Transgender Support Radar

Using NLP to identify Tweets in support of transgender rights

Milestone Report - Jessica Wulzen

Note - as this project studies language and how it is used in sensitive settings, this writeup contains some vulgar language and terms used as slurs.

Motivation

LGBTQ+ rights have made tremendous strides in recent years. However, some issues, particularly those facing the transgender community, remain contentious. As a result, trasngender people often find themselves in a position of not being sure of if others are accepting, or how they will be treated should they choose to share information about their gender with others. As a reflection of this, online transgender communities often share tweets from celebrities, politicians, and other well-known figures who vocalize their stances on trans rights, doing so either gleefully in response to support, or with derision in response to opposition.

This project will develop a tool using NLP to answer the question: using someone's history of Twitter posts, is that person supportive of or opposed to trans rights?

Approach

Our tool will use the following structure:

1. Take a twitter username from the user
2. Find all tweets from the given user
3. Identify which tweets, if any, are trans-related
4. Perform sentiment analysis on trans-related tweets to classify them into negative/supportive
5. Report findings to the user

For the first two steps, we'll lean on the Python package Twitterscraper. This package helps scrape data from Twitter, making it simple to get the data that we need. Additionally, we'll use this package to gather the data we need to train the machine learning algorithms used for classification.

The third step, identifying trans-related tweets, is not too complex. For the most part, the words and phrases that identify trans-related posts are unambiguous - if a tweet has words and phrases like 'transgender', 'trans man' or 'gender nonbinary', then it is clearly relevant to our project. Thus, building up a vocabulary of trans-related terms to search for gets us most of the way there. However, there are some crucial terms which are not used exclusively in the context we are interested in - 'tranny', a slur used against trans people, shows up in tweets from car enthusiasts as a slang term for 'transmission', and from financial experts who refer to the Dow Jones Transportation Index as 'the trannies'. For this reason, we'll build a classification model to determine if a tweet is transgender-relevant. This model was out of scope for the first milestone report, so presently, a tweet is labeled as trans-related if it has a word out of a list of specific terms. .

The fourth step, identifying whether or not a trans-related tweet is supportive or not, is by far the most difficult part of the process. This is where the vast majority of development time has been spent, and so the majority of this report will focus on how to classify tweets.

Finally, the user will be given an attractive little report, detailing several things. They will be shown the volume of tweets from their given user over time, the distribution of how many of those tweets were trans-related and when those tweets occurred, and several example tweets, so that the user can determine for themselves if they agree with our model's decisions.

In this report, we'll dive into the sentiment classification (are tweets supportive or not), and focus on the process of answering this question. Then, we'll make some remarks on the development of the other aspects of the project.

Data Collection

We'll need as much data our algorithm can learn from as we can get. Unfortunately, in order to tackle this question, the data needs to be labelled as 'supportive' or 'negative', and no such publically available datasets exist (and likely none exist privately, either!). As a result, a huge amount of time was needed to gather trans-related tweets, read them, and label them. Labelling has been done on a scale of 1-5, where 1 means 'extremely hateful' and 5 means 'highly supportive'. On this rating, a '3' represents a tweet which is mostly neutral - ie, a tweet which may be about trans topics but does not contain any particular value judgement. For the purpose of modelling, these 'neutral' tweets are considered to be supportive. Thus, the two categories can be thought of as; anti-trans, and neutral/supportive.

In order to find tweets, a list of trans-related terms was compiled. Tweets containing these terms and phrases were then gathered by searching on these terms with Twitterscraper. Then, I read and scored these tweets. As the scoring began, it became clear that there was a greater volume of supportive tweets than negative tweets, so more 'negative' terms and phrases were added to the search list, so that the data could be more balanced.

One potential issue with this approach is that it relies upon my own personal knowledge of relevant terms and phrases. If there are terms and phrases charged with meaning which I did not collect any data for, then if the algorithm encounters these terms and phrases, it will not be trained on how to handle them. Additionally, language is constantly evolving, especially in these online discussion contexts where terms and acronyms can quickly come into and fall out of favor. To combat this as much as possible, while labelling data, I made a note of any phrases which seemed to frequently appear, and added them to the list of search terms.

A second challenge was capturing certain terms which have multiple meanings, many of which are unrelated to trans topics. For example, the word 'trap' is a slur with negative connotations used towards transgender women - but scraping tweets for the word 'trap' results mostly in unrelated tweets. Adding more terms to the search, such as 'trap + trans' or 'trap + gay' results more in tweets from well-intentioned people discussing how trap is a slur than people who are using it in their common parlance, leading to training data that is not entirely reflective of reality.

