STANCE DETECTION IN FAKE NEWS

BY VAIBHAV RAWAT AND SAMBARTIKA GUHA

Why fake news detection?

Jackie Chan Dead At Age 63 Is A Celebrity Death Hoax

Shawn Rice – December 17, 2017 Follow @TheShawnRice

Pope Francis Shocks World, Endorses Donald Trump for President, Releases Statement

Pope Francis Shocks World, Endorses Donald Trump for President, Releases Statement VATICAN CITY – News outlets around the world are reporting on the...

INTRODUCTION

- News is a relevant part of our daily life.
- Social media sites are the most accessed and impactful sources of news.
- News from social sites turns viral in a matter of hours.

MOTIVATION

- Fake News is spread across social media sites.
- Recent US elections demonstrate the influence of fake news on general public opinion and election results.
- Manually impossible to determine the authenticity of all the news on the web.
- Need for an automated mechanism to identify the fake news.

PROBLEM STATEMENT

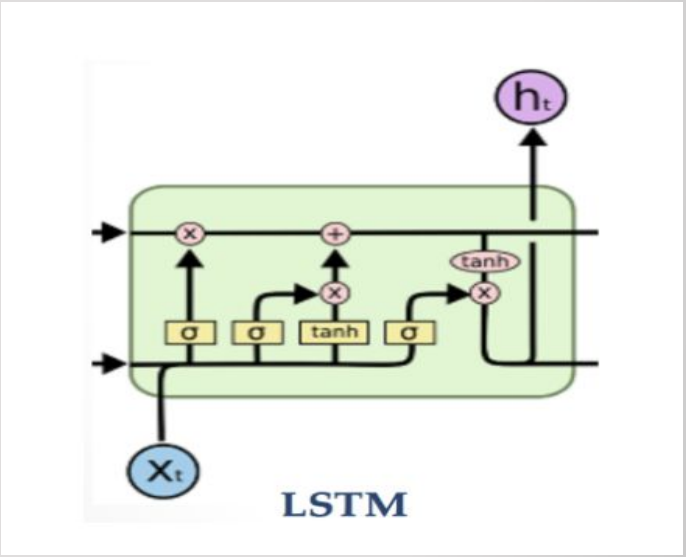
- Fake news detection is a challenging problem.
- Stance detection among news headline - body pairs can significantly help in Fake News detection.
- Fake News Challenge (FNC1) presents a dataset, a collection of news articles with resemblance to real world news for the task of stance detection.
- Aim is to identify whether the article body agrees, disagrees or discusses the news heading or is entirely unrelated.

PROPOSED SOLUTION

GLoVE embeddings (50 dimensional) for words in news corpus were used with following models :

- Feed-forward neural network:
  Embeddings of headlines and body are averaged separately and concatenated to train the model.
- Long Short Term Memory Model (LSTM):

  Since LSTMs are good at capturing long term dependencies, embedded vectors of concatenated header and article are used for training LSTM layers.



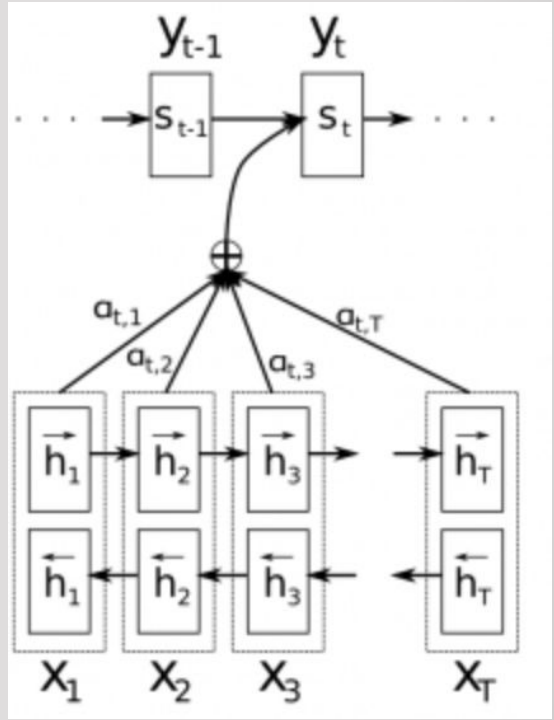
PROPOSED SOLUTION

- LSTM with attention:

❏ Attention mechanism is good at learning a summarized context and performs well with large inputs.

❏ So attention model is constructed over the first L output states from headlines of the articles.

❏ Relevant context is extracted from headlines and articles and so should improve the performance for stance detection.



PROPOSED SOLUTION

- LSTM with Attention equations:

$$
\begin{aligned}
\mathbf{M} &= \tanh(\mathbf{W}^y\mathbf{Y} + \mathbf{W}^h\mathbf{h}_N \otimes \mathbf{e}_L) & \mathbf{M} &\in \mathbb{R}^{k\times L} \\
\alpha &= \mathrm{softmax}(\mathbf{w}^T\mathbf{M}) & \alpha &\in \mathbb{R}^{L} \\
\mathbf{r} &= \mathbf{Y}\alpha^T & \mathbf{r} &\in \mathbb{R}^{k} \\
\mathbf{h}^* &= \tanh(\mathbf{W}^p\mathbf{r} + \mathbf{W}^x\mathbf{h}_N) & \mathbf{h}^* &\in \mathbb{R}^{k}
\end{aligned}
$$

- Attention mechanism presented in Rocktaschel et al. https://arxiv.org/pdf/1509.06664.pdf

Evaluation

- FNC dataset is divided into training set (60%), development set (20%) and test set(20%).
- Since data is skewed, ‘Computation Score’ for the algorithm is computed for [HEADLINE, BODY TEXT] pair as follow:

❑ 0.25 increment in score for a correct prediction of unrelated pair
❑ 0.25 increment in score for a correct prediction of related pair.
❑ 0.75 increment in score if pair is correctly predicted as agrees, disagrees or discusses.

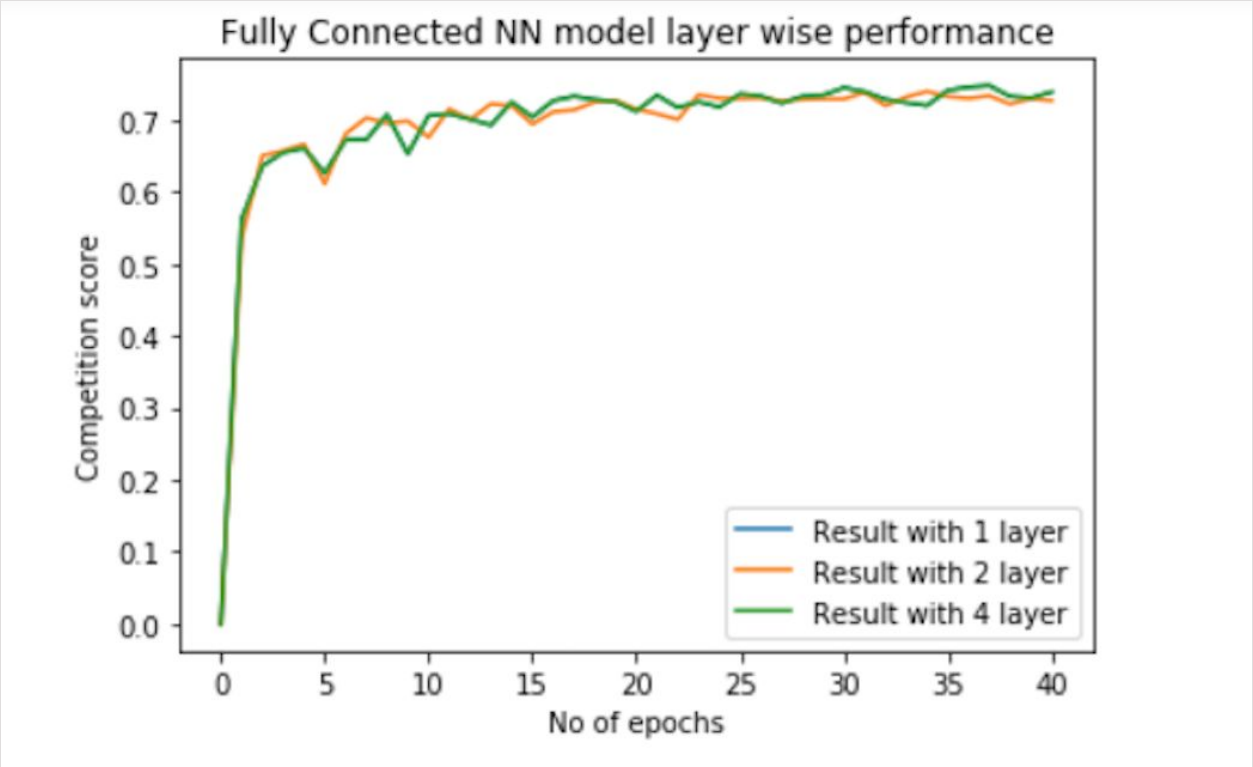
RESULTS

- Table for competition scores and F1 scores obtained from each of the three models: Feed-forward network, LSTM, LSTM with attention for the four classes: Agrees, Disagrees, Discuss, Unrelated are shown below:

Models	Competition Score	Agrees	Disagrees	Discuss	Unrelated
Feed-forward network	0.7458	0.631502	0.4637931	0.806563	0.965368
LSTM	0.781332	0.783748	0.561194	0.89695	0.9687995
LSTM with attention	0.79501	0.833917	0.72327	0.91811	0.97327

