**Social Media from Conflict Zones**

**Text analysis of social media posts from Syria**

**Michael L. Davies**  
**mld9s**

**Overview**

What can we learn about conflict environments from text analysis of social media posts? I assume that social media post from within a conflict zone reflects the environment. Of course, sentiments will likely ebb and flow, but are likely more negative in nature. Additionally, I wonder if text can predict (or are associated with) particular events on the ground.

For this project, I leveraged data maintained by [The Armed Conflict Location & Event Data Project (ACLED)](https://acleddata.com/#/dashboard). I narrowed the scope to the Syrian conflict, which persisted for more than a decade. Since 2017, ACLED has collected more than 80,000 social media and open source posts. The posts are then tagged by a curator with the date, province, city, and various associated events such as which actor (the Syria Army or non-state armed group) gained territory.

**Research questions and approach**

1. I expect that the nature of conflict is different in different regions of Syria. So, when conducting sentiment analysis, I control for sentiments by province.
2. I also plot sentiments over time.
3. Last, I expect that the text from postings reflect events "on the ground." So, I implement a number of classification models to test whether the text can predict changes in territorial control. As mentioned, the posts tagged by a curator with data on which actor (the Syria Army or non-state armed group) gained territory.

**Note**, because I'm interested in data science (rather than Anthropological) conventions, I chose to diverge from the project description in a few ways.

* First, to meet the project requirements, I perform data cleaning and a number of EDA/ETA methods as presented in the class--in part to create the TOKEN, VOCAB, and LIB csv's. From there, I chose to implement classification models from standard Python packages, which leverage much more streamlined pipelines than those learned in this class.
* Second, I was interested in several methods for text analysis not taught in this class, and I only found documentation on them in R. So, I implemented additional text analysis in R, which is included as an aside.

Interestingly, and possibly because we didn't leverage standard Python libraries for this class, I found R much cleaner in it's approach and pipelines. And I found much more interesting results with R.

The outline of this paper is as such:

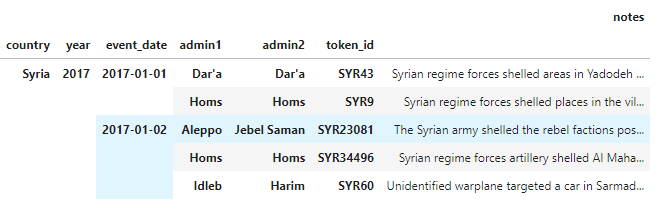
* **Section One: Pre-processing and building dataframes**
* **Section Two: Sentiment Analysis**
* **Section Three: Topic Modelling**
* **Section Four: Cluster Analysis**
* **Section Five: Classification - conventional pipelines**
* **Appendix: Implementation in R**

