NATURAL LANGUAGE PROCESSING: ELECTION TWEETS

Angelo Taranto

DATA COLLECTION

- Used Twitter API's GetStreamFilter to capture 43,000 live tweets on November 10th, with the topic tracked as "election"
- Created Pandas DataFrame of tweets including tweet ID, time of posting, username, and text
- Preliminary cleaning of text and saved file as a CSV

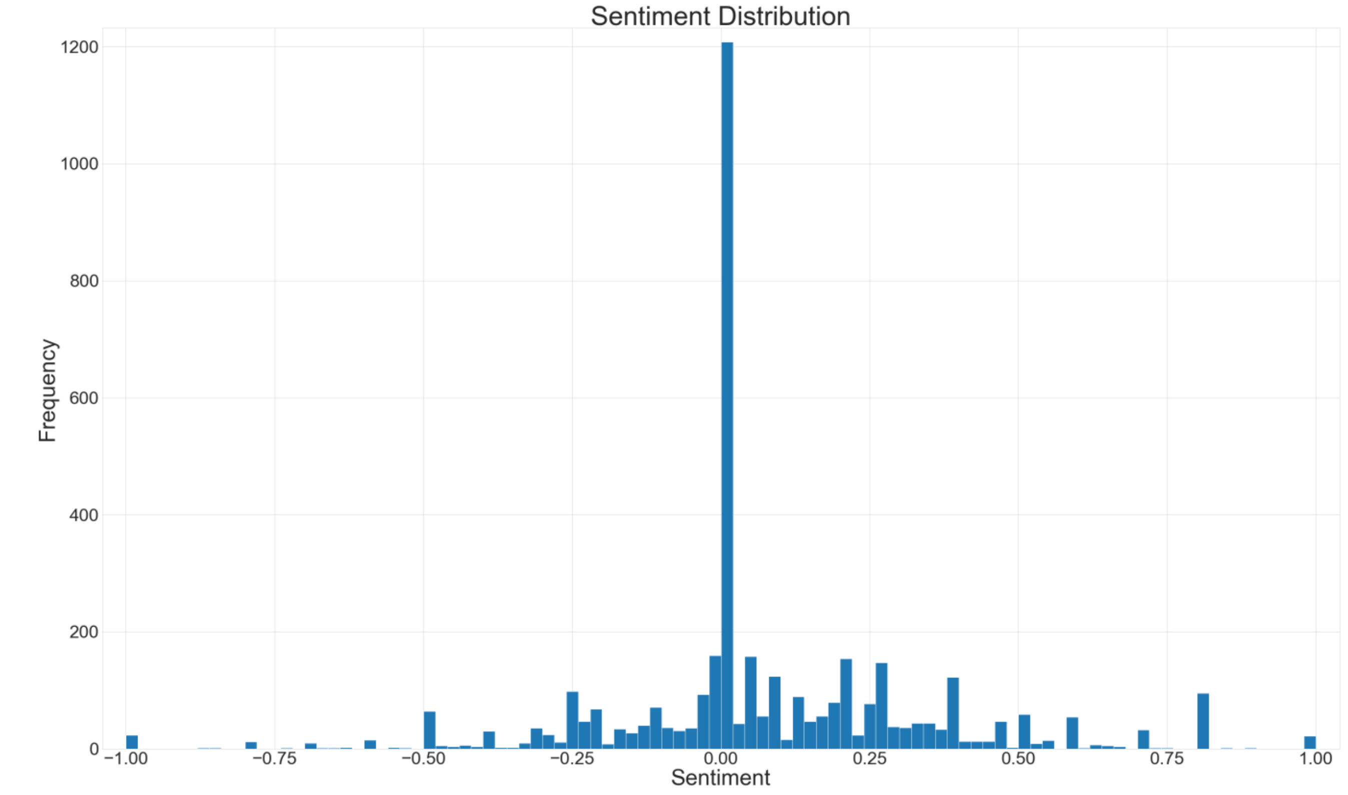
DATA COLLECTION

- So far, data is unsupervised – there is no objective to predict: Let's classify the tweets by their political affiliation.
- Hand-labeled the first 4,000 tweets as Pro-Trump/Anti-Biden, Neutral, and Pro-Biden/Anti-Trump.
- Note potential source of error: GetStreamFilter gathers tweets as they come in, so the first 4,000 tweets were made prior to the rest of the data. This may ruin independence assumptions between the observations. All the data was gathered over a span of about 17 minutes, so some later tweets may be replies to earlier ones.

DATA ANALYSIS

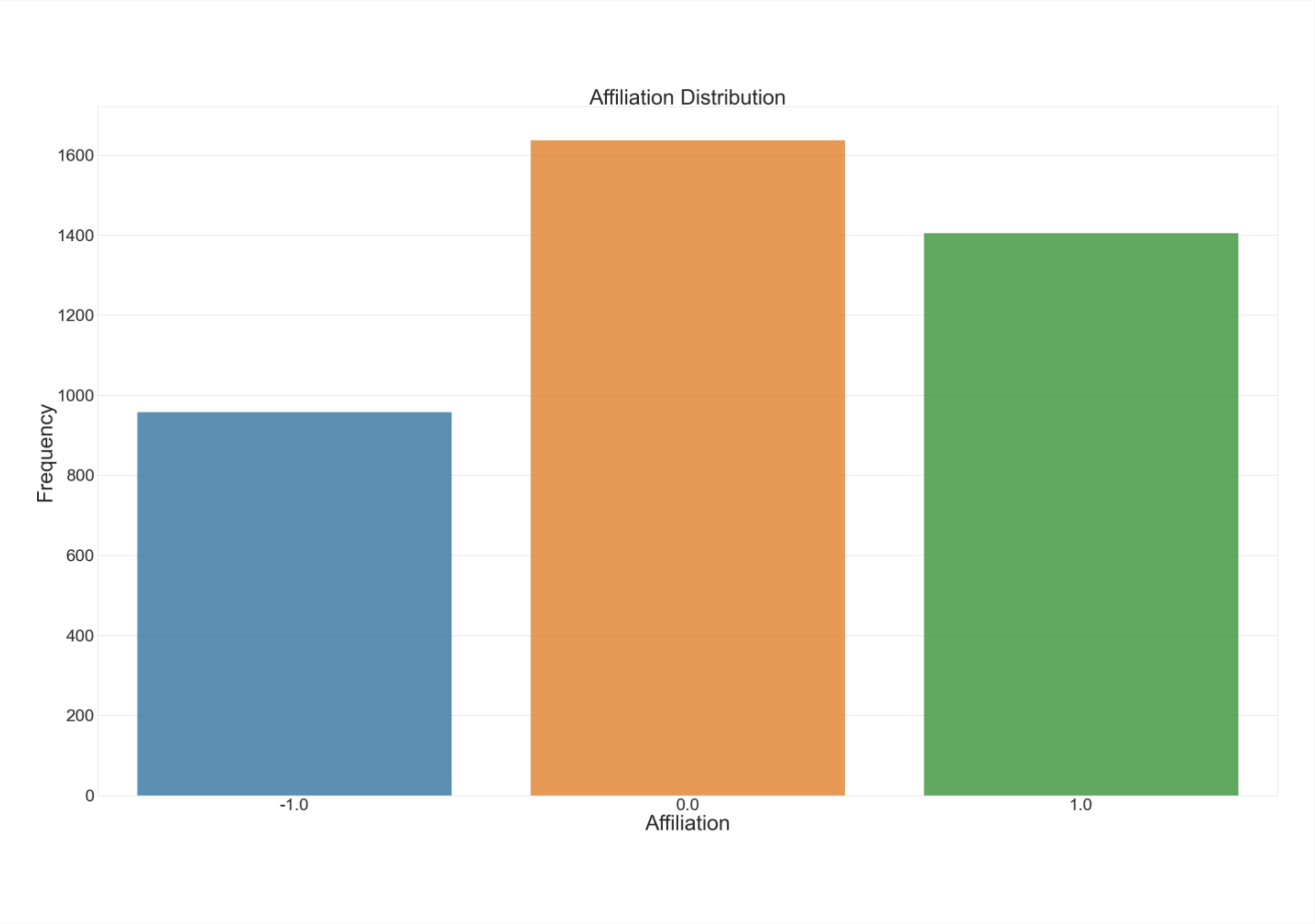
- Main component of data to be analyzed: text
- Text cleaned of contractions, tokenized, made lowercase, cleaned of punctuation, cleaned of "stop words", and then lemmatized

- **First analysis**: TextBlob Sentiment Polarity, ranging from –1 (bad) to 1 (good) on the 4,000 labeled tweets.