Data Preprocessing

Feeding raw tweets directly into an algorithm isn't the best way to handle the data. To help the algorithm as much as possible, there's a variety of data preprocessing steps that we can take. Most of this is fairly standard in NLP applications, such as stripping punctuation (with the exception of hashtags and quotation marks, which will be discussed in more detail), removal of stop words, lemmatization, etc.

One of the most common rallying cries in support of trans people are the phrases 'trans women are women' and 'trans men are men', with their negative corollaries 'trans women aren't women' and 'trans men aren't men'. The appearance of these phrases in a tweet are strong evidence if that tweet should be labelled as supportive or negative. However, the words 'are' and 'aren't' are typically stop words - if we removed all stop words, then the phrases 'trans men are men' and 'trans men aren't men' would both be reduced to 'trans men men', losing critical information. For this reason, we'll use a standard list of stopwords but will remove 'aren't' from that list.

Quotations

One area of particular complexity for our models are quotation marks. In general, many approaches to NLP remove all special characters from the text, to simplify the data as much as possible. However, in this context, quotation marks are surprisingly relevant. There are two common occurrences of meaningful quotation marks. The first is using quotation marks to quote someone, which often is done when the speaker disagrees with the person they are quoting, eg:

My dad just said "trans people are mentally ill". I can't believe how wrong he is!

A second common usage is putting quotation marks to put focus on an individual word or phrase, often to indicate disdain for use of that word. This is frequently used both in pro-trans and anti-trans contexts:

The word "tranny" is a slur - please don't use it.
People saying "trans women are women" are delusional. Learn some basic biology.

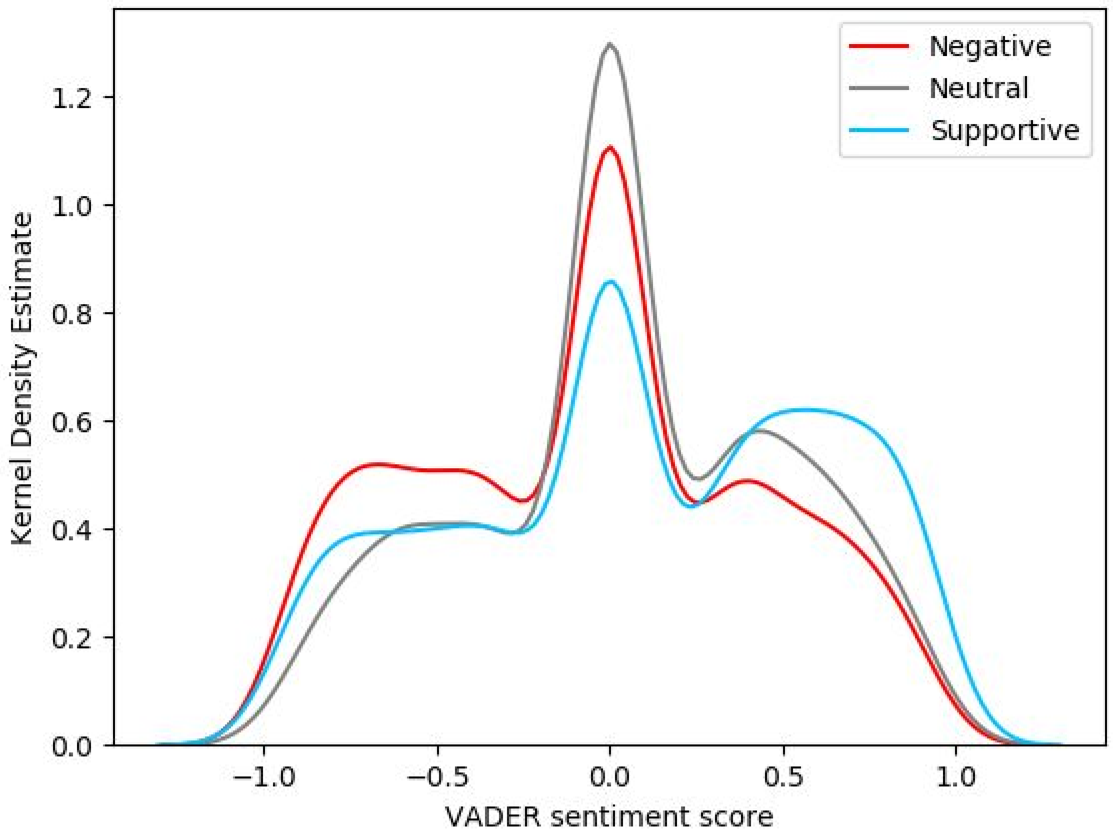
In many of these cases, the quotes indicate that the words within them are somehow different from their normal usage, and by removing these quotes, an important level of meaning is lost. Specifically, using TF-IDF models with Naive Bayes for classification, usage of slurs will strongly indicate that a text is anti-trans. However, well-meaning people will frequently put these slurs in quotation marks to discuss them while making it clear they are not using them. By stripping these quotes, the ability to differentiate this usage will be lost, likely leading to misclassification.