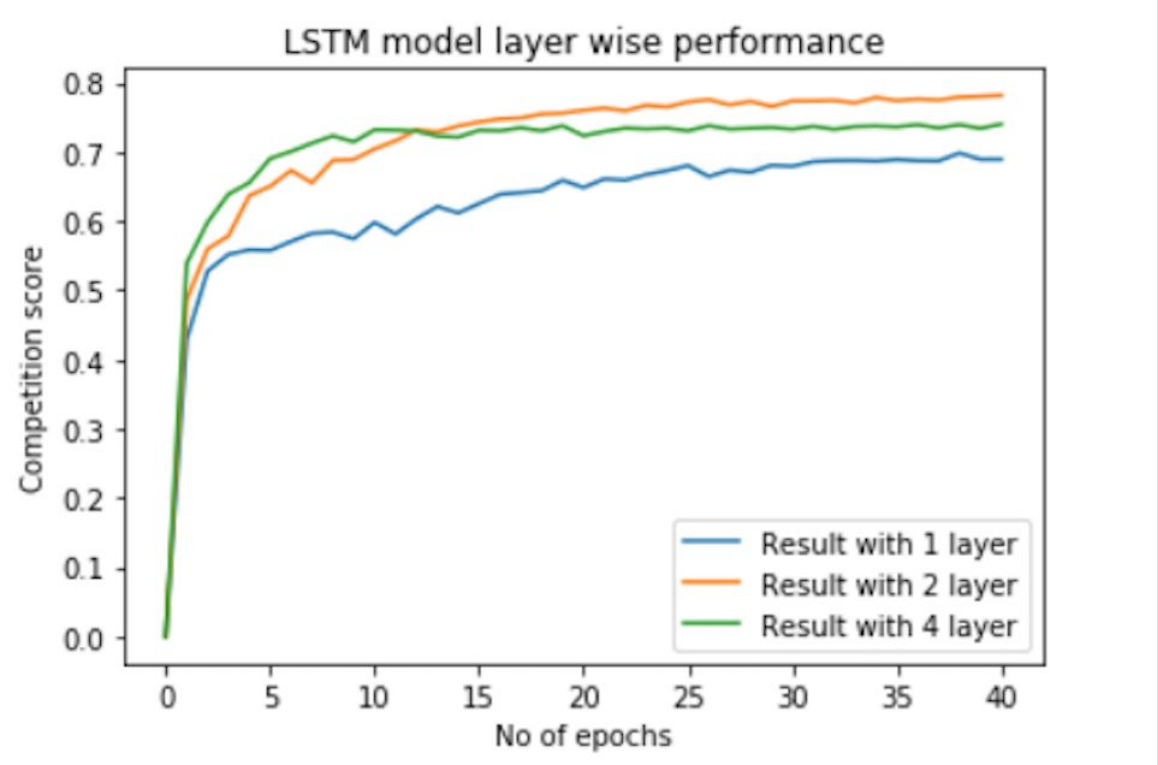
RESULTS

- Competition scores with varying number of hidden layers



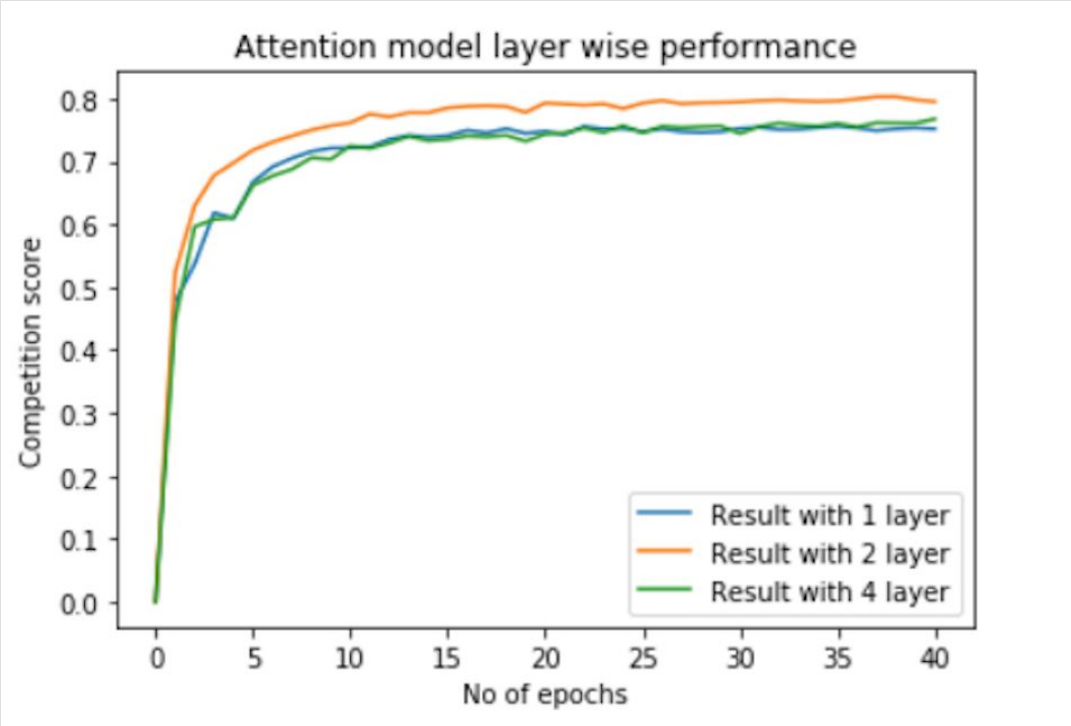
RESULTS

- Competition Scores with LSTM layers.



RESULTS

- Competition scores for LSTM with attention.



DISCUSSION

- Nice performance was observed with 40 epochs and 2 hidden layers in the above models.
- Attention model outperforms basic LSTM model and feed forward Neural Network.
- Score didn’t improve further with 4 LSTM layers, maybe cause of limited data and less number of epochs for training.

DISCUSSION

- Score improves till sequence length of 100 tokens. Beyond that score show no improvement.

- Long sequences are difficult to learn from for LSTM models with limited layers.

- Examples for “disagrees” labels were less and basic neural network and LSTM performed poorly with them. LSTM with attention was able to show significant improvement in identifying headline-body pairs that disagrees.

FUTURE WORK

- Bidirectional LSTM model is widely used for providing more insights from context learned by traversing from the forward and reverse direction.

- Bidirectional LSTM with attention can be tried to determine if there is any significant improvement in performance by learning context from upcoming words.

THANK YOU!

Toxic Comment Classification

CSCE 689: Natural Language Processing

Mayank Sharma **Rashmi Ojha**

Department of Computer Science and Engineering
Texas A&M University

Outline

1. Motivation
2. Problem Overview
3. Dataset
4. Baseline
5. Approaches
6. Architecture
7. Evaluation Metric
8. Comparison with other approaches
9. Results
10. Demo

Motivation

- Kaggle Challenge by Jigsaw
- The ‘Conversation AI’ team works on improving online discourse.
- The personal attacks and derogatory remarks on discussion forums and social media cripples the zeal of users to express themselves openly and seek others’ opinions.
- Aim is to develop a healthy and abuse free discussion environments.
- The Big Question:
  How to identify Negative Online Behavior?

Problem Overview

Model the toxicity in user comments and calculate the regression probabilities for each comment under the following labels:

- Toxic
- Severe toxic
- Obscene
- Threat
- Insult
- Identity Hate

A multi-label classification problem.

Dataset

I. Comments from Wikipedia Talk pages.
II. Each comment contains:
    A. Unique Comment Id
    B. Review Text
    C. Binary values for the 6 labels:

       Toxic, Severe Toxic, Obscene, Threat, Insult, Identity Hate
III. Pre splitted Train - Test set.

Test set required probability values instead of binary classification for each label.

Baseline

Perspective API.

- Model Architecture:
  - Logistic Regression
  - Multi Layer Perceptron
- Used word and character n-grams.
- We chose the best AUC score obtained from an individual model as our baseline, which was 0.9659 using MLP.

Our Approaches

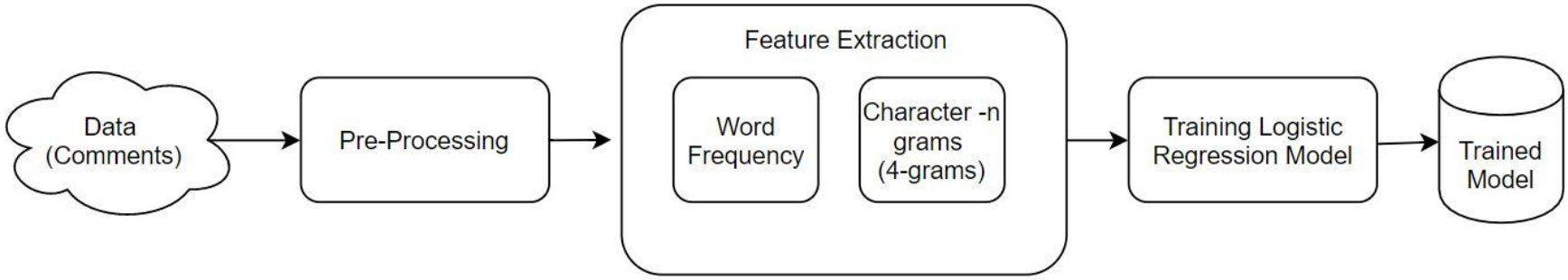
Category	Feature Name	Description
Tf-Idf	Word Frequency	Top 20k TF-IDF features considering each comment as a document
n-grams	Character Frequency	Top 30k Character 4-grams features.
Part of Speech Tags	Noun count *	Count of all Nouns occurring in the text.
	Adjective count *	Count of all Adjectives occurring in the text.
	Verb count *	Count of all Verbs occurring in the text.
Bad-Words	Bad Word Count *	Count of all words present in bad-bad-words' Kaggle data-set.
Symbols	Filler/Masking symbols *	Count of all Foul-filler symbols.
	User Mentions *	Number of times another user is mentioned
	Smileys *	Count of all the smileys (specific pattern of symbols showing emotions)
	Exclamation Mark *	Count of Exclamation Mark
	Question Mark *	Count of Question Mark
	Punctuation *	Count of all punctuation
	All Symbols *	Count of all Unicode(utf-8) symbols
Text Features	Total Words	Total number of words in the comment
	Text Length	Length of comment text same as total character count
	Normalized Word Count	Word Count divided by Text length
	Capital Words *	Count of capital words
	Paragraphs *	Number of paragraphs in comment
	Stop Words *	Count of Stop Words present in NLTK corpus
	Unique Words *	Count of unique words present in the text
	Repetitive Words *	Count of words repeating 10 or more times

1. **Feature Extraction**
   a. Word TF-IDF
   b. Character 4-grams
   c. Part of Speech Tags
   d. Bad Words Dictionary
   e. Symbols
   f. General Text features

2. Experiments with Deep Learning models
   a. LSTM
   b. Bidirectional LSTM
   c. GRU with GloVe Embeddings

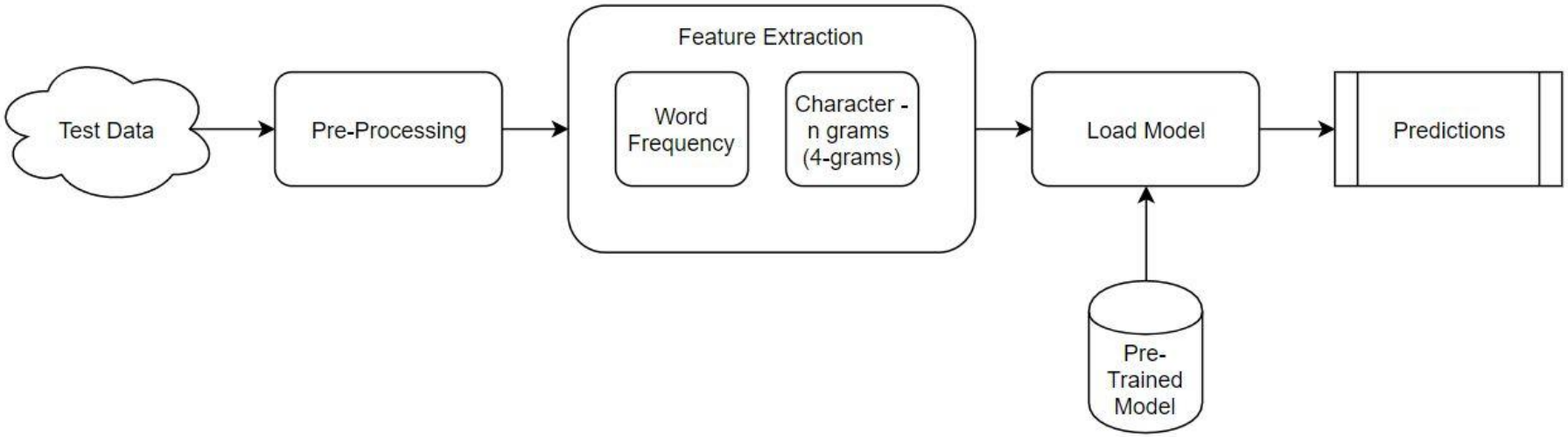
Architecture

Training a model



Architecture

Predicting probabilities



Evaluation Metric

Primary Metric:

Area Under Curve of Receiver Operating Characteristics (ROC-AUC).