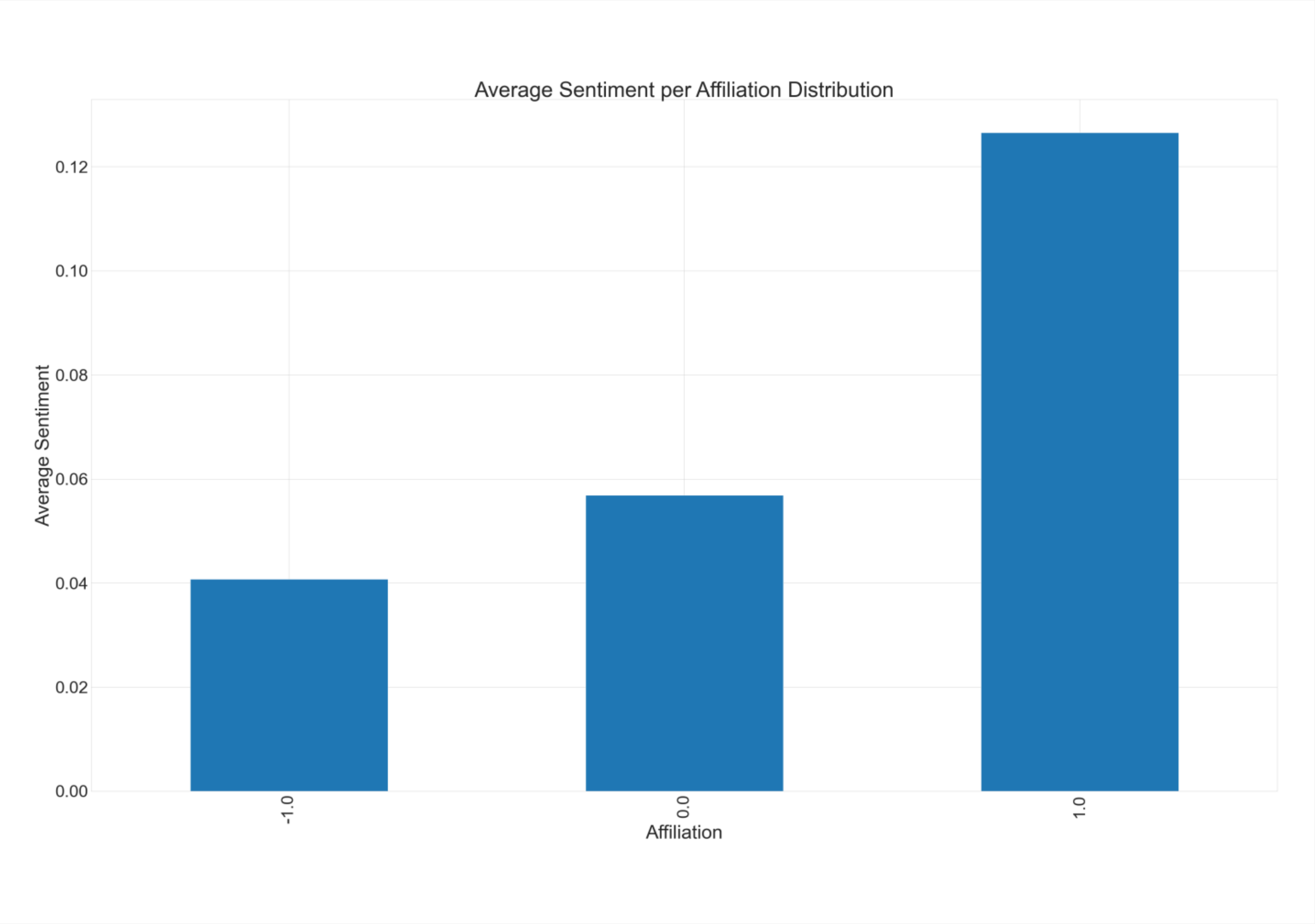
DATA ANALYSIS

- Affiliation:
  - -1 Pro-Biden/Anti-Trump
  - 0 Neutral
  - 1 Pro-Trump/Anti-Biden

