Hate Speech Detection

Week 9

# Team Member Details

Name: Faris Chaudhry

Email: [faris.chaudhry@outlook.com](mailto:faris.chaudhry@outlook.com)

Country: United Kingdom

University: Imperial College London

Specialization: NLP

# Problem Description

“The term hate speech is understood as any type of verbal, written or behavioural communication that attacks or uses derogatory or discriminatory language against a person or group based on what they are, in other words, based on their religion, ethnicity, nationality, race, colour, ancestry, sex or another identity factor.

Hate Speech Detection is generally a task of sentiment classification. So, for training, a model that can classify hate speech from a certain piece of text can be achieved by training it on a data that is generally used to classify sentiments. We will use the Twitter tweets to identify tweets containing Hate speech.”

# Data Validation

A screenshot of a computer program

Description automatically generatedA screen shot of a computer

Description automatically generated

Validation done:

* Standardise column spaces (lowercase, underscores instead of spaces).
* Remove duplicate tweets, keeping the last one (extra tweets of the same kind don’t benefit model).
* Remove rows with null tweets.
* Remove unlabelled data in training set.
* Rename duplicate or null index by reindexing the whole set (this wasn’t required in practice).

A screenshot of a computer

Description automatically generated

# Feature Extraction

Features extracted:

* Word count
* Average word length (sometimes people forget spaces when typing angrily)
* Hashtags
* Exclamation marks
* Question marks (people ask rhetorical questions when angry at someone e.g., ‘Are you stupid?”)
* All uppercase words (indicates rage)

Correlation was checked on these simple features since if there were multiple features with strong correlation it would only add extra dimensions to the model.

A screenshot of a computer

Description automatically generated

# Data Cleaning

Data cleaning done:

* Standardisation of tweets
  + Make all words lowercase so that model is not case-sensitive (e.g., “Hello” == “hello”)
  + Remove punctuation (although certain punctuation structure might be a predictor for negative sentiments, this is extremely hard to analyse especially since informal sentence structure is used in tweets)
* Noise reduction
  + Remove stop words such as “the”, “is” and “and” which are common and have a neutral sentiment. This reduces the amount of noise in the model and the dimensionality of analysis.
  + Removal of other common words. For the same reason as above, the most common word was @user.
  + Removal of rarely used words (used only once in all of data set). These words add extra noise since there is not a great enough sample to effectively analyse them.
* Spelling correction (Optional)
  + People often make spellings on short-form tweets. A lexicon of words and TextBlob can be used to fix easy to make corrections. However, this can be unreliable because non-standard words and abbreviations of words might occur e.g., ‘u’ instead of ‘you’ would not be automatically corrected. I experimented with this approach, but it is too unreliable and, since language changes quickly on social media, it is untenable to make a long-term model from this.
* Suffix and prefix removal (Optional)
  + The lexicon of analysed words can be reduced by reducing words to their ‘unembellished form’. For example, ‘hating’, ‘hater’, ‘hated’, ‘hate’ is all reduced to ‘hate’. Two approaches to this exist. The first is by removing any instances of a suffix or prefix directly (this is the simpler approach). The other is to use lemmatization to find the root word or conjugate that belongs to a specific word. Both approaches have been tried here although it makes sense to only use one at a time.
* Tokenization
  + After all common, rare, and insignificant words have been removed from tweets, tokenization can be used to create a list of the words that exist in the tweet. This removes duplicates but comes with the advantage that we can later vectorise each word, therefore the tweet, so it can be used as a feature for the model.

# Appendix: Various Code Samples

A screen shot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated