() GPU** methods

- ➔ Faster, more efficient training (~12 min to train and evaluate 1M+ comments)

Applied pre-trained word embeddings via Stanford's **GloVe**: Global Vectors for Word Representation (Pennington, Socher, Manning, 2015)

- ➔ Increased performance in key metrics (recall)
- ➔ Putting correct amount of weight on words based on co-occurrence

Second Iteration Challenges

Challenges:

- Conversion to executable script; testing and debugging with limited access to RCC GPU's
- Amazon GPU Costs

Solution:

Google CoLaboratory for free GPU access

Future

Challenges

Classification

1

Difficult to find a model that performs well for both toxic and non-toxic comments

Overfitting

2

In the LSTM model we saw that many models began overfitting after 2 or 3 epochs.

Size of data

3

Large data size led to sampling and we saw inconsistencies in the language used across different samples

Next Steps

Layer models

Layer the models that are better for each group on top of each other.

- Use a model optimized to find toxic comments and then narrow it down using a model optimized for non-toxic comments.

Vary the data

Use a wider variety of non-toxic comments from different source types (i.e. Twitter, Reddit, YouTube)

- Although the overall quantity of comments was sufficient, the toxic comments were limited both in comparative quantity and their source.

Thank you!

Loren Hinkson, Andrea Koch, & Natasha Mathur