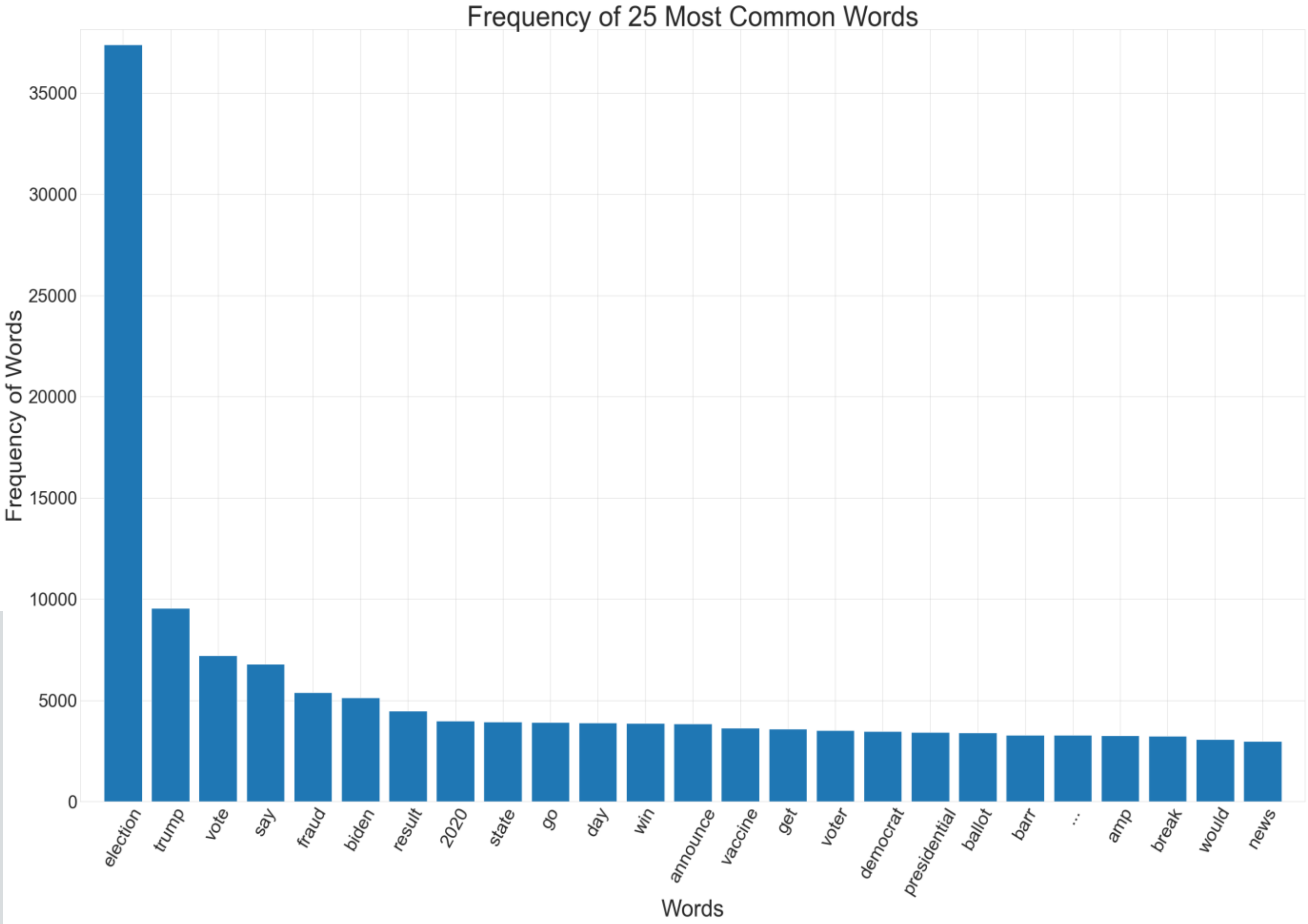


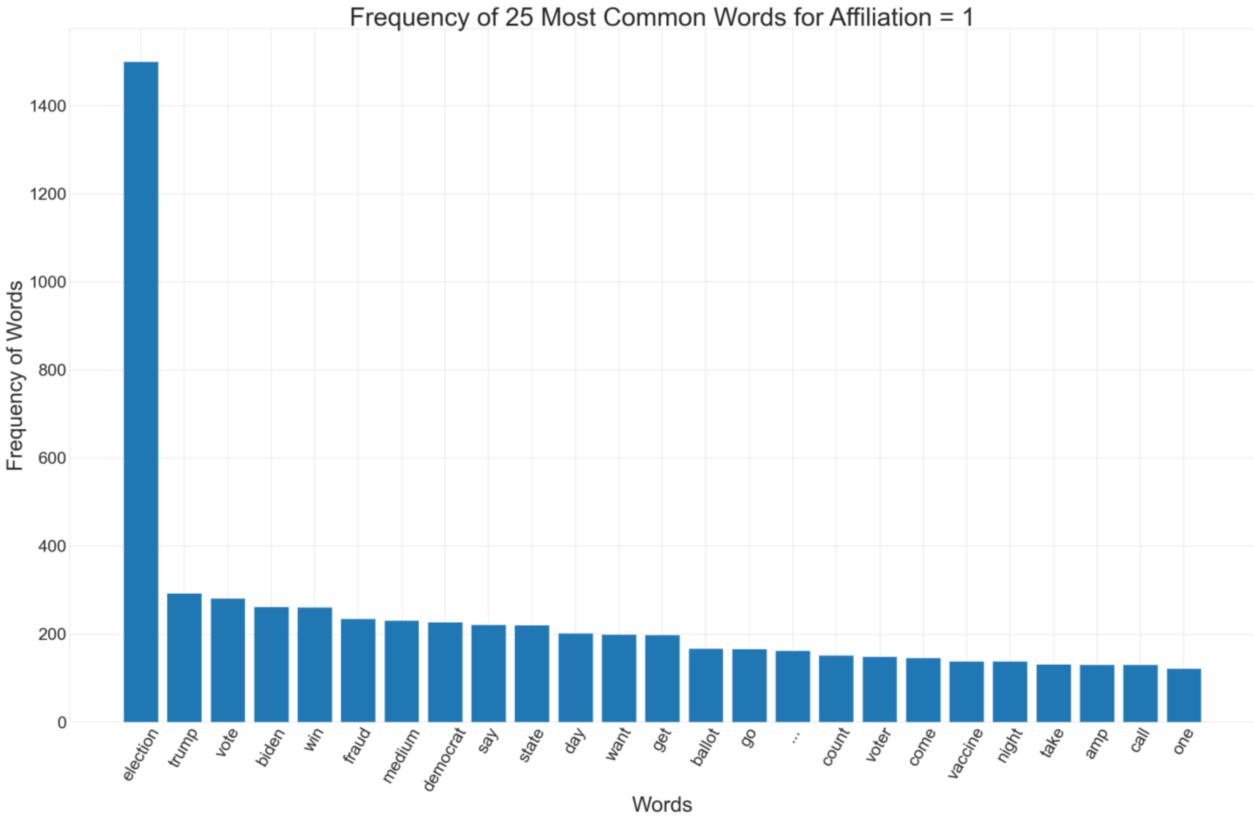
DATA ANALYSIS

- Sentiment is relatively high in the Pro-Trump/Anti-Biden group.
- It seems those in this camp were saying positive things about Trump, while those on the other team were saying negative things about Trump (more than saying anything about Biden).
- However, note the small scale.

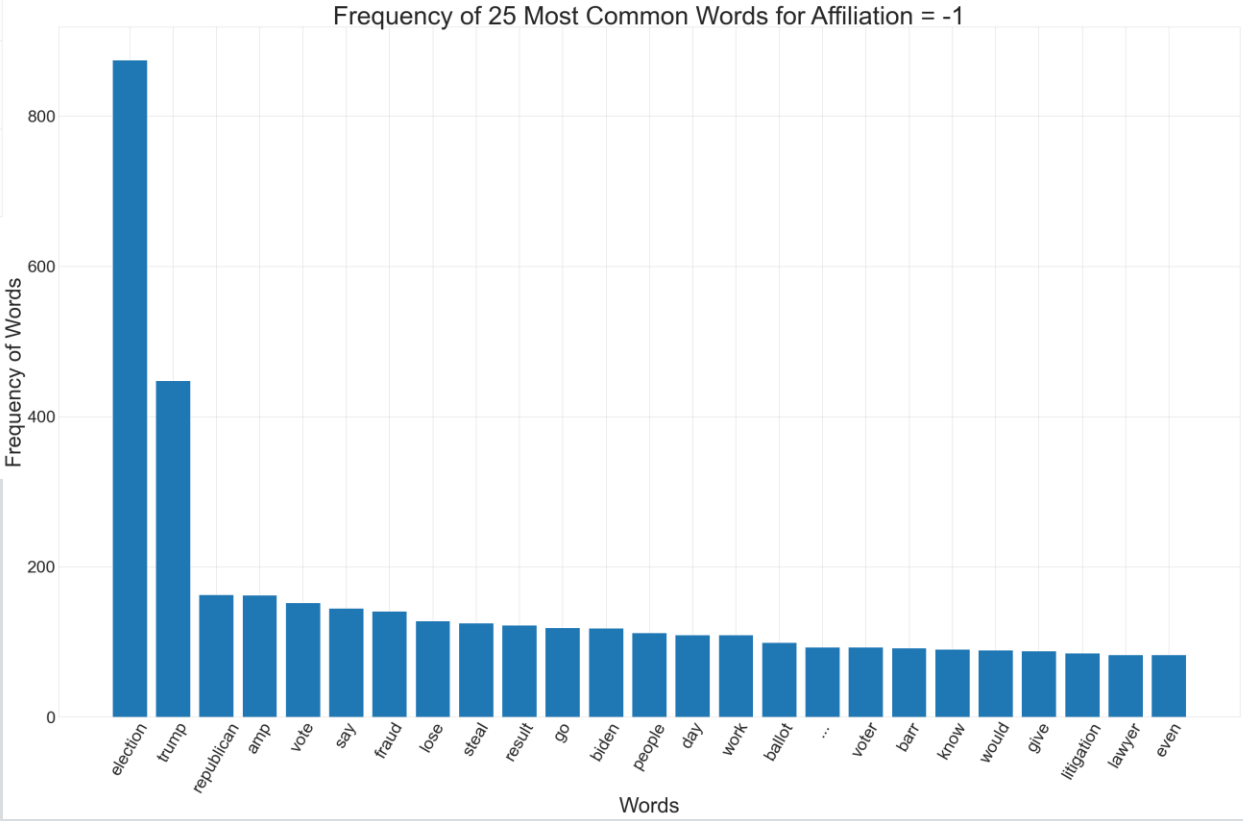


WORD FREQUENCY





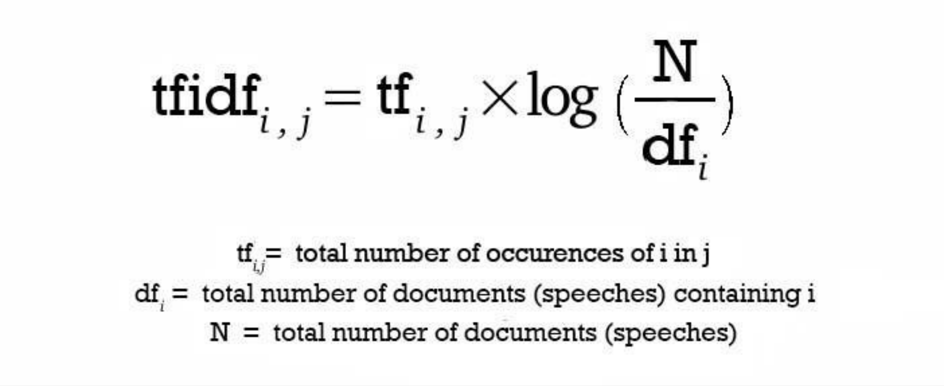
Frequency of 25 Most Common Words for Affiliation = 1



Frequency of 25 Most Common Words for Affiliation = -1

TOPIC MODELING: LDA AND NMF

- For Latent Dirichlet Allocation (LDA) topic modeling, lemmatized text data was word-bagged using CountVectorizer to count the occurrence of all words in a single tweet and represent that as a vector.
- For Non-negative Matrix Factorization (NMF) topic modeling, lemmatized text data was word-bagged using a term frequency – inverse document frequency (TF-IDF) vectorizer, which weights words in a tweet by how often it appears in that tweet and inversely by how often it appears in many tweets.



$$\mathrm{tfidf}_{i,j} = \mathrm{tf}_{i,j} \times \log\left(\frac{\mathrm{N}}{\mathrm{df}_i}\right)$$