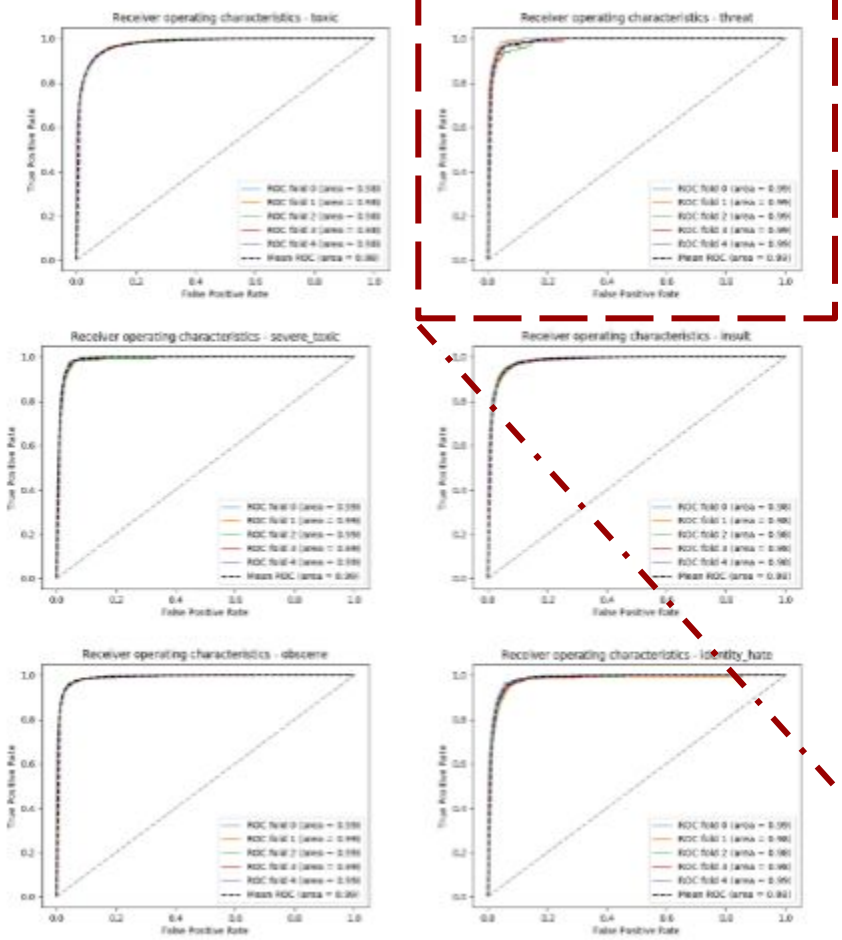
ROC-AUC values were obtained by submitting the predicted test results on Kaggle website.

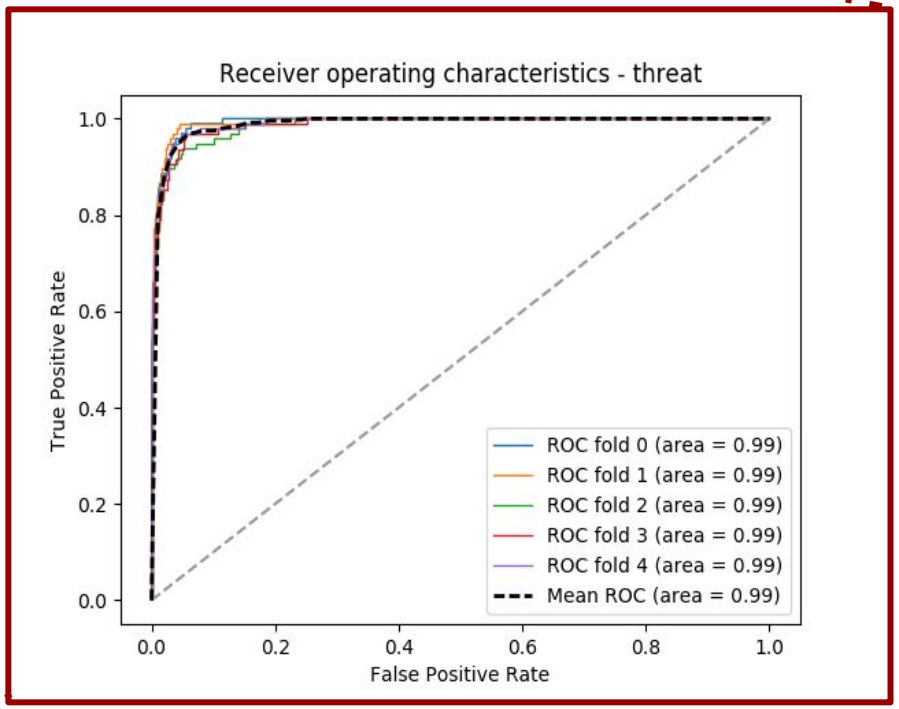
Also calculated 5 fold cross validation scores over the training set.

Comparison of different approaches

Approach	ROC AUC Score
Perspective API (Baseline using MLP)	0.9659
All features + Logistic Regression	0.7408
LSTM	0.9694
Bidirectional LSTM	0.9723
GRU + GloVe	0.9756
Word and Character features + Logistic Regression	**0.9805**

Results


Receiver operating characteristics - toxic
Receiver operating characteristics - threat
Receiver operating characteristics - severe_toxic
Receiver operating characteristics - insult
Receiver operating characteristics - obscene
Receiver operating characteristics - identity_hate


Receiver operating characteristics - threat
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 0 (area = 0.99)
ROC fold 1 (area = 0.99)
ROC fold 2 (area = 0.99)
ROC fold 3 (area = 0.99)
ROC fold 4 (area = 0.99)
Mean ROC (area = 0.99)

Demo

Questions???

Let’s proceed to the Demo...

Hate Speech Recognition

By-
Manish Singhal

Mentor -
Ruihong Huang

Google

Hate Speech

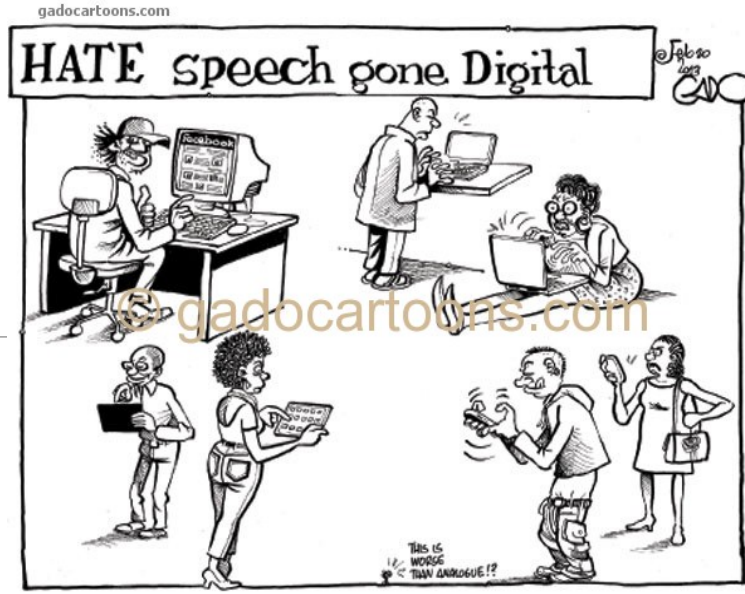
Our products are platforms for free expression. But we don't support content that promotes or condones violence against individuals or groups based on race or ethnic origin, religion, disability, gender, age, nationality, veteran status, or sexual orientation/gender identity, or whose primary purpose is inciting hatred on the basis of these core characteristics. This can be a delicate balancing act, but if the primary purpose is to attack a protected group, the content crosses the line.

facebook
hatebook?

twitter

As explained in the Twitter Rules,

- **Hateful conduct:** You may not promote violence against or directly attack or threaten other people on the basis of race, ethnicity, national origin, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease. We also do not allow accounts whose primary purpose is inciting harm towards others on the basis of these categories.



facebook.

Hate speech

Facebook does not permit hate speech, but distinguishes between serious and humorous speech. While we encourage you to challenge ideas, institutions, events, and practices, we do not permit individuals or groups to attack others based on their race, ethnicity, national origin, religion, sex, gender, sexual orientation, disability or medical condition.