**Section One: Pre-processing and building dataframes**

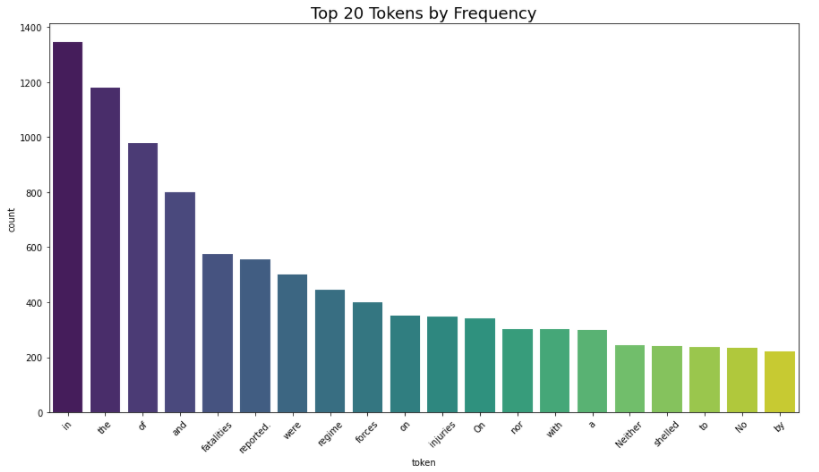
**Steps:**

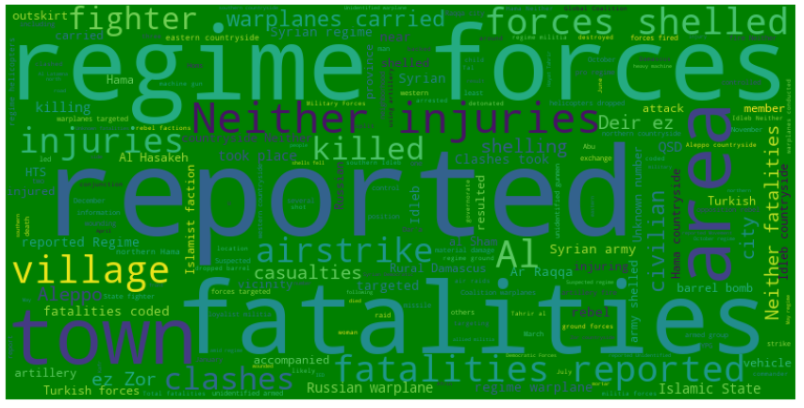
* View the data and set the OHCO configurations
* Initial exploratory plot of the word frequencies without preprocessing
* Preprocess the data, including generating parts-of-speech and TF-IDF
* Plot the results from pre-processing
* Build TOKEN, VOCAB, LIB, BOW, DTM/DTCM dataframes

To get a glimpse of the data after configuring a variant of an OHCO, we have:

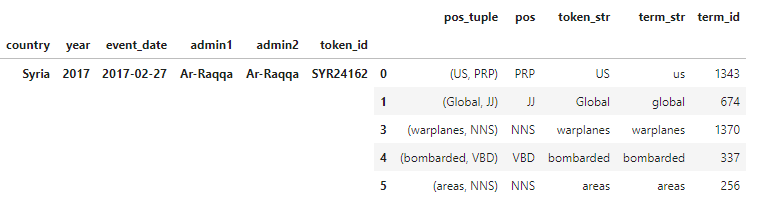


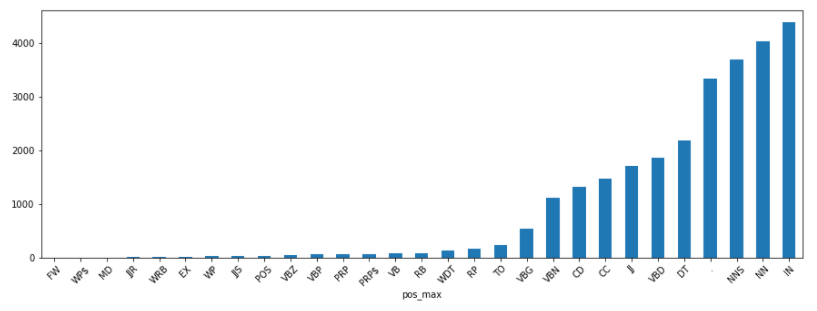
Before implementing any pre-processing, I implement the following analysis of raw token Frequencies. We can easily see that, without some pre-processing, stopwords and nouns dominate the data.

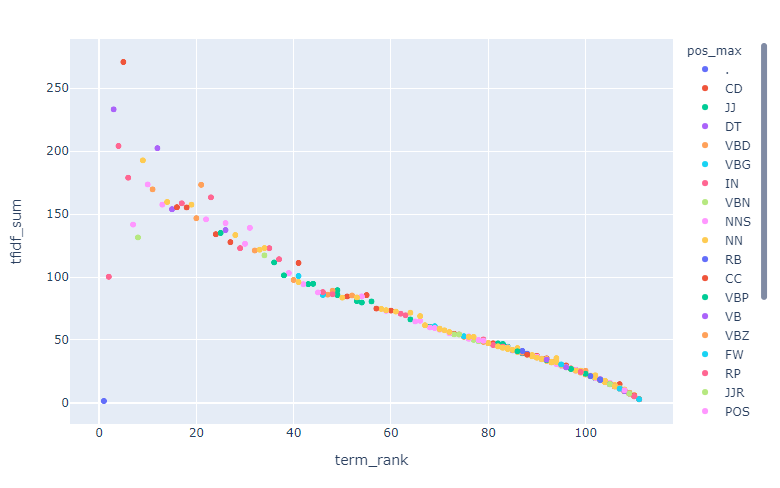




After some initial pre-processing, we have:





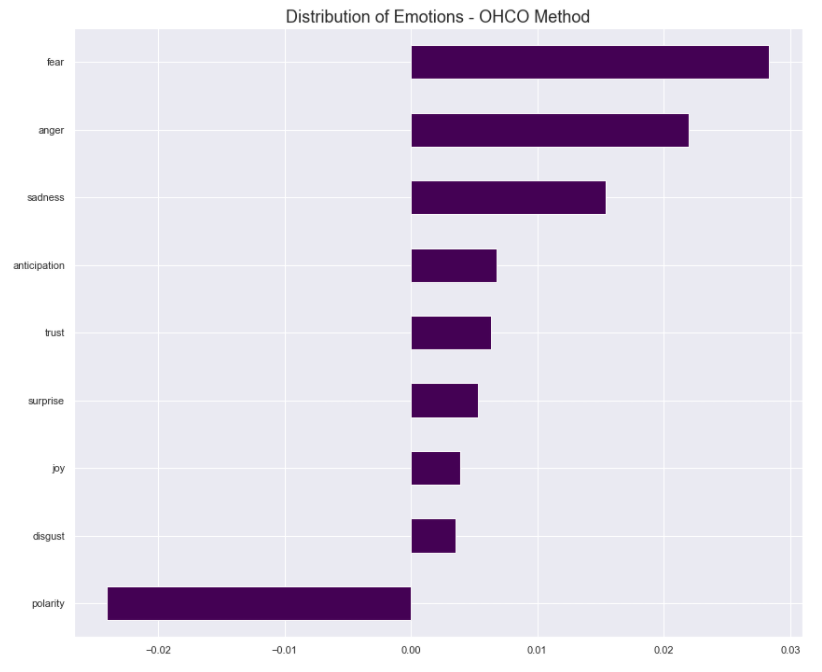


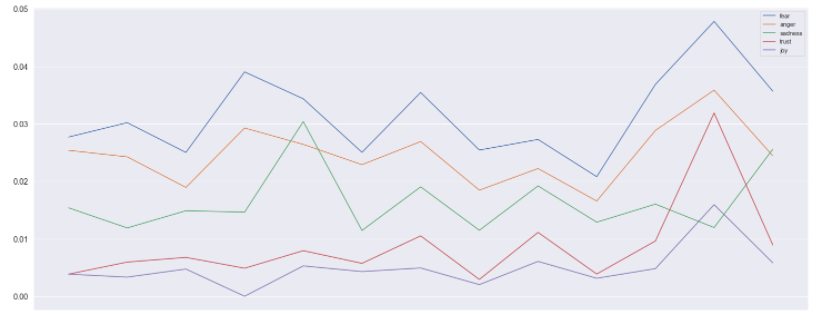
**Section Two: Sentiment Analysis**

**Steps:**

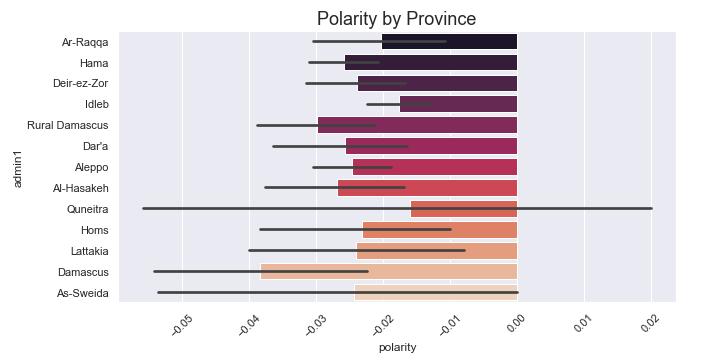
* Generate sentiments and polarity using the given sentiment analysis lexicon
* Plotting polarity grouped by token ID, province, and over time
* Plot distributions of emotions (anger, trust, etc)
* HTML plot of sentiments at the sentence level
* VADER plots for comparison

First, viewing the overall distribution of emotions, we see that fear, anger, and sadness are the most dominant emotions. This is expected.