$\mathrm{tf}_{i,j}$ = total number of occurences of i in j

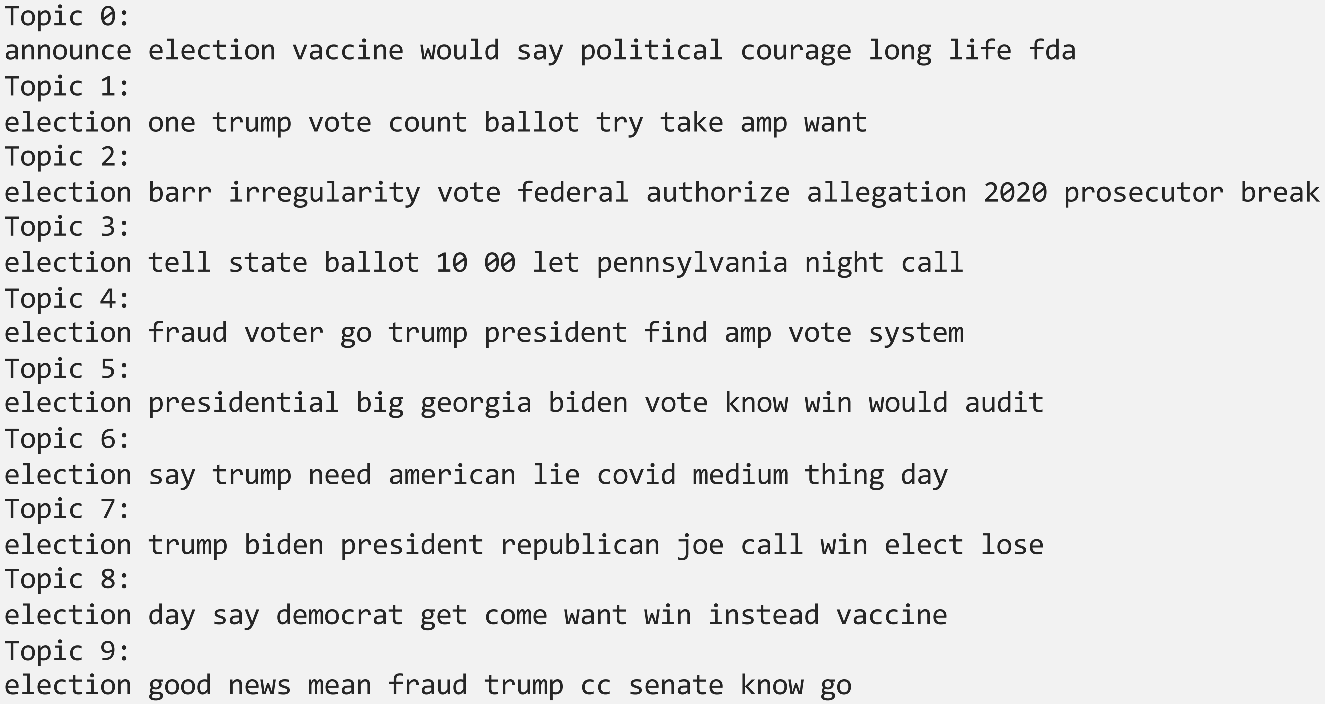
df_i = total number of documents (speeches) containing i

N = total number of documents (speeches)

ASIDE: WORD-BAGGING VOCAB

- Both CountVectorizer and TFIDFVectorizer generated the following list of words as a vocabulary for word-bagging:
  - [00, 000, 09, 10, 100, 1000, 109, 10m, 11, 118, …, york, you, young, youth, youtube, yr, zero, zoom, zuckerberg, zuckerbooger]
  - Example: The tweet "zero youths, you zuckerbooger!" Would be vectorized (at least by CountVectorizer) as [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1]
- There are 2907 words here, because both vectorizers ignored words that were present in 90% or more of the tweets, and words that appeared fewer than 15 times in all tweets.

TOPIC MODELING: LDA RESULTS



```
Topic 0:
announce election vaccine would say political courage long life fda
Topic 1:
election one trump vote count ballot try take amp want
Topic 2:
election barr irregularity vote federal authorize allegation 2020 prosecutor break
Topic 3:
election tell state ballot 10 00 let pennsylvania night call
Topic 4:
election fraud voter go trump president find amp vote system
Topic 5:
election presidential big georgia biden vote know win would audit
Topic 6:
election say trump need american lie covid medium thing day
Topic 7:
election trump biden president republican joe call win elect lose
Topic 8:
election day say democrat get come want win instead vaccine
Topic 9:
election good news mean fraud trump cc senate know go
```

Topic 0: FDA announcing vaccine on election night

Topic 1: Count ballots (Trump will try to take election?)

Topic 2: Alleged vote irregularity, Barr & prosecutors

Topic 3: Pennsylvania state called, night of Nov 10th

Topic 4: Finding voting fraud in the system (for Trump)

Topic 5: Auditing of Georgia election if Biden wins

Topic 6: Trump needs Americans, covid is a lie(?)

Topic 7: Election called; Biden elected president

Topic 8: Vaccine reported near election day

Topic 9: Senate knew, good news means fraud

TOPIC MODELING: NMF RESULTS

```
Topic 0:
authorize prosecutor irregularity barr federal allegation substantial pursue
certify break
Topic 1:
five later prior instead along come want democrat fda get
Topic 2:
announce likewise purpose earlier save life others courage political long
Topic 3:
biden call rescind pennsylvania realclearpolitics win election president result
live
Topic 4:
cc good mean news cruz ted decide medium fox election
Topic 5:
trump donald go election steal lose try attempt accept openly
Topic 6:
count ballot observer state tell arena secret farm shut tabulation
Topic 7:
georgia big win night presidential election donald trump legally cast
Topic 8:
donald disgraceful bureaucracy weaponize totally always ago impossible
government lie
Topic 9:
audit orange board california race recount county announce presidential
election
```

Topic 0: Allegations of voter irregularities pursued by Barr & prosecutors, authorized to pursue etc.

Topic 1: ???

Topic 2: (Vaccine) could have been announced earlier to save lives

Topic 3: Realclearpolitics rescinds their call of Biden winning Pennsylvania, live

Topic 4: Ted Cruz, Fox News, Medium... something decides the election?

Topic 5: Trump claims election steal due to lose, vs. openly accepting

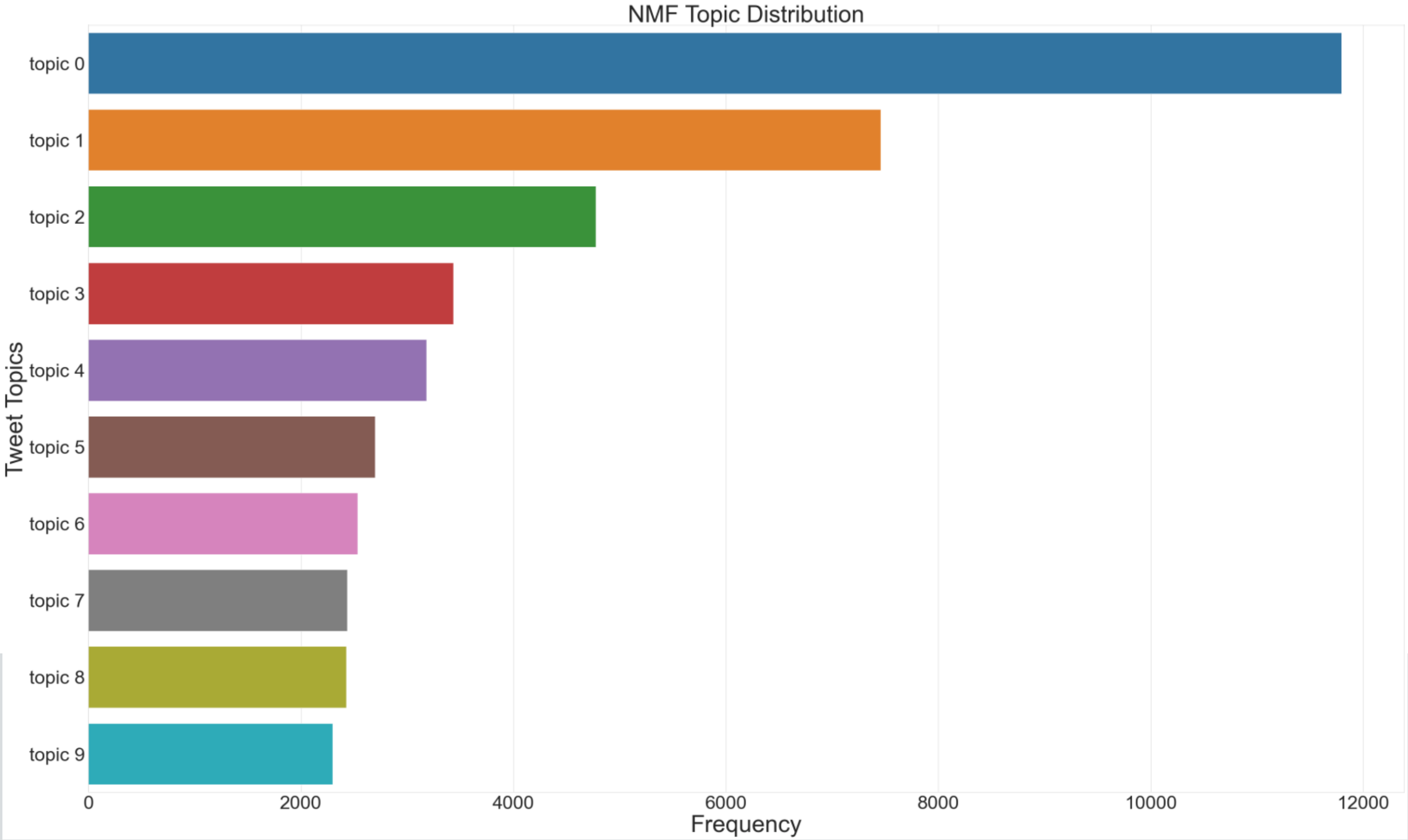
Topic 6: Vote tabulation and ballot observers