Problem Definition

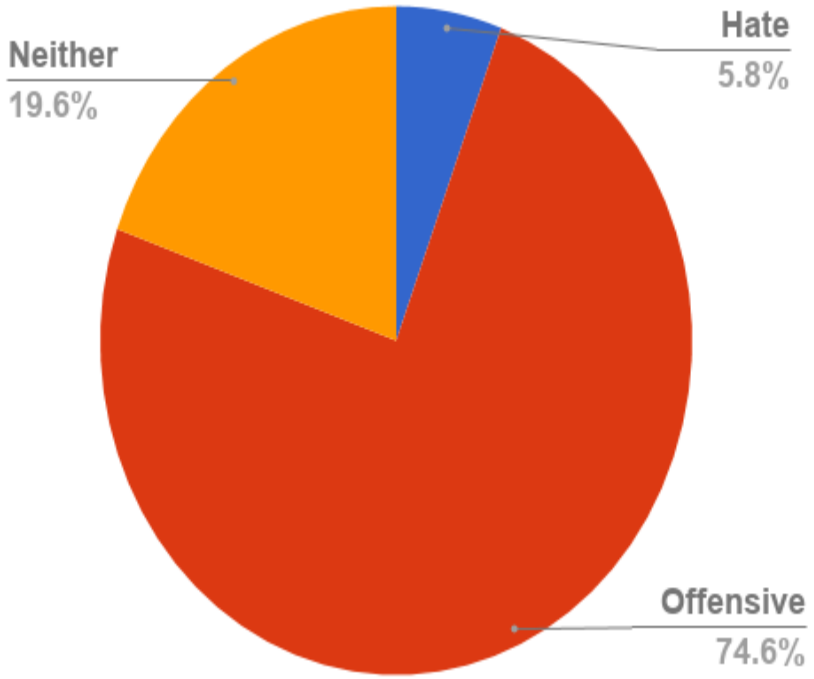
- Hate Speech Recognition is a classification problem aimed at classifying the text into –
  - Hate Speech
  - Offensive
  - Neither
- This text can be anything from a Facebook post to a tweet
- The problem is hard as
  - No specific definition of hate speech
  - Number of hate speech examples are less than what is needed to train a model fully
  - The distinction between hate speech and offensive is blurry (may depend from community to community)

Motivation

- Increasing number of people of social media
- Users sometime are so vocal about their opinion that they do not pay heed to the consequences of what they communicate
- Harmful communication of social media can have severe consequences including unrest and riots
- Hate speech recognition has many critical applications –
  - controversial event extraction
  - building AI chatterbots
  - content recommendation
  - sentiment analysis

Approach: Dataset

- Crowdflower Hate speech identification dataset
- Contains about 24783 tweets classified into 3 categories -
  - Hate – 1430
  - Offensive – 18500
  - Neither – 4853
- Tweets are classified by crowd workers
  - A tweet is classified by minimum of 3 people
  - The majority class is taken as the final class

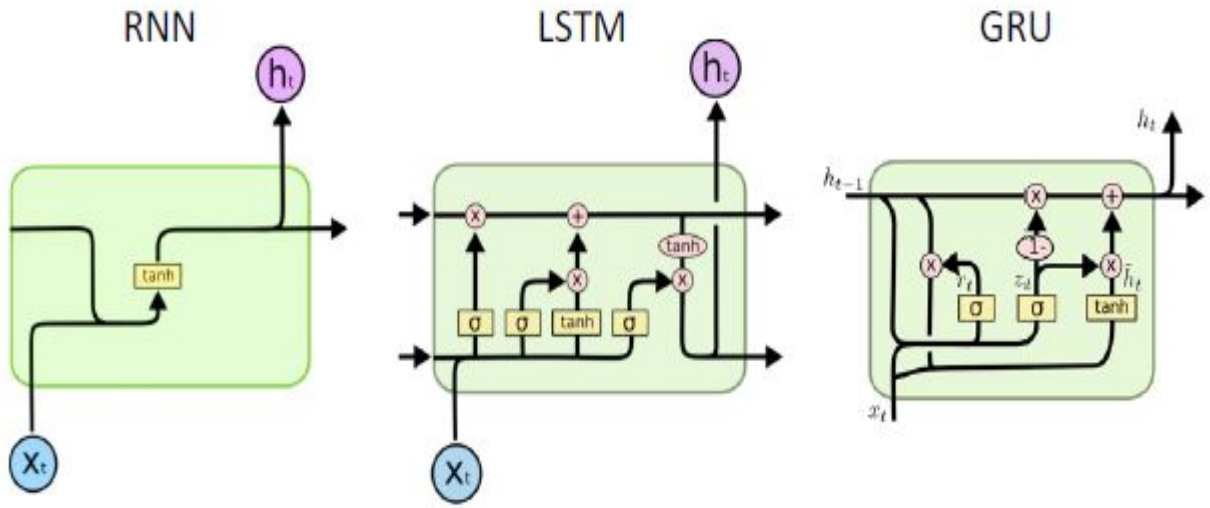


Approach: Pre-processing

- Removed hashtags, URLs, usernames (Tweet specific)
- To counter the high skewness of data, thought of two ways –
  - Make a uniform dataset (by replicating or removing the data)
  - Take into consideration the “class weights” – the error made on wrong classification of hate tweet is more than the error made on wrong classification of neither tweet
- Used stratifying sampling to divide data –
  - 80% data - training
  - 10% data – validation
  - 10% data – testing

Approach: Models

- Baseline Model –
  - Multi class SVM with n-gram (1,3) features
- Deep learning Models –
  - Recurrent Neural Network (RNN)
  - Long Short Term Memory (LSTM)
  - Gated Recurrent Unit (GRU)

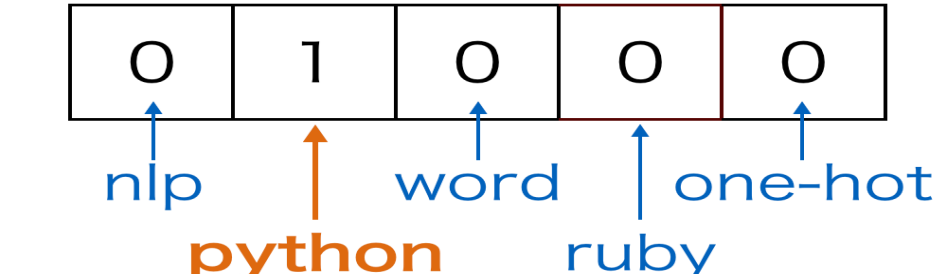


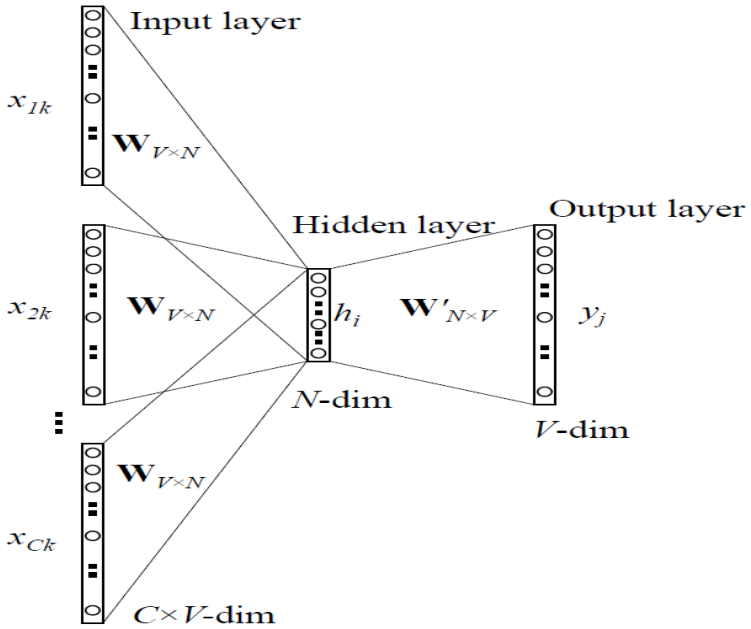
Learning Algorithms and Models

- **Recurrent Neural Network:**
  - RNNs have feedback loops in the recurrent layer which lets them maintain information in memory over time
- **Long Short Term Memory:**
  - It is difficult to capture long-term temporal dependencies using RNN than in LSTM. Hence, LSTM is used to capture these dependencies in the tweets, which may play a vital role in hate speech detection.
- **Gated Recurrent Unit :**
  - GRUs are similar to LSTMs, but use a simplified structure. This simplification can lead to better generalization and performance

Approach: Input Representations

- One hot vector Representation
- Pretrained Word2Vec (genism)
- Pretrained Word2Vec into trainable Embedding layer





Implementation details

- Environment
  - Python - 3.6
  - Keras with Tensorflow backend
  - scikit-learn library
- Experimented by changing different **hyper parameters** like percent of Dropout, L1/L2 regularization, size of sentence embedding and number of nodes/layers in the final classifier (MLP)
- Evaluated the results by based on heat map(confusion metrics) containing class level metrics-
  - Precision
  - Recall (R)
  - F1-measure

System Architecture

Results

Model	Class	Precision	Recall	F1 Score
SVM	Hate	0.17	**0.57**	0.26
	Offensive	0.95	0.73	0.82
	Neither	0.66	0.86	0.75
	Avg/Total	0.66	0.86	0.75
RNN	Hate	0.17	0.61	0.27
	Offensive	0.95	0.73	0.82
	Neither	0.70	0.86	0.77
	Avg/Total	0.87	0.74	0.78
LSTM	Hate	0.19	**0.66**	0.29
	Offensive	0.96	0.76	0.85
	Neither	0.76	0.85	0.80
	Avg/Total	0.88	0.77	0.81
GRU	Hate	0.15	0.66	0.24
	Offensive	0.92	0.66	0.77
	Neither	0.67	0.74	0.71
	Avg/Total	0.84	0.68	0.73

Table 1: **Model Performance**

- Baseline model SVM has a recall of 57 % for hate class.
- As evident from the table, the deep neural networks perform much better than the SVM baseline.
- This can be explained using the fact that SVM is a linear model which is not able to capture all the complex features of the language.

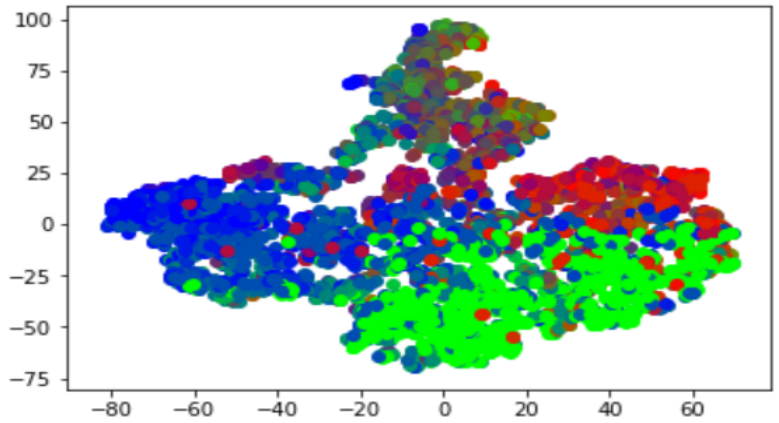
Results(continued)

Model	Class	Precision	Recall	F1 Score
SVM	Hate	0.17	**0.57**	0.26
	Offensive	0.95	0.73	0.82
	Neither	0.66	0.86	0.75
	Avg/Total	0.66	0.86	0.75
RNN	Hate	0.17	0.61	0.27
	Offensive	0.95	0.73	0.82
	Neither	0.70	0.86	0.77
	Avg/Total	0.87	0.74	0.78
LSTM	Hate	0.19	**0.66**	0.29
	Offensive	0.96	0.76	0.85
	Neither	0.76	0.85	0.80
	Avg/Total	0.88	0.77	0.81
GRU	Hate	0.15	0.66	0.24
	Offensive	0.92	0.66	0.77
	Neither	0.67	0.74	0.71
	Avg/Total	0.84	0.68	0.73

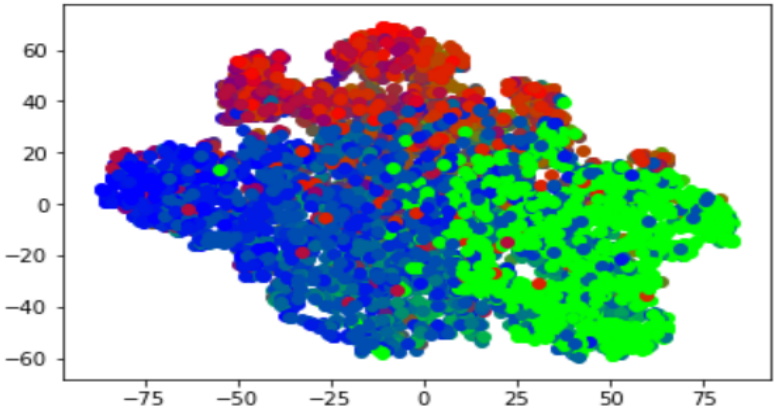
Table 1: **Model Performance**

- Among deep models, LSTM and GRU have almost similar performance (recall of 66% for hate).
- This is in sync with the empirical evaluations of these two models.
- The performance (recall of Hate class) of RNN is comparatively less than deep gate-based models (GRU and LSTM) which can be attributed to the fact that it is not able to capture the long-range dependencies in the tweets as efficiently.