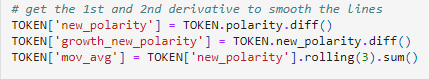




As described above, I’m interested in differences across provinces. Therefore, I plotted polarity scores by province. While there are some differences, the differences are not as stark as I anticipated.

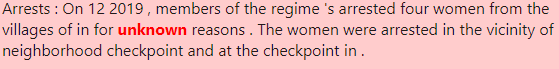


I was interested in view polarity over time. The python packages do not have a convenient smoothing function. So, I calculated a moving average. This didn’t turn out as well as I had hoped. However, we do see wild fluctuations over time and differences among the provinces.





Exploring sentiment at sentence level, we can see the words like “opposition”, “rebel” and “unknown” are automatically coded as negative—and this is not always accurate.

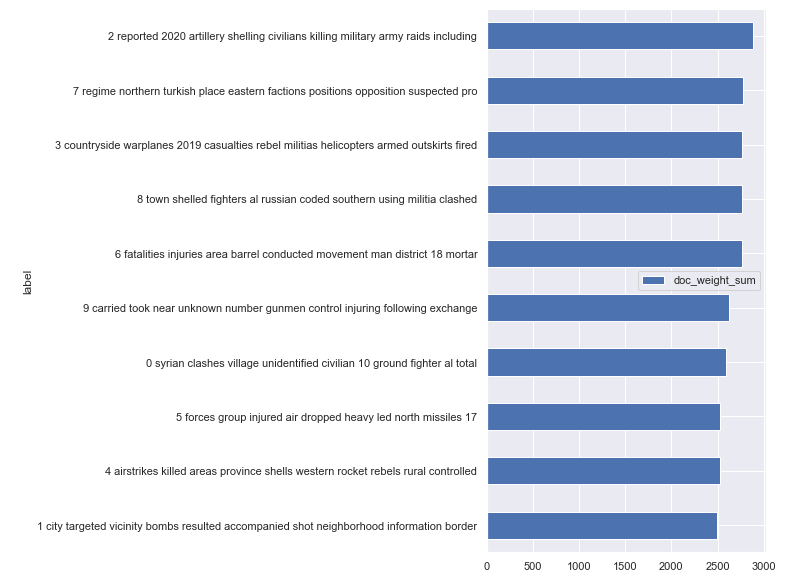


**Section Three: Topic Modelling**

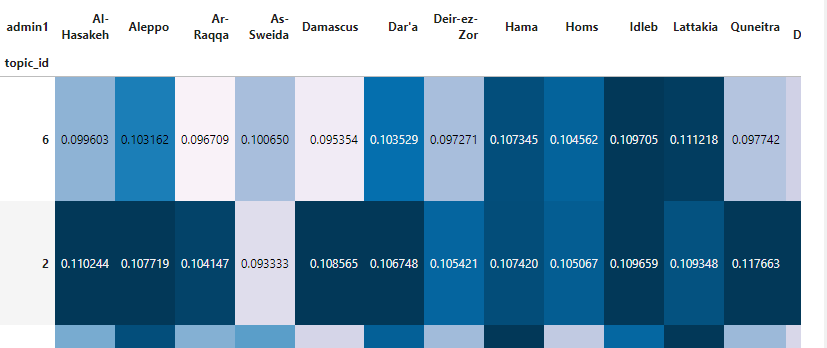
**Steps:**

* Generate and plot THETA and PHI
* Plot topics (sorted)
* Plot topics by province
* Scatter plot comparing topics by two provinces

I set the number of topics to 10, which in retrospect was too high. (Note, when I did it in R, I only used 6 topics. See appendix) It’s not clear to me that there are meaningful interpretations to these topics.