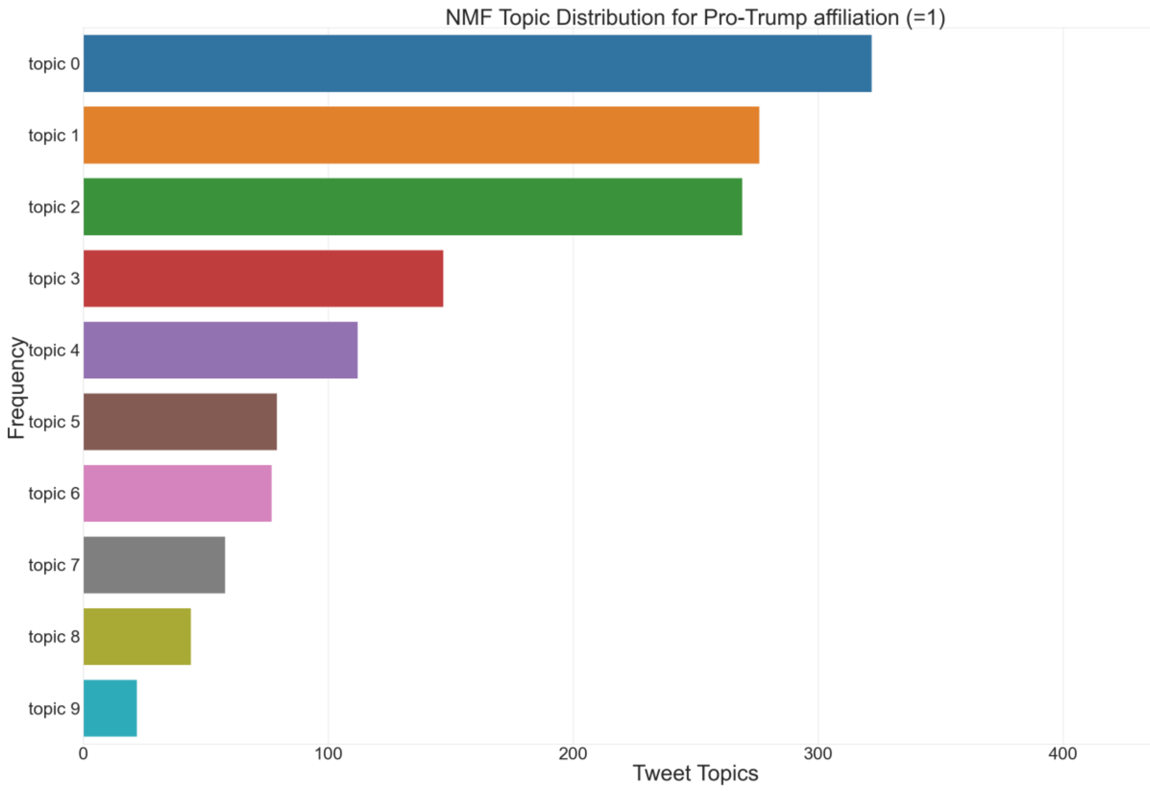
Topic 7: Big win in Georgia on night of Pres. election

Topic 8: ???

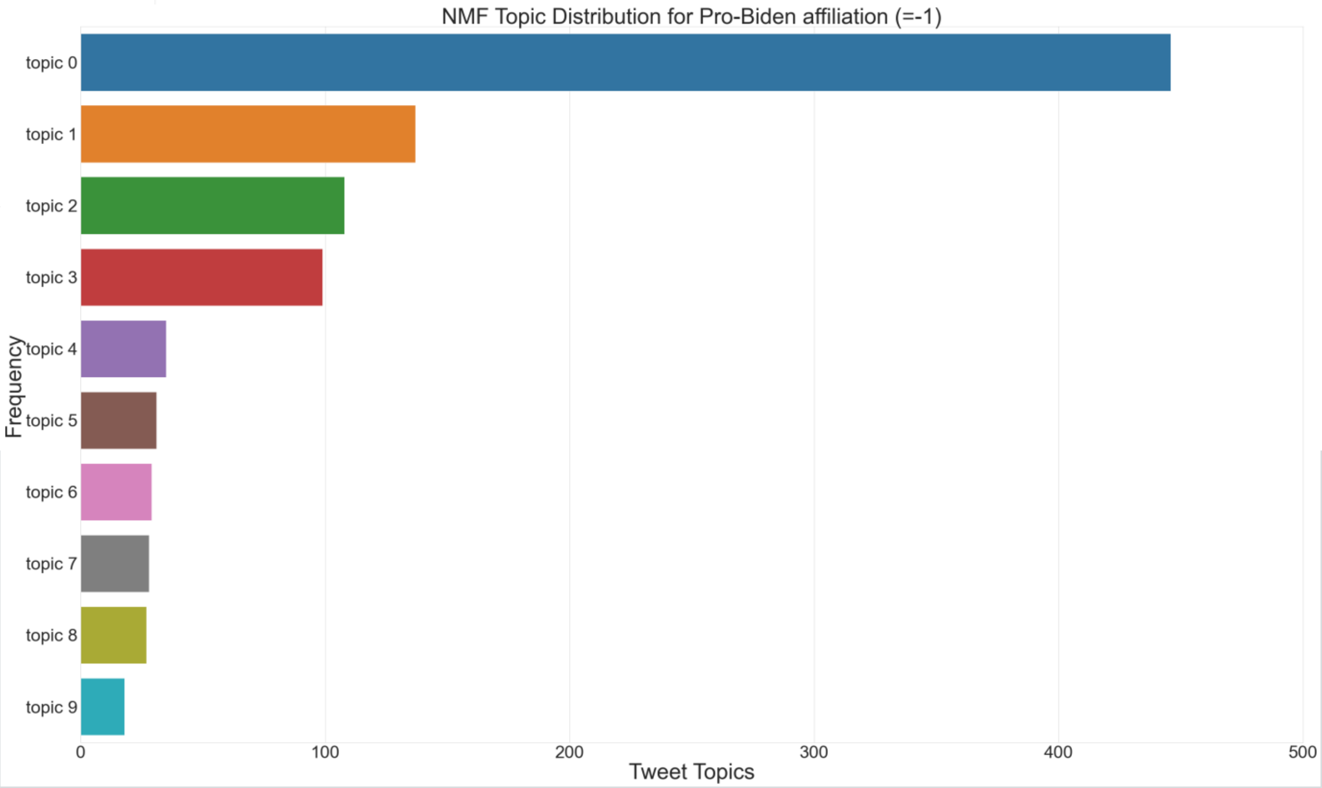
Topic 9: Orange county, CA audits race, announces recount