Feature Analysis



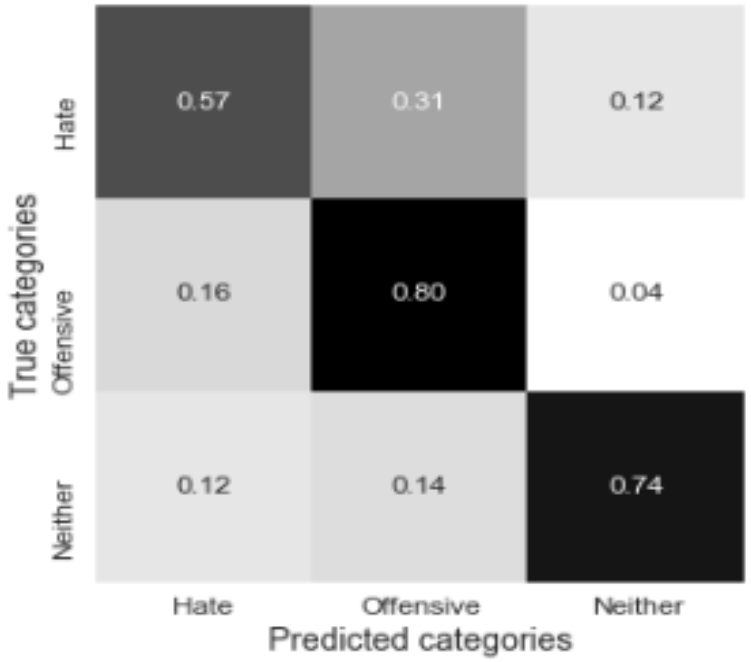
(a) LSTM with one-hot



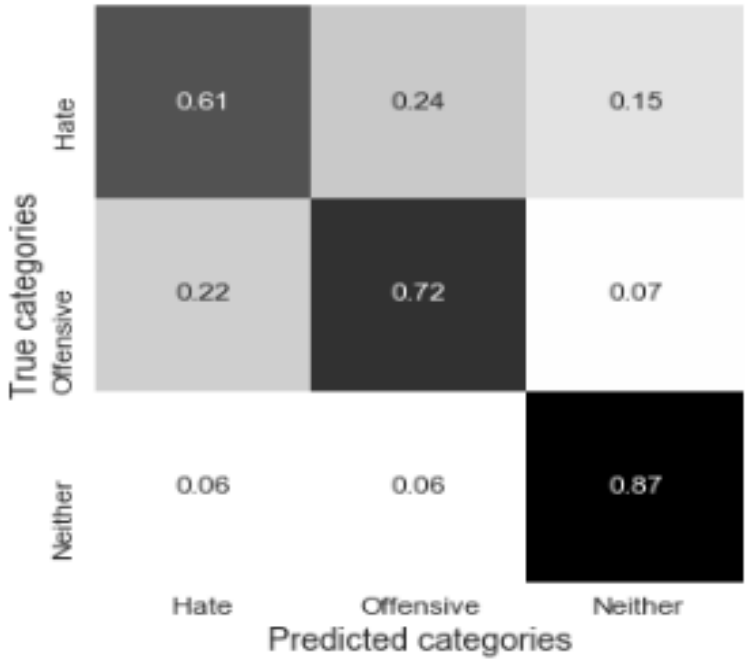
(b) LSTM with trainable word2vec

- Plotted the weights of a feature (word) in the embedding layer in a 2d scatter plot
  - Converted the weights to 2d using **TSNE** transformation.
  - Hate – Red
  - Offensive – Green
  - Neither – Blue
- The weights in case of trainable word2vec form clear cluster boundaries implying that the feature weights are learned in a way that similar features are close to each other.
- However, for 1-hot, the weights cannot be separated that clearly as the initial points are not that good.

Heat Maps

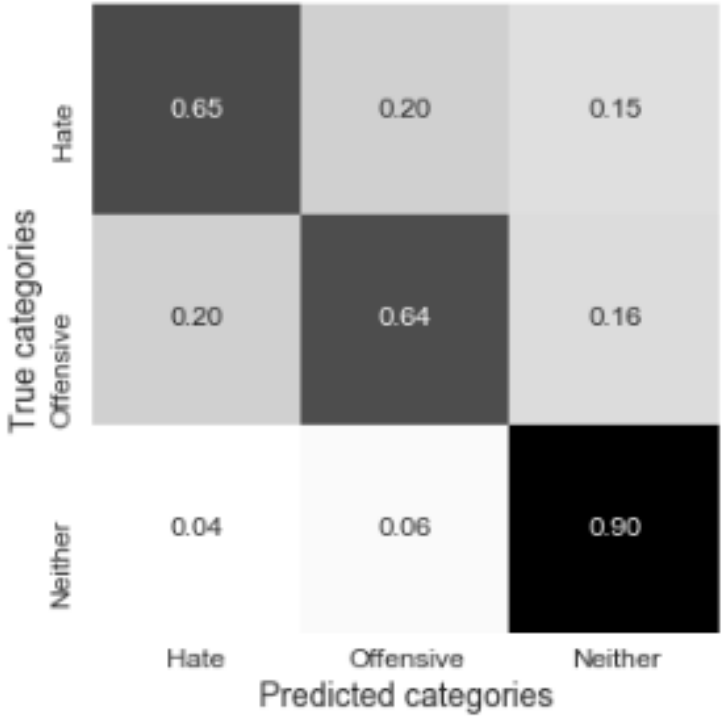


(a) Support Vector Machine

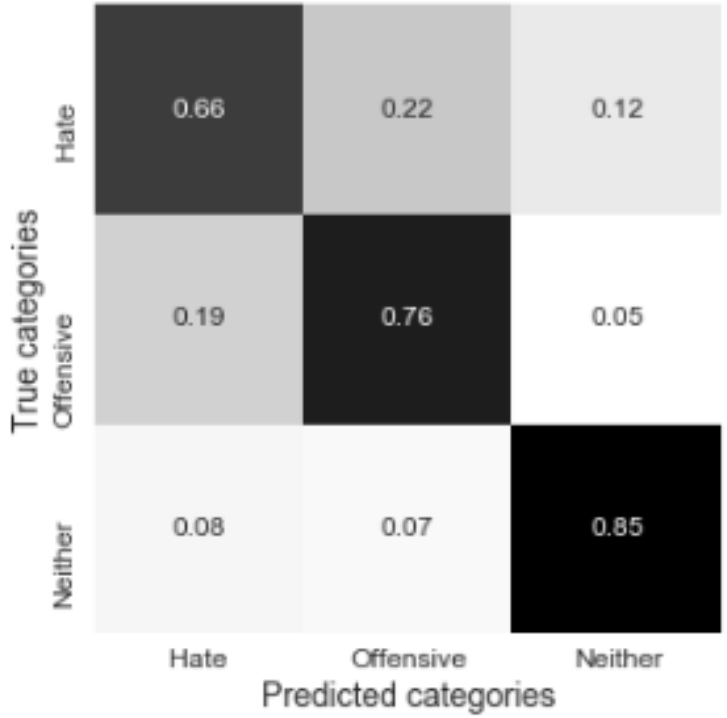


(b) Recurrent neural network

Heat Map



(c) LSTM with 1-Hot input



(d) LSTM with Trainable Word2Vec

Conclusion

- Deep neural models are able to capture complex features and learn their input representation based on the specified task. Hence, they tend to perform significantly better than the linear models.
- The best performing models - LSTM and GRU achieve high recall for hate class, although their F1 scores are dropped due to the conflation of hate and offensive texts.
- It is observed that the broad definition of hate speech, or no commonly accepted difference between offensive and hate speech words worsens the situation.
- During the experiments, I also reinforced the theoretical concept of improving the model performance by using pre-trained feature embedding's (Word2Vec) in a practical scenario like Hate Speech Classification.

References

- P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion, pages 759–760, 2017.
- T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. In Proceedings of the 11th Conference on Web and Social Media. AAAI, 2017
- J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In NIPS 2014 Deep Learning and Representation Learning Workshop, 2014.
- BBCNews. Countering hate speech online, Last accessed: July 2017, http://eeagrants.org/News/2012/.

Thank You!

Detection of Abusive Language in Social Media

Diptanil Chaudhuri

UIN: 825006673

April 24, 2018

Table of contents

Motivation

- Social media has become an integral part of the day-to-day life of modern community. But unfortunately, online abuse and harassment have seen a surge in the recent decade.

Motivation

- Social media has become an integral part of the day-to-day life of modern community. But unfortunately, online abuse and harassment have seen a surge in the recent decade.
- Recent cases show the profound impact it has had on the society.
  - In 2013 Facebook had come under a lot of criticism for hosting pages with abusive names.
    - *Violently Raping Your Friend Just for Laughs*
    - *Kicking your Girlfriend in the Fanny because she wont make you a Sandwich,*
  - In 2017, an Indian student received *"rape threats"* on Twitter for supporting a social media campaign.
  - Organizations such as ISIS and al Qaida are using hate speech on social media platforms.

Objective

The objective of this project is to automate the detection of abusive language in social media networks.
Not only that, the project further sub-classifies abusive languages as hate speech and offensive languages.

Objective

The objective of this project is to automate the detection of abusive language in social media networks.
Not only that, the project further sub-classifies abusive languages as hate speech and offensive languages.
Generally, racist and homophobic slurs are viewed as hateful whereas, sexist slurs are viewed as offensive.

Objective

The objective of this project is to automate the detection of abusive language in social media networks.
Not only that, the project further sub-classifies abusive languages as hate speech and offensive languages.
Generally, racist and homophobic slurs are viewed as hateful whereas, sexist slurs are viewed as offensive.
The project algorithm uses supervised learning techniques to labels tweets into three categories: hate speech, offensive language or neither.

Table of contents

Data

- The data consists of about24,800 labeled tweets containing terms from the hate speech lexicon by `Hatebase.org`.
- The data was manually labeled by CrowdFlower (CF) workers.