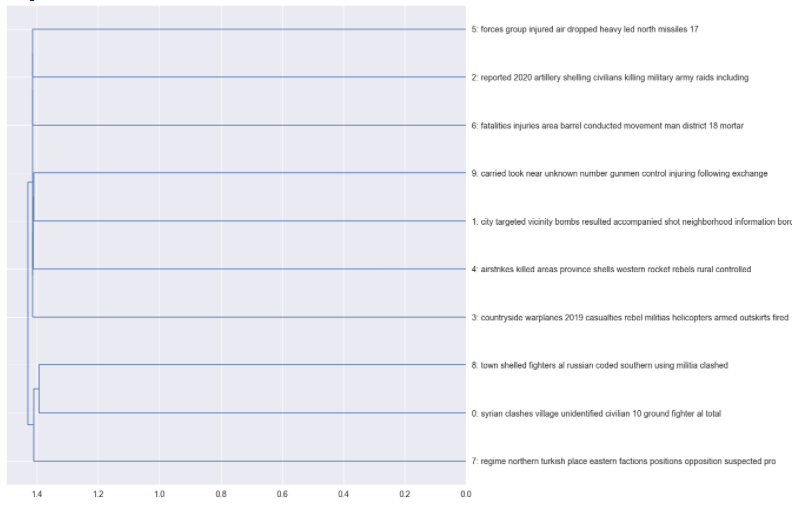


According to press reporting, the violence is currently greatest in Idleb province. So, I sorted the topics based on Idleb in descending order. We see topic 6 is the most strongly correlated.



**Section Four: Cluster Analysis**

* Plot clusters (dendogram) of topics



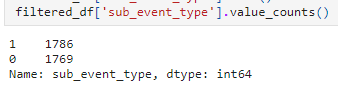
**Section Five: Classification - conventional pipelines**

**Steps:**

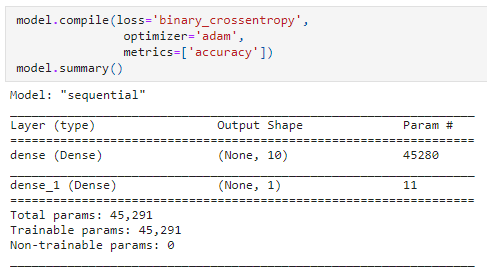
* Leverage spaCy
* Neural Networks - Keras/TensorFlow

In Python, I competed three classification approaches. First, I use a base line model, which is the simplest, form sklearn. Second, I leveraged spaCy for preprocessing and added the TF-IDF module with the sklearn for classification. Last, I implemented a Keras neural network, which is a high-level language that sits on top of TensorFlow. I only present the findings of the TensorFlow model here.

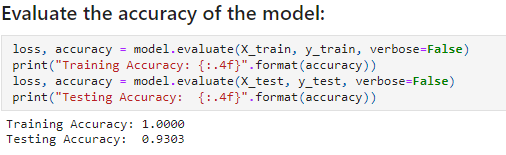
My primary interest was – can we use text from social media to predict the outcome of a battle. As such, my response variable is binary where the social media text is associated with "Syrian regime regains territory" or "Non-state actor gains territory" - recoded as 1 and 0 respectively I first clean the data for only those cases code according to the response variable. (Many posts are only associated with various types of clashes and battles with no turn over in terrain.) Fortunately, the data is balanced across labels, so no up/down sampling required and “accuracy” is sufficient for evaluating the models.