NMF TOPIC MODELING DISTRIBUTIONS



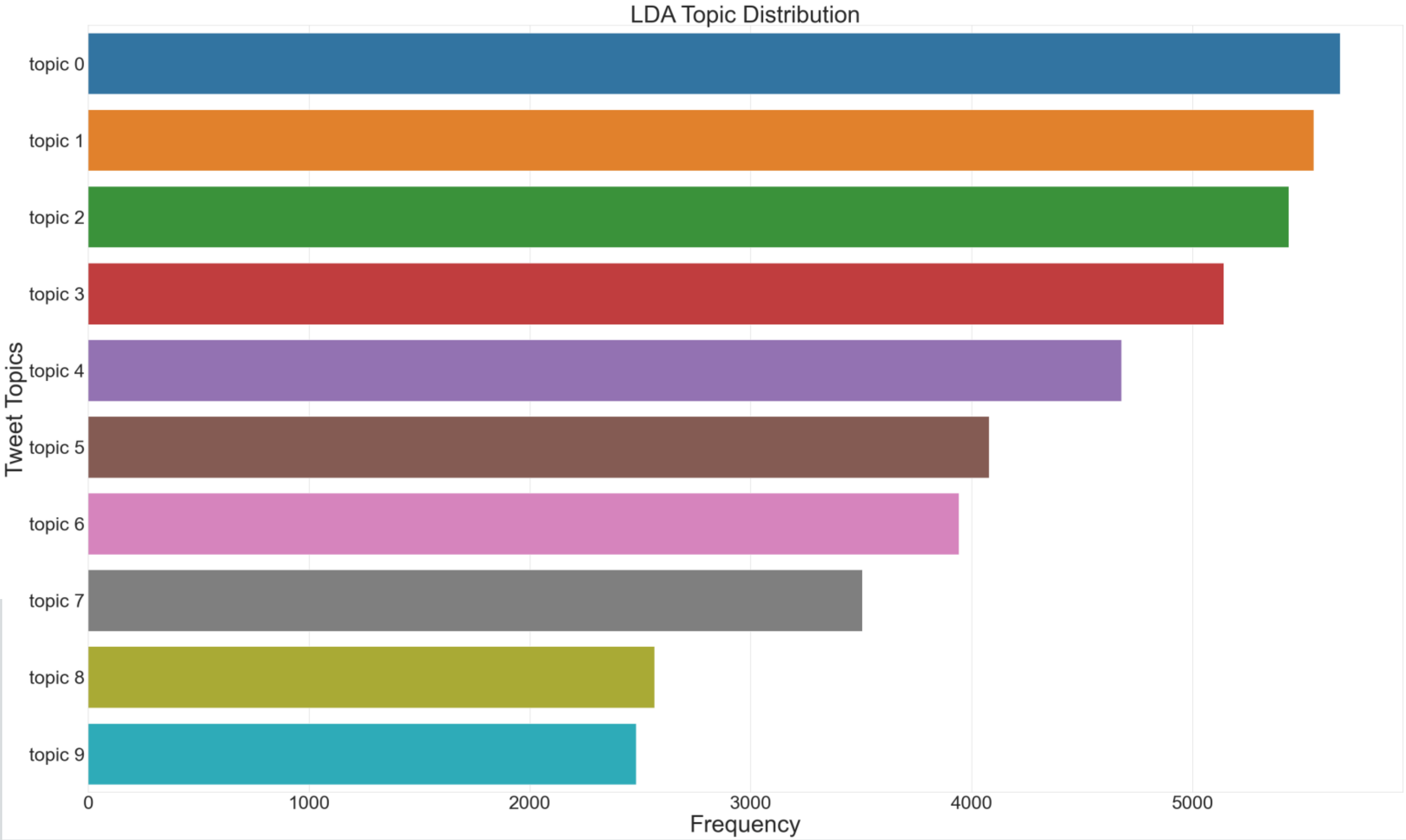


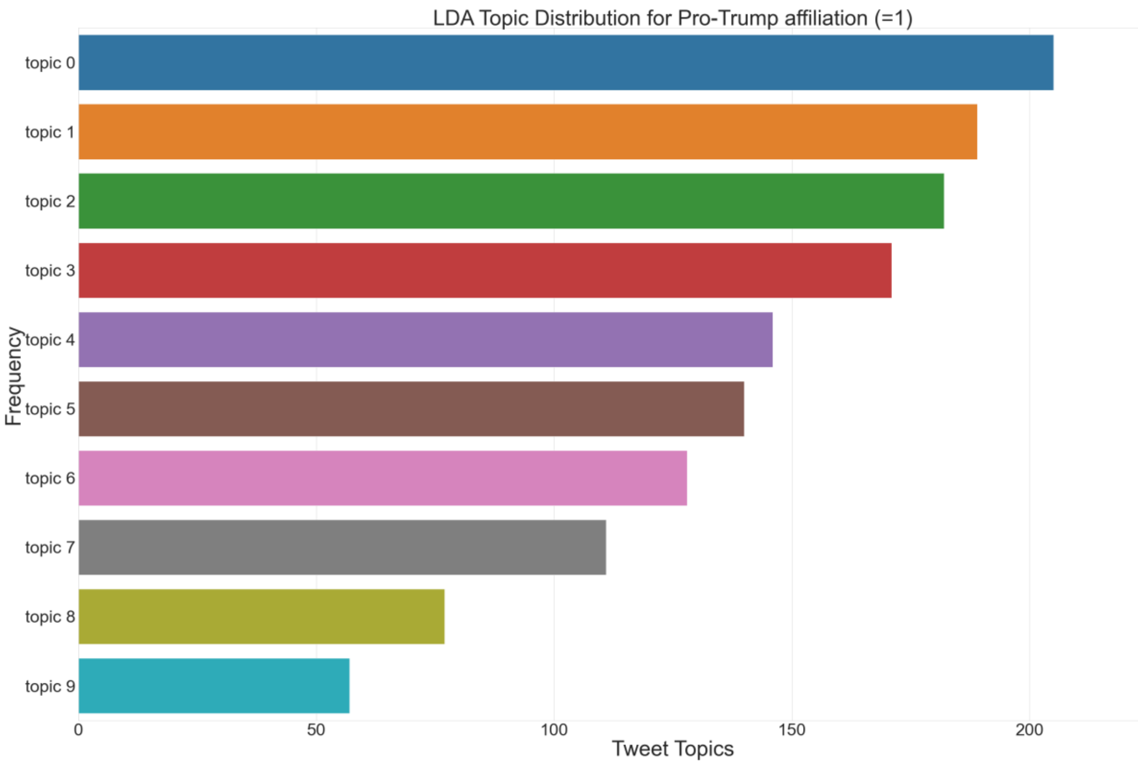
NMF Topic Distribution for Pro-Trump affiliation (=1)



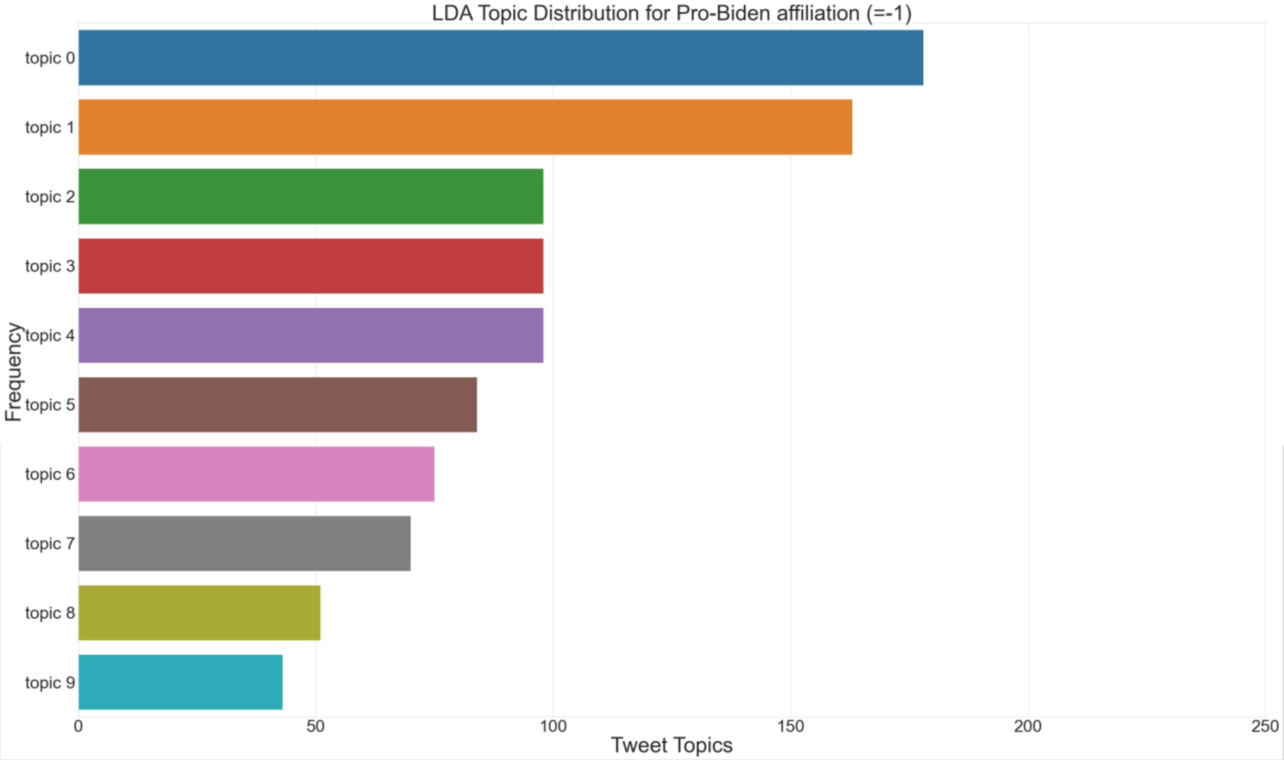
NMF Topic Distribution for Pro-Biden affiliation (=-1)