Data

- The data consists of about24,800 labeled tweets containing terms from the hate speech lexicon by `Hatebase.org`.
- The data was manually labeled by CrowdFlower (CF) workers.
  - The workers were asked to classify tweets into each of the following categories: hate speech, offensive language and neither.
  - The workers were provided with the definition of hate speech and paragraph explaining the difference between hate speech and offensive language.
  - The workers were asked to take into consideration the overall content, and not just the words, to decide the class of each tweet.

Description

	count	hate_speech	offensive_language	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
5	3	1	2	0	1	!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...
6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up ...

- `count` = number of CrowdFlower users who coded each tweet (min is 3, sometimes more users coded a tweet when judgments were determined to be unreliable by CF).
- `hate_speech` = number of CF users who judged the tweet to be hate speech.
- `offensive_language` = number of CF users who judged the tweet to be offensive.
- `neither` = number of CF users who judged the tweet to be neither offensive nor non-offensive.
- `class` = class label for majority of CF users. 0 - hate speech 1 - offensive language 2 - neither

Table of contents

Features

1. All the tweets were converted into lower-case and stemmed using Porter Stemmer. The output of the Stemmer to create unigram, bigram, and trigram feature, each weighted by it's TF-IDF.
2. To capture the syntactic structure, NLTK was used to construct Penn POS tag unigrams, bigrams, and trigrams.
3. To capture quality of the tweet, Flesch-Kincaid Grade Level and Flesch Reading Ease score was used.
4. VADER Sentiment Analysis was used to assign sentiment score to the tweets.
5. Other features

Porter Stemmer

- The Porter Stemming algorithm is used for removing the suffixes from words in English.
- Assumption that we do not have a stem dictionary and the purpose of the algorithm is to improve the performance.
- The algorithm is given an explicit list of suffixes, and, with each suffix, the criteria under which the suffix may be removed to obtain the word stem.
- Advantage : simplicity and speed

Term frequency-inverse document frequency

Numerical statistic method that is intended to reflect the impotence of a word in a document in a corpus. TF-IDF is the product of two statistics, term frequency and inverse document frequency.

Term frequency-inverse document frequency

Numerical statistic method that is intended to reflect the impotence of a word in a document in a corpus. TF-IDF is the product of two statistics, term frequency and inverse document frequency.
The **term frequency**, denoted $tf(t, d)$, is the raw count of a term in a document, that is, the number of times the term t appears in a document d devided by the total number of words in the document.

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{1}$$

where:

- $f_{t,d}$: number of times term t appears in documents d.

Term frequency-inverse document frequency

Numerical statistic method that is intended to reflect the impotence of a word in a document in a corpus. TF-IDF is the product of two statistics, term frequency and inverse document frequency.
The **term frequency**, denoted $tf(t,d)$, is the raw count of a term in a document, that is, the number of times the term t appears in a document d devided by the total number of words in the document.

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{1}$$

where:

- $f_{t,d}$: number of times term t appears in documents d.

The **inverse document frequency** is a measure of how much information is conveyed by by a word, that is, how common is the word across all documents.

$$idf(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{2}$$

where:

- N: total number of documents in corpus $N = |D|$
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears.

Term frequency-inverse document frequency

Numerical statistic method that is intended to reflect the impotence of a word in a document in a corpus. TF-IDF is the product of two statistics, term frequency and inverse document frequency.
The **term frequency**, denoted $tf(t,d)$, is the raw count of a term in a document, that is, the number of times the term t appears in a document d devided by the total number of words in the document.

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{1}$$

where:

- $f_{t,d}$: number of times term t appears in documents d.

The **inverse document frequency** is a measure of how much information is conveyed by by a word, that is, how common is the word across all documents.

$$idf(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{2}$$

where:

- N: total number of documents in corpus $N = |D|$
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears.

The **TF-IDF** is then calculated as:

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D) \tag{3}$$

Penn Part-Of-Speech

The Penn treebank POS tag set has 36 POS tags along with 12 other punctuations and special characters.

1. CC	Coordinating conjunction	25. TO	*to*
2. CD	Cardinal number	26. UH	Interjection
3. DT	Determiner	27. VB	Verb, base form
4. EX	Existential *there*	28. VBD	Verb, past tense
5. FW	Foreign word	29. VBG	Verb, gerund/present participle
6. IN	Preposition/subordinating conjunction	30. VBN	Verb, past participle
7. JJ	Adjective	31. VBP	Verb, non-3rd ps. sing. present
8. JJR	Adjective, comparative	32. VBZ	Verb, 3rd ps. sing. present
9. JJS	Adjective, superlative	33. WDT	*wh*-determiner
10. LS	List item marker	34. WP	*wh*-pronoun
11. MD	Modal	35. WP$	Possessive *wh*-pronoun
12. NN	Noun, singular or mass	36. WRB	*wh*-adverb
13. NNS	Noun, plural	37. #	Pound sign
14. NNP	Proper noun, singular	38. $	Dollar sign
15. NNPS	Proper noun, plural	39. .	Sentence-final punctuation
16. PDT	Predeterminer	40. ,	Comma
17. POS	Possessive ending	41. :	Colon, semi-colon
18. PRP	Personal pronoun	42. (	Left bracket character
19. PP$	Possessive pronoun	43.)	Right bracket character
20. RB	Adverb	44. "	Straight double quote
21. RBR	Adverb, comparative	45. ‘	Left open single quote
22. RBS	Adverb, superlative	46. “	Left open double quote
23. RP	Particle	47. ’	Right close single quote
24. SYM	Symbol (mathematical or scientific)	48. ”	Right close double quote

Figure: Penn Part-of-speech tags

Readability

Flesch-Kincaid grade level is a score to represent a sentence as a U.S. Grade level, making it easier for parents, librarians, and others to judge the readability level of various texts.

$$0.39\left(\frac{t_w}{t_s}\right) + 11.8\left(\frac{t_{sy}}{t_w}\right) - 15.59 \tag{4}$$

where:

- t_w: is the total number of words
- t_s: is the total number of sentences
- t_{sy}: is the total number of syllables

Readability

Flesch-Kincaid grade level is a score to represent a sentence as a U.S. Grade level, making it easier for parents, librarians, and others to judge the readability level of various texts.

$$0.39\left(\frac{t_w}{t_s}\right) + 11.8\left(\frac{t_{sy}}{t_w}\right) - 15.59 \tag{4}$$

where:

- t_w: is the total number of words
- t_s: is the total number of sentences
- t_{sy}: is the total number of syllables

Flesch Reading Ease score is used to determine the readability of a document.

$$206.835 - 1.015\left(\frac{t_w}{t_s}\right) - 84.6\left(\frac{t_{sy}}{t_w}\right) \tag{5}$$

Higher score indicate easier readability whereas a lower score suggests that the document is difficult to read.

VADER: Sentiment Analysis

VADER is a simple rule-based model for general sentiment analysis.
Combines lexical features with consideration for five generalizable that embody grammatical and syntactical conventions that humans use when expressing or emphasizing sentiment intensity.
The classification accuracy of VADER in social media domain $(F1 = 0.96)$ which outperforms even individual human raters $(F1 = 0.84)$.

VADER: Sentiment Analysis - RULES

1. Punctuation, namely the exclamation mark (!), increases the magnitude of intensity without modifying the semantic orientation.
2. Capitalization, specifically using ALL-CAPS emphasizes a sentiment-relevant word in presence of non-capitalized words, increases the magnitude of intensity without modifying the semantic orientation.
3. Degree modifiers (also known as *intensifiers, booster words* or *degree adverbs* impact sentiment intensity by either increasing or decreasing the intensity.
4. Contrastive conjugation *"but"* signals shift in sentiment polarity of the text following the conjugation, being dominant.
5. Examining the tri-gram preceding a sentiment-laden lexical feature, the algorithm catches nearly 90% of cases where negation flips the polarity of the text.

Other Features

1. Hashtag counts
2. Mention counts
3. Retweet counts
4. URL's
5. Number of characters
6. Number of words
7. Number of syllables

Table of contents

Model

The project uses a multi-class Logistic Regression with L2 regularization as the model.

Learning Logistic Regression The logistic regression is trained with conditional maximum likelihood estimation. That is, we choose parameters w in such a way that maximizes the log-probability of the label y, given the vector of feature observation X.

$$\hat{w} = \arg\max_{w} logP(y^{(j)}|x^{(j)}) - \alpha R(w) \tag{6}$$

For L2 regularization:

$$R(W) = ||W||_2^2 = \sum_{j=1}^{N} w_j^2 \tag{7}$$

Classification

New data is classified into categories by calculating the probability of the data being in a class c, given the features of the data X. The class label with the highest probability is then selected.

$$\hat{c} = \arg\max_{c \in C} P(c|X) \tag{8}$$

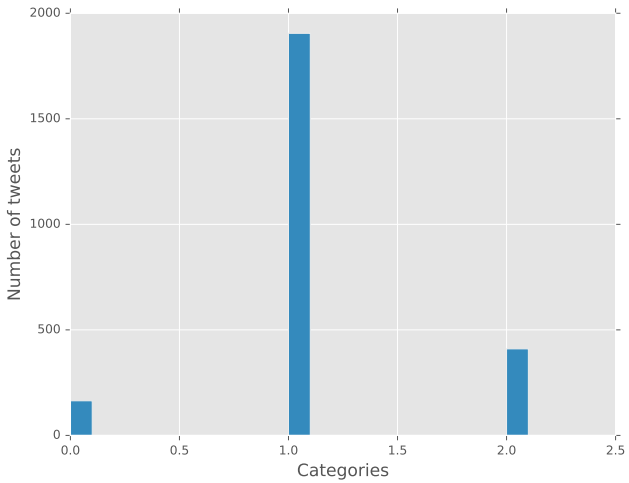
Table of contents

Algorithm

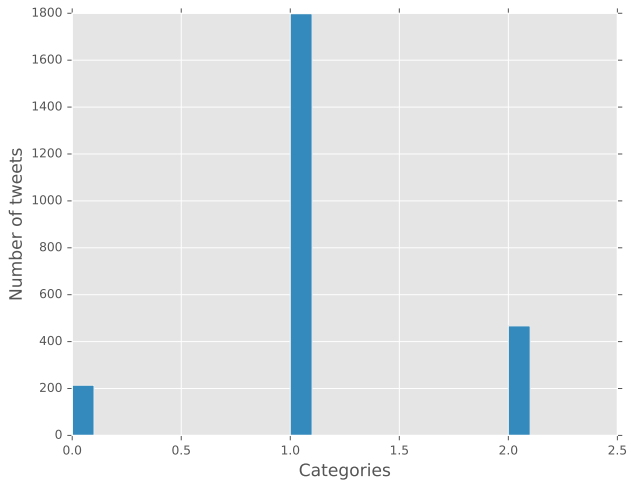
- From the input corpus, extract the features from all the tweets.
- Divide the data into training set and test set.
- Train the model using the features and label from the training set.
- Use the model to predict the category label of the training data.
- Compare the predicted label's and the gold label's, to calculate the precision, recall and F1 score of the model.