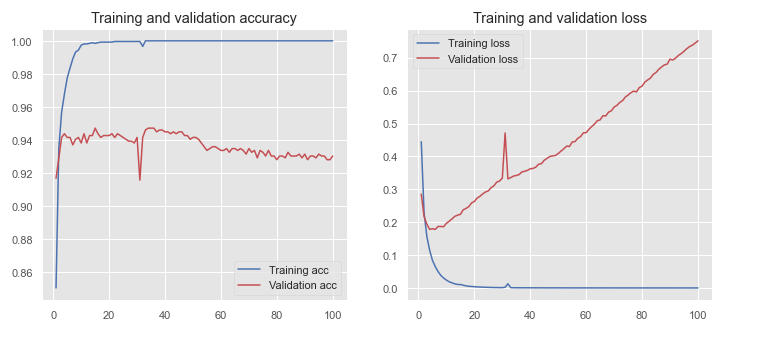


Setting up the TensorFlow pipeline initializes the following model:



Fitting the model on a train and then test set resulting in an accuracy of 93, suggesting the text analysis generated features that were strong predictors of territorial turn over.





**Findings**

Beyond the analysis of parts of speech and word frequencies…

**Sentiments/polarity:**  
Polarity counts to vary across provinces. This is to be expected. The conflict has taken on different characteristics in different provinces. In addition, polarity has varied widely over time, and differentially by province. The dominant emotions have been “fear”, “anger”, “sadness”. These emotions ebbed and flowed over time, but always outpaced “trust” and “joy” – regardless of which province we look at. This is later supported by the Vader approach. An important caveat with respect to sentiments – the word “opposition” appears to be coded in the lexicon as a negative emotion. However, it’s not always clear that the sentence itself is expressing negativity.

**Topic Modeling**  
The text did not warrant a large number of topics, so I set the number of topics in Python to 10. (With R, I lowered the number of potential topics to 6) From a heat map of the topic scores, we once again see that topics vary by province. Here we see variation in the topics by province. Notably, it is hard to distinguish clear topics as many similar words show up in each topic.

**Classification**  
My primary interest was – can we use text from social media to predict the outcome of a battle. As such, my response variable is binary where the social media text is associated with "Syrian regime regains territory" or "Non-state actor gains territory" - recoded as 1 and 0 respectively I first clean the data for only those cases code according to the response variable. (Many posts are only associated with various types of clashes and battles with no turn over in terrain.) Fortunately, the data is balanced across labels, so no up/down sampling required and “accuracy” is sufficient for evaluating the models.

I then compete multiple classification algorithms. I begin with the simplest model. sklearn's feature extraction package handles the text preprocessing before sending it to the logistic regression. Additionally, I import the module for the train/test split (set at 75/25)

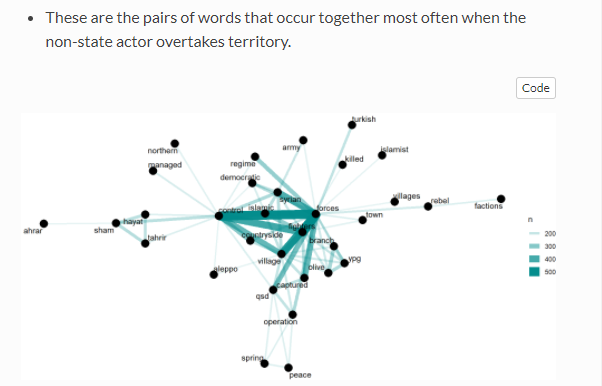
In Python, I compete three classification approaches.

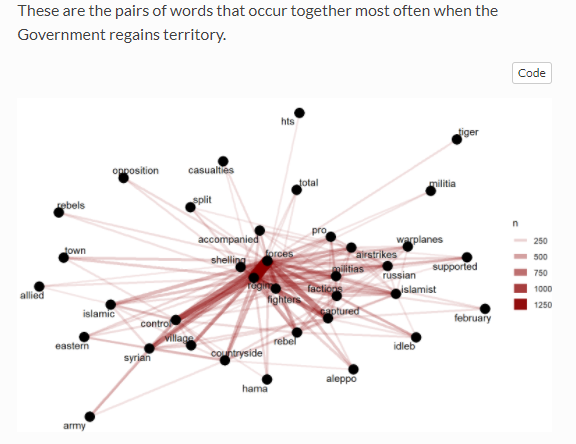
First, I leverage spaCy for preprocessing and added the TF-IDF module with the sklearn for classification. sklearn's feature extraction package handles the text preprocessing before sending it to the logistic regression. Second, I use a base line model, which is the simplest, form sklearn without TF-IDF. Last, I implemented a Keras neural network, which is a high-level language that sits on top of TensorFlow. Note, for all models, I used a train/test split (set at 75/25)