LDA TOPIC MODELING DISTRIBUTIONS





LDA Topic Distribution for Pro-Trump affiliation (=1)



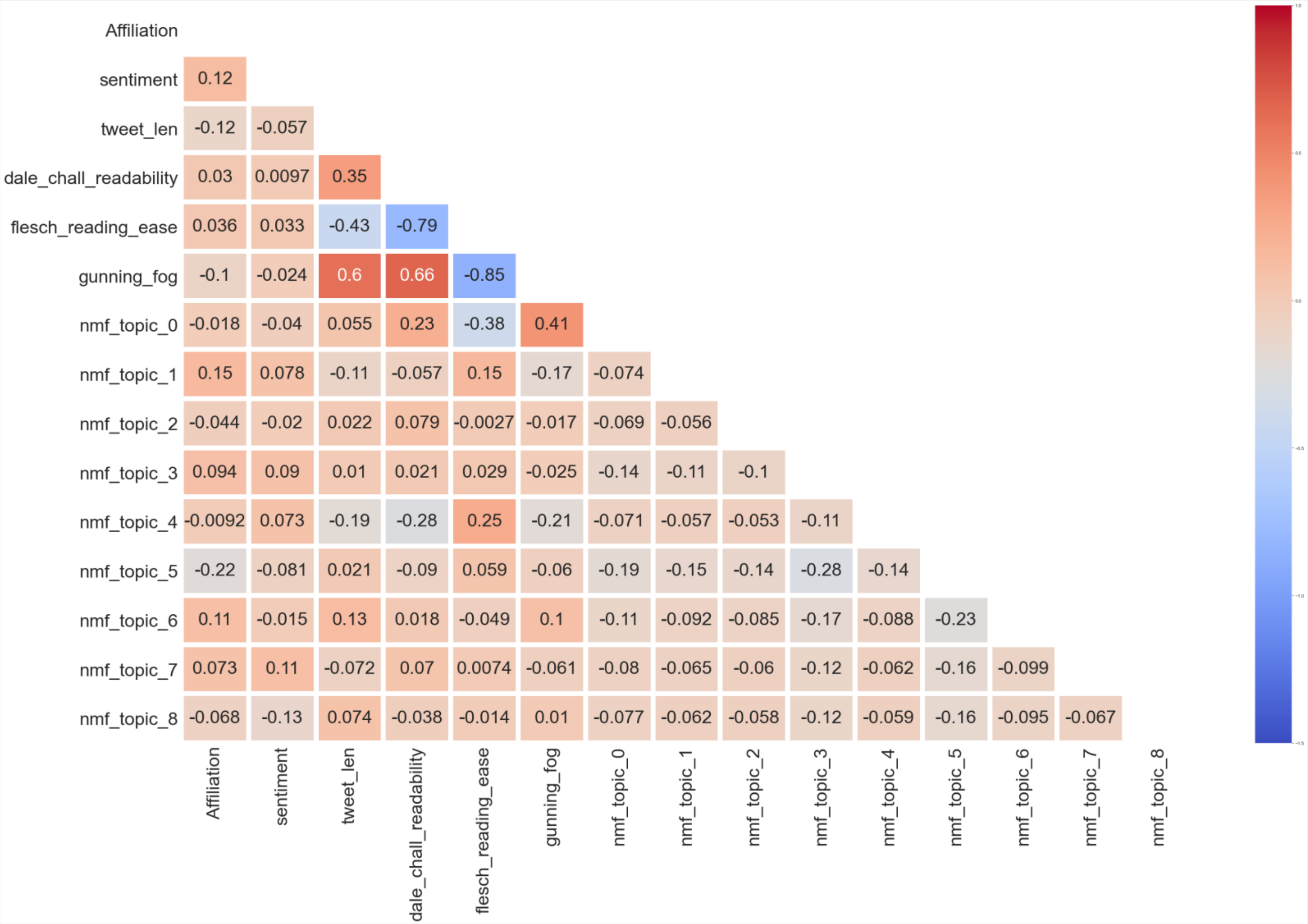
LDA Topic Distribution for Pro-Biden affiliation (=-1)

READABILITY ANALYSIS

- Dale Chall Readability score (mean)
- Flesch Reading Ease (mean)
- Gunning – Fog score (mean)
- Text Standard score (mode)

None showed significant difference between the affiliations.

CORRELATIONS



	Affiliation	sentiment	tweet_len	dale_chall_readability	flesch_reading_ease	gunning_fog	nmf_topic_0	nmf_topic_1	nmf_topic_2	nmf_topic_3	nmf_topic_4	nmf_topic_5	nmf_topic_6	nmf_topic_7	nmf_topic_8
Affiliation															
sentiment	0.12														
tweet_len	-0.12	-0.057													
dale_chall_readability	0.03	0.0097	0.35												
flesch_reading_ease	0.036	0.033	-0.43	-0.79											
gunning_fog	-0.1	-0.024	0.6	0.66	-0.85										
nmf_topic_0	-0.018	-0.04	0.055	0.23	-0.38	0.41									
nmf_topic_1	0.15	0.078	-0.11	-0.057	0.15	-0.17	-0.074								
nmf_topic_2	-0.044	-0.02	0.022	0.079	-0.0027	-0.017	-0.069	-0.056							
nmf_topic_3	0.094	0.09	0.01	0.021	0.029	-0.025	-0.14	-0.11	-0.1						
nmf_topic_4	-0.0092	0.073	-0.19	-0.28	0.25	-0.21	-0.071	-0.057	-0.053	-0.11					
nmf_topic_5	-0.22	-0.081	0.021	-0.09	0.059	-0.06	-0.19	-0.15	-0.14	-0.28	-0.14				
nmf_topic_6	0.11	-0.015	0.13	0.018	-0.049	0.1	-0.11	-0.092	-0.085	-0.17	-0.088	-0.23			
nmf_topic_7	0.073	0.11	-0.072	0.07	0.0074	-0.061	-0.08	-0.065	-0.06	-0.12	-0.062	-0.16	-0.099		
nmf_topic_8	-0.068	-0.13	0.074	-0.038	-0.014	0.01	-0.077	-0.062	-0.058	-0.12	-0.059	-0.16	-0.095	-0.067	

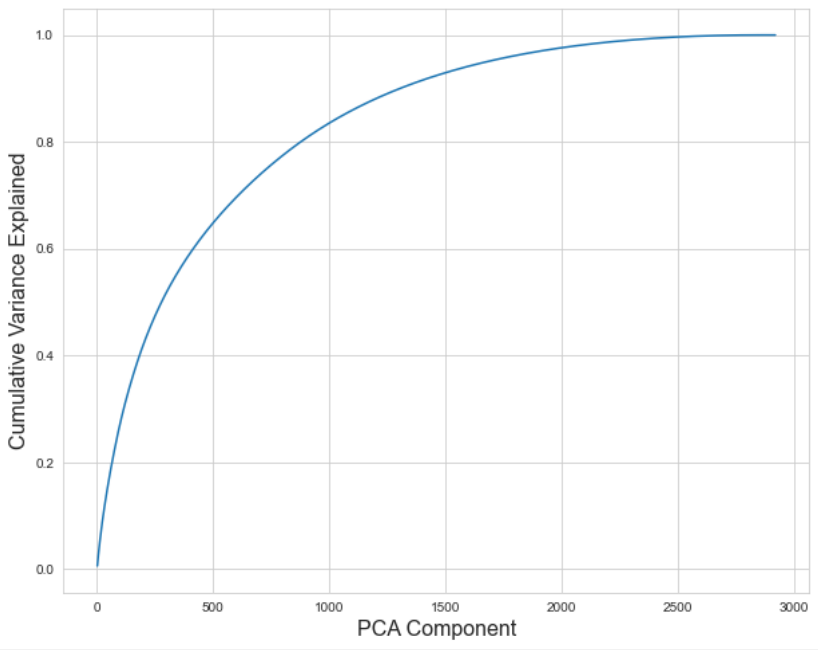
FINAL FEATURE SELECTION

- The only features whose statistics indicate that they can differentiate between the political affiliations are:
  - The tweet text itself (vectorized either by CV or TF-IDF)
  - The NMF topics
  - Sentiment
  - Tweet length (possibly)
- NMF topics must be one-hot encoded to be run through all the models we will use.