Table of contents

Histogram of predicted and true labels



(a) True distribution of test data



(b) Predicted distribution of test data

Figure: Histogram of data

Result

Category	Precision	Recall	F1-score	# cases
0 (Hate speech)	0.44	0.58	0.50	164
1 (Offensive language)	0.96	0.91	0.94	1905
2 (Neither)	0.83	0.94	0.88	410
avg/total	0.91	0.89	0.90	2479

Table: Results on test data

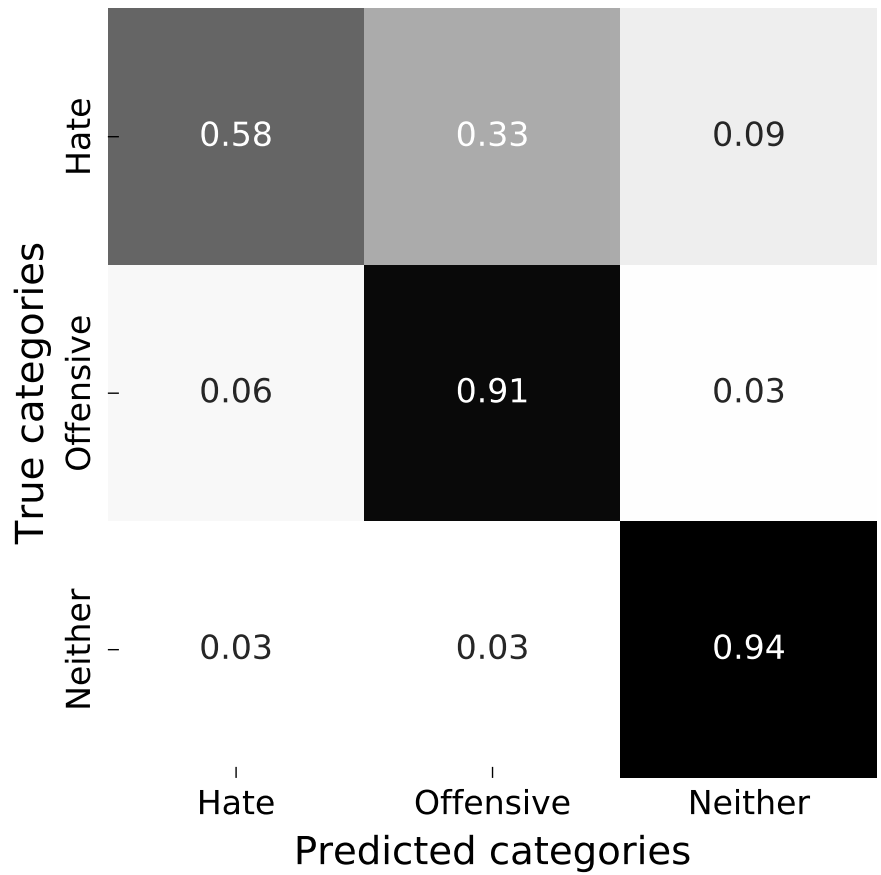
Result

Category	Precision	Recall	F1-score	# cases
0 (Hate speech)	0.44	0.58	0.50	164
1 (Offensive language)	0.96	0.91	0.94	1905
2 (Neither)	0.83	0.94	0.88	410
avg/total	0.91	0.89	0.90	2479

Table: Results on test data

- Overall precision of 0.91, recall of 0.89 and a F1-score of 0.90.
- Precision, recall and F1 of hate speech is 0.44, 0.58, and 0.50 respectively.

Confusion matrix



Analysis

- Most of the misclassification occurs on the upper triangle of the matrix, which suggests that the model is biased towards classifying tweets as less hateful or less offensive then the human coders.

Analysis

- Most of the misclassification occurs on the upper triangle of the matrix, which suggests that the model is biased towards classifying tweets as less hateful or less offensive then the human coders.
- Very few tweets were classified more hateful or more offensive then their true category.

Analysis

- Most of the misclassification occurs on the upper triangle of the matrix, which suggests that the model is biased towards classifying tweets as less hateful or less offensive then the human coders.
- Very few tweets were classified more hateful or more offensive then their true category.
- Only 6% of offensive language, and 3% of non-offensive languages were labeled as "Hate speech", and only 3% of non-offensive language were misclassified as "offensive language".

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs.

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with.

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with. For example: *When you realize how curiosity is a b*tch #CuriosityKilledMe* is not that hateful, and might have been erroneously misclassified as "hate speech" by the human coders.

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with. For example: *When you realize how curiosity is a b*tch #CuriosityKilledMe* is not that hateful, and might have been erroneously misclassified as "hate speech" by the human coders.
- Looking at tweets that were misclassified as hate speech, we find a common feature that most of the tweets contain multiple slurs.

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with. For example: *When you realize how curiosity is a b*tch #CuriosityKilledMe* is not that hateful, and might have been erroneously misclassified as "hate speech" by the human coders.
- Looking at tweets that were misclassified as hate speech, we find a common feature that most of the tweets contain multiple slurs. For example: *My n*gga mister meaner just hope back in the b*tch.*

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with. For example: *When you realize how curiosity is a b*tch #CuriosityKilledMe* is not that hateful, and might have been erroneously misclassified as "hate speech" by the human coders.
- Looking at tweets that were misclassified as hate speech, we find a common feature that most of the tweets contain multiple slurs. For example: *My n*gga mister meaner just hope back in the b*tch.*
- The tweets that were misclassified as neither, tend not to contain any hate or curse words.

Analysis

- Tweets with the highest predicted probabilities of being hate speech tends to contain multiple instances of racial or homophobic slurs. For example: *RT @eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot.*
- Some tweets with true hate speech labels, that were predicted to be classified as offensive language, are not that hateful to begin with. For example: *When you realize how curiosity is a b*tch #CuriosityKilledMe* is not that hateful, and might have been erroneously misclassified as "hate speech" by the human coders.
- Looking at tweets that were misclassified as hate speech, we find a common feature that most of the tweets contain multiple slurs. For example: *My n*gga mister meaner just hope back in the b*tch.*
- The tweets that were misclassified as neither, tend not to contain any hate or curse words.For example: *If some one isn't Anglo-Saxon Protestant, they have no right to be alive in US. None at all, they are forgein filth.*

Table of contents

Conclusion

- Hate speech is a difficult thing to define and is not monolithic.
- Generally, racist and homophobic slurs are viewed as hateful whereas, sexist slurs are viewed as offensive.
- The algorithm is able to achieve an overall precision of 0.91, recall of 0.89 and a F1-score of 0.90.
- The model described in this project can be used to distinguish abusive language among hate speech and offensive, with some errors.

Thank you all

Sentiment Analysis of Twitter Data to Predict Price Fluctuations in Crypto-Currencies

RAHUL SAINI

Natural Language Processing

Final Project

CryptoCurrencies?

- Decentralized
- Anonymized

- Why BITCOIN (BTC)?
  - Transaction Volume
  - Value?