Our implementation of this required a custom function to process quotation marks. Part of the complexity of this problem was centered around multiple word phrases in quotation marks. Using a bag of words or TF-IDF model, it is necessary to be able to denote that every word in quotes - such as "trans women are women" - are in quotes. Thus, our function identifies blocks of text in quotes, and puts each individual word in quotes, turning “trans women are women” into “trans” “women” “are” “women”.

When using a TF-IDF with Naive Bayes model, we can compare the performance of preprocessing that completely removes quotes with our alternate processing method outlined above. Ultimately, it ends up having a small impact on accuracy, improving model performance by about ~1%. It also has the effect of increasing the number of words in our corpus, from 15,200 to 16,500. Ultimately, I would expect that the benefit of this approach would be even more meaningful with a large sample size, as with a body of only 10,000 tweets, further split into test and training data, the opportunities for our ML algorithms to learn the nuance of quoted words may not be fully realized.

VADER

Pre-trained algorithms for sentiment analysis on text exist for out of the box usage. One such package, VADER, computes a score showing both if the sentiment of the text is positive or negative, but also the intensity of the sentiment, to differentiate between text which is extremely positive vs only slightly positive. Below, we compare the distribution of VADER scores for tweets that are transgender-supportive, neutral, and negative:



The supportive and negative tweets have different distributions, seeing as the supportive tweets tend to have more positive VADER scores than negative tweets. However, the difference is not stark enough to use Vader as the cornerstone of our approach. Using only the VADER score in a logistic regression does not do very well, creating a model which has an embarrassing 55% accuracy on test data - only five percentage points better than flipping a coin.

Ultimately, there are trans-supportive tweets which VADER sees as being very negative, and trans-negative tweets which Vader sees as being very positive, such as:

*f*** transphobes! trans women ARE adult human females and if you disagree go ahead and suck my *****

Gender is binary. That's the end of the topic. LGBT people please get help to overcome your twisted minds. God bless you all.

These types of nuanced tweets mean VADER by itself is not reliable for our task, and we'll need to build something more sophisticated.

Model: TF-IDF Vectorizer with Naive Bayes

A widely-used approach to similar problems, a TF-IDF vectorizer takes our corpus of tweets and determines the frequency of every term, while also weighing how many documents (tweets) the term appears in. To capture key multiple word phrases, we'll set up the vectorizer to take ngrams of 1-4 words, so that phrases such as 'trans rights', 'adult human female', and 'trans women are women' can all be properly captured.

After vectorizing the data, we'll use a Naive Bayes model to handle the actual classification. Naive Bayes is widely used in this context because it is excellent at determining the probability that a document belongs to a certain category given which individual words and phrases are in that document.

One glaring weakness of this approach (and most NLP approaches) is that we expect the model to be very weak to sarcasm and tweets in response to things said by others. For example:

Oh sure, trans women are women! And trans men are men! And biology doesn't matter so we can ignore it and I can just decide to be a unicorn.

He literally said that trans people are dangerous predators and they should be rounded up and thrown in jail. He's just disgusting.

Both of these tweets are going to be very difficult for a Naive Bayes approach to handle successfully, because they share something in common; most of the text seems to deliver one message, but humans can clearly see the tone of the message and the critical phrases which flip the meaning of the text on its head. Knowing that these sort of nuances are common in everyday speech, how well does our model do?

Model Evaluation

There are several different metrics we can use to see how our model is performing. A good starting place is the accuracy, or what percentage of test observations our model predicted correctly on. Using test data separate from the data the model was trained on, the Naive Bayes classifier is able to correctly classify 84.7% of tweets - pretty good! We'll look at a few more metrics to get a better sense of how the model is doing.

	Predicted Supportive	Predicted Negative
Actual Supportive	2185	358
Actual Negative	247	1173

	Precision	Recall	F1-Score
Actual Supportive	0.90	0.86	0.88
Actual Negative	0.77	0.83	0.79

From these findings, we can see that the model tends to be a little better about predicting supportive tweets, with better precision and recall compared to negative tweets, but that it handles both types of tweets fairly well.