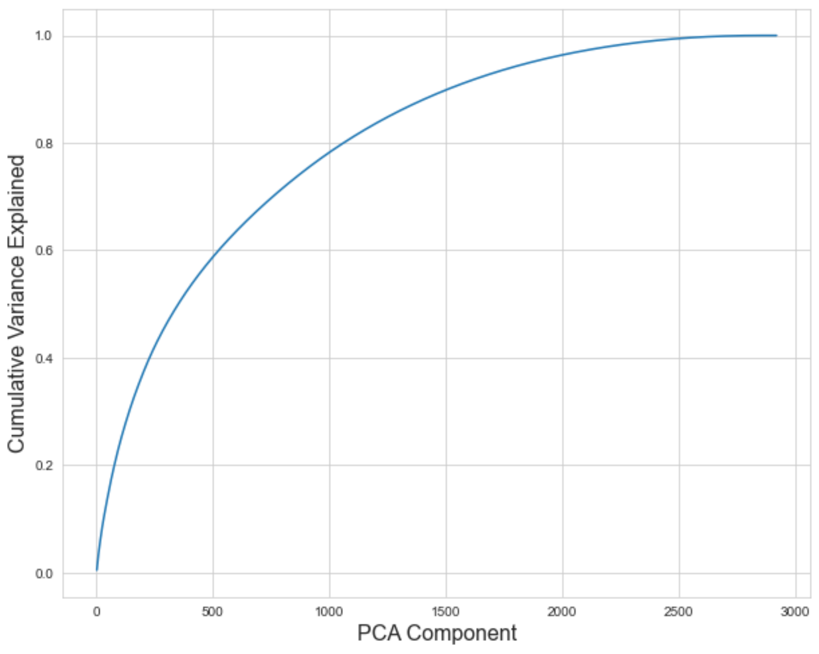
ASIDE: DIMENSION REDUCTION

- Used PCA to check if we could reduce dimension without losing too much variance

PCA on CV-word-bagged data



PCA on TF-IDF-word-bagged data



	Predicted: -1	Predicted: 0	Predicted: 1
Actual: -1	TP(for -1)	FN(for -1) = FP(for 0)	FN(for -1) = FP(for 1)
Actual: 0	FP(for -1) = FN(for 0)	TP(for 0)	FN(for 0) = FP(for 1)
Actual: 1	FP(for -1) = FN(for 1)	FP(for 0) = FN(for 1)	TP(for 1)

- We will compare models using Accuracy, Precision, and Recall.
    - Accuracy = (TP + TN) / (TP + TN + FP + FN)
    - Precision = TP / (TP + FP)
    - Recall = TP / (TP + FN)
- Note that we have 3 classes – so precision and recall don't make sense. We average the precision and recall of each class's predictions using the macro-averaging method.
    - Macro Precision = (Prec1 + Prec2 + Prec3) / 3
    - Macro Recall = (Rec1 + Rec2 + Rec3) / 3

ASIDE: SCORING

PREDICTIVE MODELS:
LINEAR DISCRIMINANT ANALYSIS

	Accuracy	Precision (macro)	Recall (macro)
CV word-bagged full data	77.0%	76.2%	76.6%
TF-IDF word-bagged full data	77.8%	77.1%	77.5%
CV word-bagged PCA data	74.3%	73.6%	74.3%
TF-IDF word-bagged PCA data	74.6%	73.7%	74.5%

- CountVectorizer vs. TF-IDF:
  - TF-IDF word-bagged data scored better than CV word-bagged data.
- Full data vs. PCA-reduced data:
  - There was an acceptable drop in scoring but LDA did not take that long to run with the full data so it's likely not necessary.

RANDOM FOREST

- GridSearchCV used to search for optimal hyperparameters
  - Number of trees: 50, 100, 150, 250, 500, 1000
  - Maximum depth: 1 – 31

	Accuracy	**Precision (macro)**	**Recall (macro)**	**Hyperparameters**
CV word-bagged full data	82.2%	84.5%	80.2%	Trees: 150, max depth: 30
TF-IDF word-bagged full data	82.0%	84.1%	80.3%	Trees: 500, max depth: 30

SVM

- GridSearchCV used to search for optimal hyperparameters
  - Kernel: polynomial or rbf
  - C: 0.1, 1, 10
  - (poly) degree: 1, 2, 3, 4
  - (rbf) gamma: 0.01, 0.1, 1, 10

	Accuracy	**Precision (macro)**	**Recall (macro)**	**Hyperparameters**
CV word-bagged full data	81.3%	81.1%	80.2%	Polynomial, degree 1, C = 10
TF-IDF word-bagged full data	80.6%	80.3%	79.6%	Polynomial, degree 1, C = 10

NEURAL NET (MLP CLASSIFIER)

- GridSearchCV used to search for optimal hyperparameters
  - Shape: (500), (100, 100, 100, 100, 100), (50, 50, 50, 50, 50, 50, 50, 50, 50, 50)
  - Alpha: 0.00001, 0.0001, 0.001, 0.01, 0.1, 1

	Accuracy	Precision (macro)	Recall (macro)	Hyperparameters

CV word-bagged full data	80.9%	80.4%	80.2%	Shape = (500)
				Alpha = 0.01

TF-IDF word-bagged full data	80.9%	80.4%	80.2%	Shape = (500)
				Alpha = 0.01

CONSIDERATIONS

- Choosing model:
  - Concerns: getting high scores & over-fitting.
- Over-fitting:
  - Most models have at least one hyperparameter chosen by GridSearchCV which tends towards over-fitting. 5-fold cross-validation should combat this, but the extent to which this helps is unclear.
- Proposed solution: compare each model, with said over-fit/regularization parameters chosen as sub-optimally as possible while maintaining decent scores.
- Which score to optimize for: in this setting, precision and accuracy make the most sense. We don't mind false negatives, but false positives are more detrimental.

CONSIDERATIONS

- We choose SVM with rbf kernel, C = 1, and gamma = 0.01.
    - The rbf kernel has been shown to generalize to more data better than linear/polynomial SVM kernels. C in this case is the default 1, and thus isn't likely so high that over-fitting occurs.
    - We do not choose LDA as it is another linear decision-boundary model and makes more assumptions on the distribution of the data than SVM.
    - RFC is dropped because even with a maximum depth of only 15, precision is lower than SVM and random forests are very easy to overfit. It is impossible to tell if the max depth of 15 is sufficient to prevent over-fitting (without more data).
    - MLP Classifier is also dropped because of how easily neural nets are over-fit.
- We further lower the variance of our fit (and possibly increase bias/error) by using a Bagging ensemble method with 50 of the above learners.

RESULTS

- SVM with C = 1, kernel = 'rbf', and gamma = 0.01:

	Accuracy	Precision (macro)	Recall (macro)
CV word-bagged full data	76.0%	86.7%	72.8%
TF-IDF word-bagged full data	76.2%	87.0%	72.9%

- BaggingClassifier on the above SVM with 50 estimators, max sample 75%:

	Accuracy	Precision (macro)	Recall (macro)
CV word-bagged full data	74.0%	86.3%	70.6%
TF-IDF word-bagged full data	74.4%	86.6%	70.9%

EXAMPLE OF TEST DATA

- Index 5920, outside of training data
- Text: "Bizarre to see Trump &amp; Co. all shocked about losing the election, when these are the same people who broke a Supreme Court confirmation process and rushed a nominee through in record time because they thought they would...lose the election."
- Predicted (CV) Affiliation: -1
- Predicted (TF-IDF) Affiliation: -1

Seems to work for this example!

FUTURE INVESTIGATION

- Use Semi-supervised learning (e.g., LabelSpreading) to boost training data and lessen the detriments of over-fitting and small training data sets.
- Test different models such as Quadratic Discriminant Analysis, Naïve Bayes Classifier.
- Test the model on new tweet data – is it robust to topic and sentiment changes over time?
- Do the predicted Affiliation values have a similar distribution to the training data?
- Investigate more of the statistics of the readability scores.
- Validate hand-labeling using k-means clustering, k = 3.

QUESTIONS