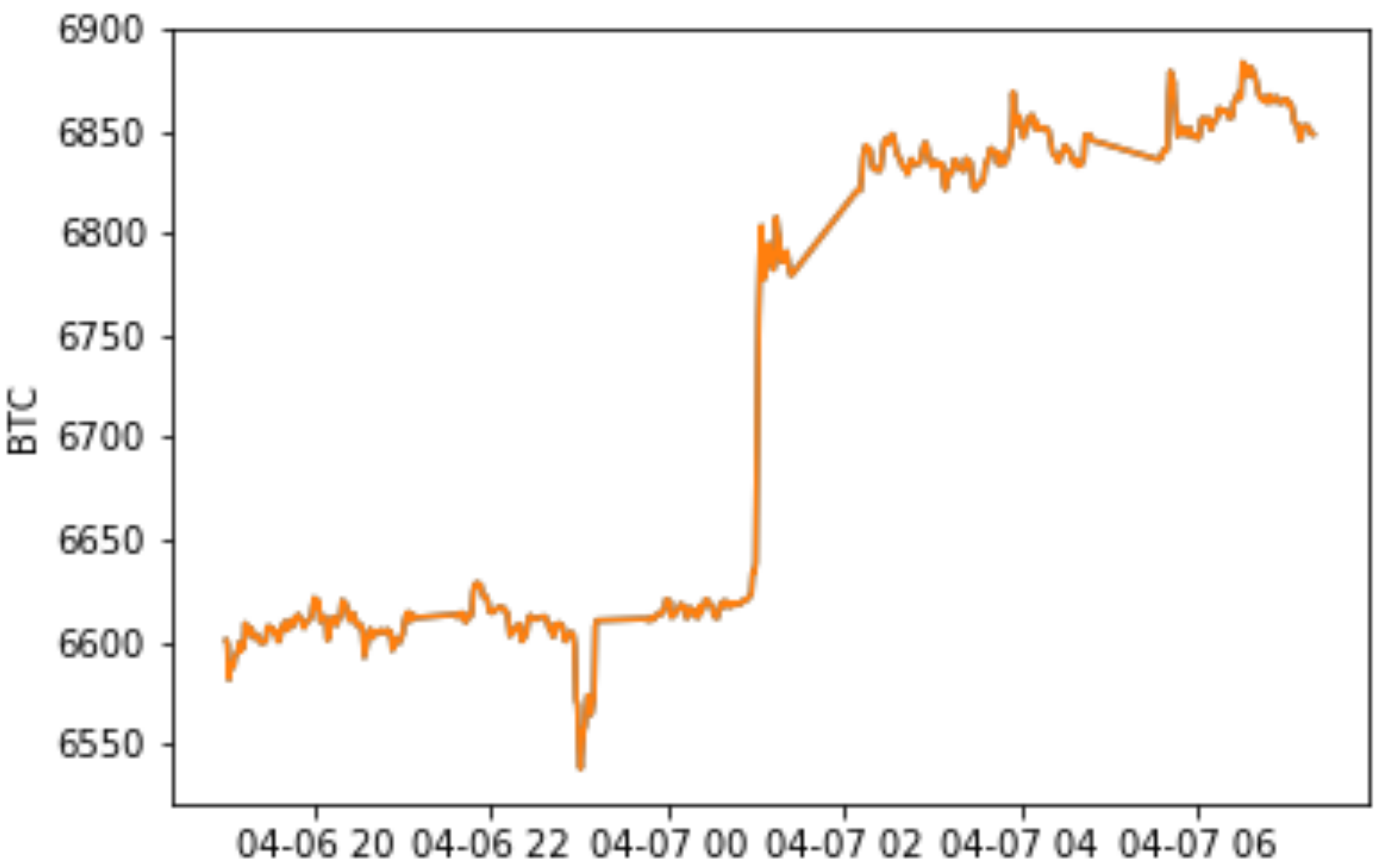
Sentiment Analysis

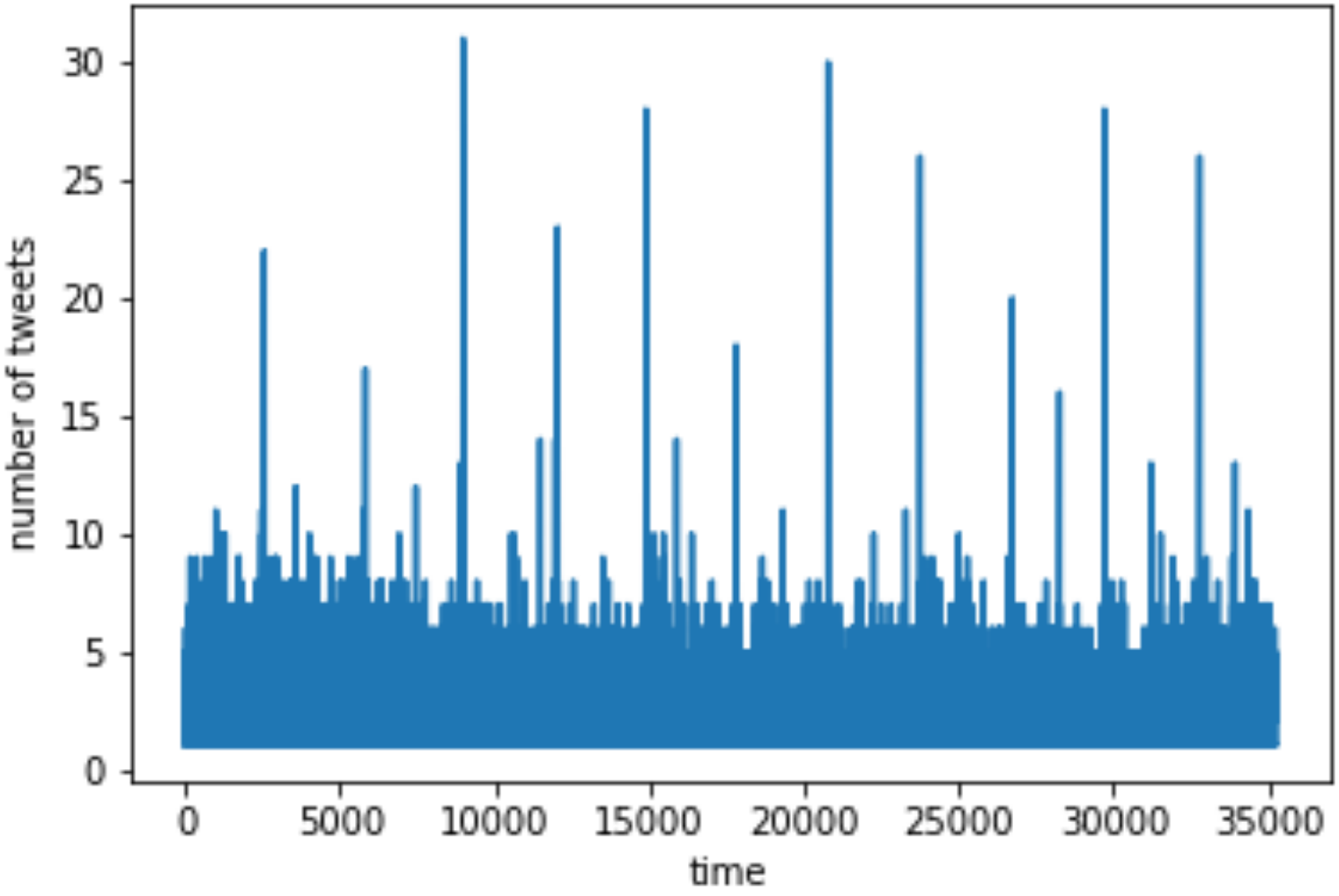
- Why Twitter?
  - Micro-Blogging
  - Bitcoin + Twitter?

Gathering DATA: CHALLENGING!

- Tweepy: Twitter API
- Coindesk: BPI API

- 3 Data Sets
  - ~30000 tweets, 500 BPI
  - ~85000 tweets, 550 BPI
  - ~85000 tweets, 600 BPI


6900
6850
6800
6750
6700
6650
6600
6550
BTC
04-06 20
04-06 22
04-07 00
04-07 02
04-07 04
04-07 06


number of tweets
0
5
10
15
20
25
30
0
5000
10000
15000
20000
25000
30000
35000
time

Preprocessing

- #Pantera Capital Says #Bitcoin Has Hit a Buy Signal https://t.co/hCn2Ijz86e

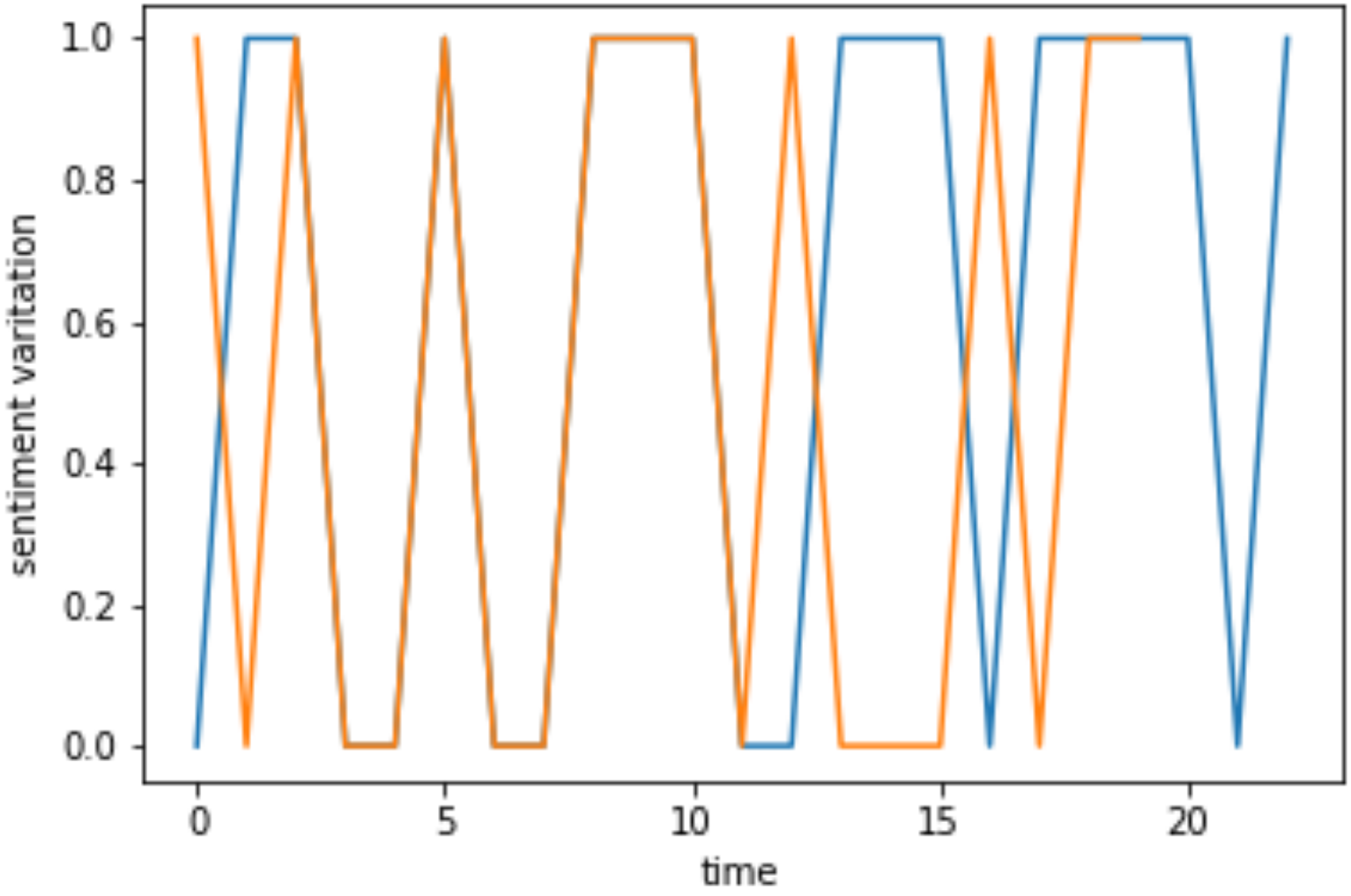
- HASHPantera Capital Says HASHBitcoin Has Hit Buy Signal URL

VADER

VADER is smart, handsome, and funny.---------- {'neg': 0.0, 'neu': 0.254, 'pos': 0.746, 'compound': 0.8316}
VADER is not smart, handsome, nor funny.---------- {'neg': 0.646, 'neu': 0.354, 'pos': 0.0, 'compound': -0.7424}
VADER is smart, handsome, and funny!---------- {'neg': 0.0, 'neu': 0.248, 'pos': 0.752, 'compound': 0.8439}
VADER is very smart, handsome, and funny.---------- {'neg': 0.0, 'neu': 0.299, 'pos': 0.701, 'compound': 0.8545}
VADER is VERY SMART, handsome, and FUNNY.---------- {'neg': 0.0, 'neu': 0.246, 'pos': 0.754, 'compound': 0.9227}
VADER is VERY SMART, handsome, and FUNNY!!!---------- {'neg': 0.0, 'neu': 0.233, 'pos': 0.767, 'compound': 0.9342}
VADER is VERY SMART, uber handsome, and FRIGGIN FUNNY!!!---------
{'neg': 0.0, 'neu': 0.294, 'pos': 0.706, 'compound': 0.9469}

Prepping Data; Prediction Vectors

- Intervals
  - 15 minutes
  - 30 minutes
- [tweet: 0.5; time = x;] [tweet: 0.6; time = y] ….
- Aggregate
- [Aggregate Sentiment Score = 0.6] [Aggregate Sentiment Score = 0.8]
- [1, 0, 1, 1,1]
- Shifts
  - [1,0,1,1,1] [1,1,1,0,1]


1.0
0.8
0.6
0.4
0.2
0.0
sentiment varitation
0
5
10
15
20
time

Evaluation Results

Shift: 1
Number of Predictions: 22.0
True Positives: 6 False Positives: 7
True Negatives: 3 False Negatives: 6
Accuracy: 0.409090909091

Shift: 2
Number of Predictions: 21.0
True Positives: 7 False Positives: 6
True Negatives: 4 False Negatives: 4
Accuracy: 0.52380952381

Shift: 3
Number of Predictions: 20.0
True Positives: 7 False Positives: 5
True Negatives: 5 False Negatives: 3
Accuracy: 0.6

Shift: 4
Number of Predictions: 19.0
True Positives: 6 False Positives: 5
True Negatives: 5 False Negatives: 3
Accuracy: 0.578947368421

Questions?

- Thank YOU! ☺