Another way we can look into the inner workings of the model is to see what words and phrases have the most influence over our model. We can do this by taking every feature in the tfidf vectorized matrix - every word and phrase that appears in our data - and treating them as entire tweets, passing them through the model and recording the predicted probability. Essentially, this means we're predicting the probability that a tweet is supportive, if the entirety of that tweet is just one word from our corpus. Then, we repeat this process for every word in the corpus. From this, we can find which words are the most commonly used by both sides.

Most supportive phrases

Phrase	P(supportive | tweet is phrase)

tranniversary	0.903
nonbinary	0.900
trans men are men	0.897
men are men	0.891
genderqueer	0.884
trans right	0.874
genderfluid	0.862
bisexual	0.855
hrt	0.847
lbtq	0.847

Most negative phrases

Phrase	P(supportive | tweet is phrase)
woman adult	0.053
woman adult human	0.054
woman adult female human	0.054
natal	0.075
trannies	0.076
biological male	0.084
adult human	0.085
natal male	0.087
human female	0.087
adult human female	0.089

These results make a good deal of sense. They tend to be the kind of phrases that are only said by transgender people or in support of them, or to attack trans people. The positive phrases tend to either be about nonbinary people/genderfluid people/trans men - who are often overlooked in anti-transgender sentiment which tends to focus on trans women - or are phrases which tend to only be used by people with a good deal of knowledge about trans topics. On the other side, many of the negative phrases revolve around the phrase “adult human female”, a term which is frequently used by women who wish to create a distinction between themselves and trans women, or around the phrase “biological males”, which is commonly used to delegitimize trans women.

We’ve now seen that the model works fairly well by reviewing several different metrics, and have confirmed that the most extreme terms for both support and negativity make sense using human logic. But can we do better?

Can VADER be useful after all?

Earlier, we saw that VADER’s sentiment analysis had some overlap with the trans sentiment classification, but not enough to have strong predictive power by itself. Is it possible that adding VADER into tf-idf Naive Bayes model we’ve established could improve performance?

We’ll do this by introducing a second model to our pipeline. The text will be processed in exactly the same way, and the Naive Bayes trained previously will be used here as well. However, instead of getting predictions from the Naive Bayes, we’ll take the predicted probabilities instead. Thus, for each tweet, we’ll get a value between 0 and 1 representing the predicted probability that the tweet is supportive. Then, we’ll feed this value, along with the VADER sentiment score, to a logistic regression model which will make our final prediction.

When comparing these results with the pure Naive Bayes approach, we find model performance actually drops. Overall accuracy falls by just over 1%. The type of mistake the model tends to make changes; this two part model gives far fewer false negatives (supportive tweets the model labels as negative) and far more false positives (negative tweets the model believes are supportive). Given that overall performance drops and model complexity increases, it seems logical to drop VADER completely and stick with Naive Bayes.

An example: start to finish on a sample Twitter profile

Having now put together code for the entire process, we can see how things are working by submitting a user’s Twitter handle and manually looking at their tweets. I picked a friend who has a fair number of tweets, of which I know some are transgender-related and all are supportive. So how does our tool end up doing?

First, the program is able to successfully scrape all 641 tweets sent by the user’s account, and process the text of all these tweets to make them suitable for machine learning. Of these, nine are found to be trans-related - and in manually reading through the remaining tweets, the simple classification used for this step seems to have worked. Out of these trans-related tweets, the Naive Bayes model used for sentiment analysis got eight tweets correct and incorrectly labelled one tweet, as follows:

God, Zuko is such a trans girl

Why did this tweet get misclassified? To understand a little better, we can check the model's predicted probability for this tweet, which is 0.46. As this is just below the threshold of 0.5, it is reassuring to see that this misclassified tweet was at least close to being classified correctly. As this is a very short tweet, with a name and several stopwords that are not part of the model, we can also look at each individual word/phrase in the tweet and see what their individual conditional probabilities are, to see where the negative prediction is coming from. Doing this creates the following four phrases, with their conditional probabilities:

- trans girl: 0.53
- trans: 0.72
- god: 0.43
- girl: 0.31

Interestingly, the word 'trans' makes the model think the tweet might be supportive, but the words 'god' and 'girl' both lead the prediction to be ultimately negative.

Next Steps

A great amount has been done to reach this point, but as often is the case in machine learning - there is always more we can do!

At the time of submitting this milestone report, the remaining work I would like to do for the project, organized by priority, are as follows:

1. Create slide deck to share results more easily, tidy up code to make public on Github
2. Explore R shiny and Flask to make this tool publicly usable and interactable
3. Use Keras to model with doc2vec and neural networks