In the end, prediction was extremely robust, with all models achieve accuracy between 92 and 94 percent. Interestingly, Keras (TensorFlow) was not the most accurate. This suggests that this text can be used to generate features that predict territorial turn over.

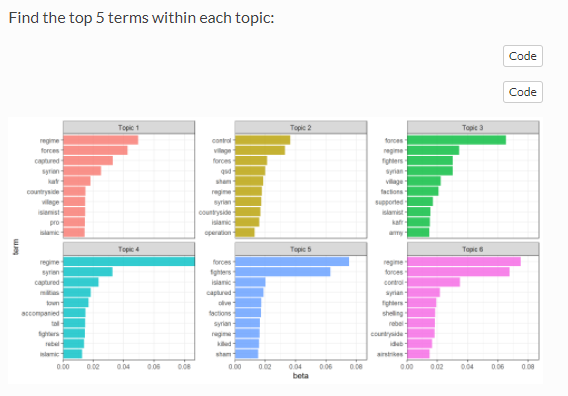
**Appendix: Findings in R**

Given that this is not required for this project, I will only high light aspects that were particularly fun. In general, I was able to implement a much more sophisticated treatment of uni-grams and bigrams. As well, I conducted a network analysis of words associated with each label, (a cleaner) topic analysis, frequencies of co-occurances with words of interest. Last, I conducted a bootstrapped logistic regression to predict the labels. For this, I plotted a variables importance plot to show which words (unigrams or bigrams) were the most important in predicting the response.

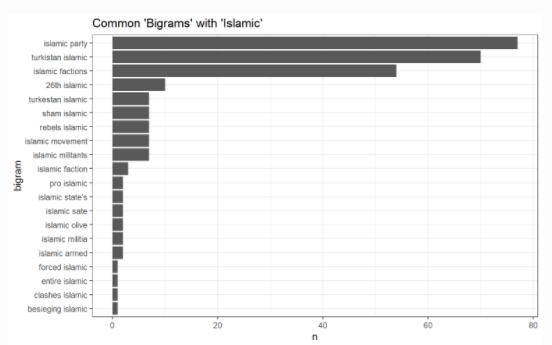


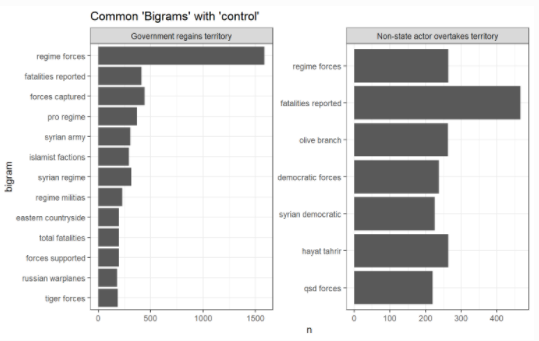


Using Latent Dirichlet Allocation with the topicmodels package (setting topics to 6), I obtained distinct topics. However, similar to above in python, interpretation seems difficult.



An interesting application was to look at common bigrams that co-occur with specific words. I chose to look at co-occurrences with “control” and “Islamic”:





Last, I used a bootstrapped logistic regression to predict which side took territory. Importantly, tidymodels and tidytext facilitated building features from the text *and* leveraging existing features in the data (other than the text).



Then, once we have the best model, we can use variable importance to find which unigrams and bigrams are important in predicting territorial change. We get:

