EEUU POLITICAL PARTY CLASSIFICATION WITH NLP AND TWITTER

FINAL THESIS
MASTER IN DATA SCIENCE

Alejandro Ferrero Briones
Madrid, July 2022

Index

1 – Introduction and motivation

NLP (Natural language processing) is a branch of artificial intelligence which deals with bridging the machines understanding human's natural language. Natural language processing has its roots in the 1950s, but has acquired a lot of popularity in the last decades for two main reasons:

1. Computational capacity mostly thanks to cloud services
2. Accessibility to huge sets of data

NLP is a powerful tool that can be used for good or bad purposes. And if we mix it with the largest text dataset of the world which is twitter and a sensible topic such as politics, this duality becomes more interesting.

Twitter has become the best way for politicians to interact with people and it's an extremely different interaction than back in the day when there was no internet. Politicians can read in real time people's opinion about everything, can answer them… nothing to do with TV in the 90s. Also, it has become the easiest way for people to communicate with each other. But this has some drawbacks, for example, radicalization and segregation between groups. Also, there is the debate of freedom of speech: Does it has limits? Was it legitimate to ban Ex-president Donald Trump from twitter while there are active terrorist accounts posting radical tweets every day? If NLP algorithms can be used to study political trends by geographical areas, is it ethical to increase electoral propaganda in some specific territories? Or to some specific users online?

Personally, these two topics have been of my interest since the rumors of election manipulation in EEUU in 2016, and it has been increasing due to the quantity of fake news all over the internet. Fake news detectors have become and indispensable tool for social media companies to control the spread of misinformation, as well as other NLP tools such as hate speech detectors. Originally, this was going to be topic of this project, however there are lots of projects online about the same topic, and I wanted to do

something a little bit different. Yeah, all of them are binary classification problems, but I wanted to give it my personal touch.

The aim of this project is to build a model than can differentiate a tweet between democrat or republican. Tweets from EEUU members of congress will be extracted from twitter, cleaned and used in a model to perform this prediction.

2 - Data and previous work

In order to use a twitter wrapper for python (in this case it's going to be *twithon*), you need twitter developer credentials. You can get them by logging in the twitter developer portal and upgrading your account to elevated, so that you have a bigger monthly tweet cap usage.

For the purpose of the project, we will use tweets from members of congress for two main reasons: users are easy to identify, and tweets are easily labeled. It is a fact that it would be better to use normal people's tweets, but we would need a manually labeled dataset.

Congressional Social Media Handles » Triage Cancer | Finances-Work-Insurance contains a table with all members of the congress and with all the information needed from them, such as their twitter account and their party. This table is included in the repository, inside data/eeuu_member_of_congress.csv.

Twython is an actively maintained, pure Python wrapper for the Twitter API. It provides an easy way to access twitter data, whether it's batch or streaming. In this project we will do a batch extraction. However, it's got some limitations: it can only loop over a user's timeline 15 times, so we will be extracting only the most recent tweets from users.

Twython requires 4 keys to authenticate via client: API Key, API Secret, Access Token and Secret Access token, all of them available via twitter developer portal. In the repository, keys are stored in a local file and exported as environment variables so that nobody can steal them.

Code in src/twitter_extractor.ipynb does the job of looping over every member of the congress, extracting data and storing it in data/ folder. Columns in the dataframes are:

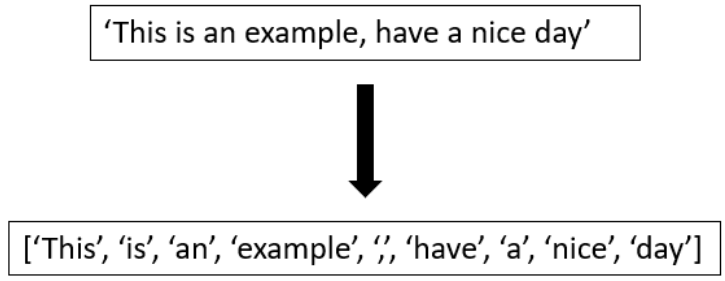
- Id: unique ID from the tweet
- User: user identifier
- Created_at: date in which the tweet was posted
- Text: text in the tweet
- Lang: language in which the tweet is written
- Retweet_count: Count of users who have retweeted the tweet
- Favorite_count: Count of users who have liked the tweet

These two last columns are not going to be relevant for the project itself, but I decided to keep them as they can be used for some future work such as predicting the impact a tweet has.

3 - Data engineering

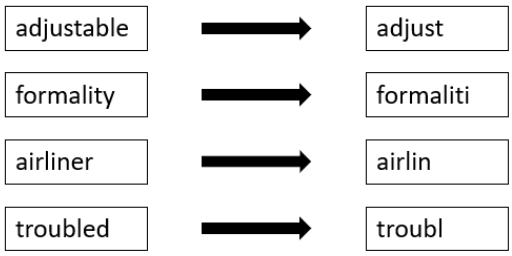
This phase consists of extracting all the valuable information from raw text. The techniques that have been applied are:

- Use regular expressions to delete useless text: regular expressions are a sequence of characters that specifies a search pattern in text. They are really useful in the project to delete old style rt, hyperlinks, hashtag signs, mentions and skip lines.
- Stopwords and punctuation signs: these are completely useless for the models, so we must get rid of them.
- Tokenization: Tokenizers break unstructured data and natural language text into chunks of information that can be considered as discrete elements.

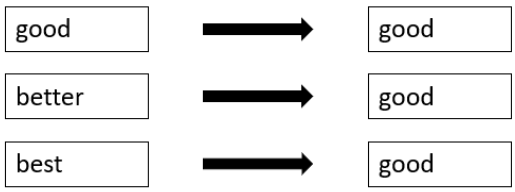


- Steaming: Steaming is process of reducing inflection towards their root forms. This occurs in such a way that depicting a group of relatable words under the same stem, even if the root has no appropriate meaning. There is another approach for this, which is lemmatization. Lemmatization refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.

Steaming:

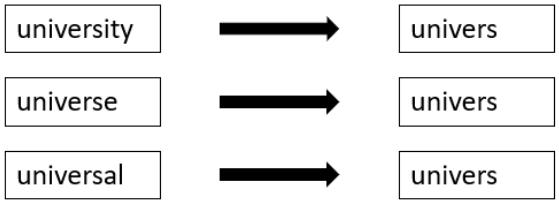


Lemmatization:

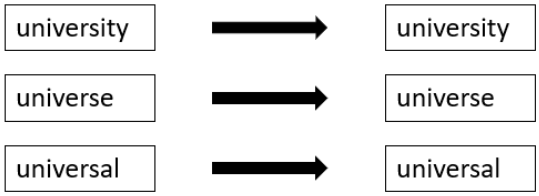


From this example, one might think that lemmatization is better. However, if we select another set of words:

Steaming:



Lemmatization:

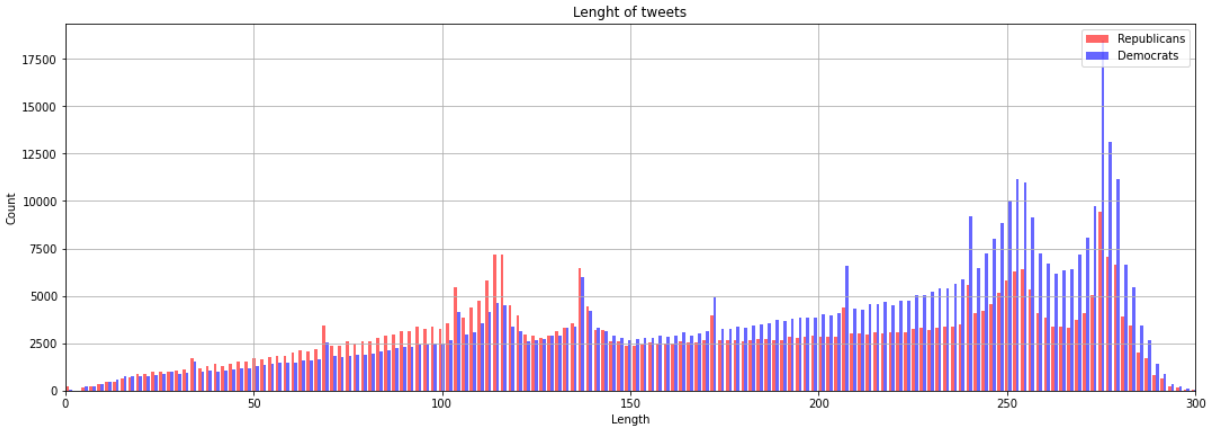


In this case, I chose steaming as the meaning of words is not important.

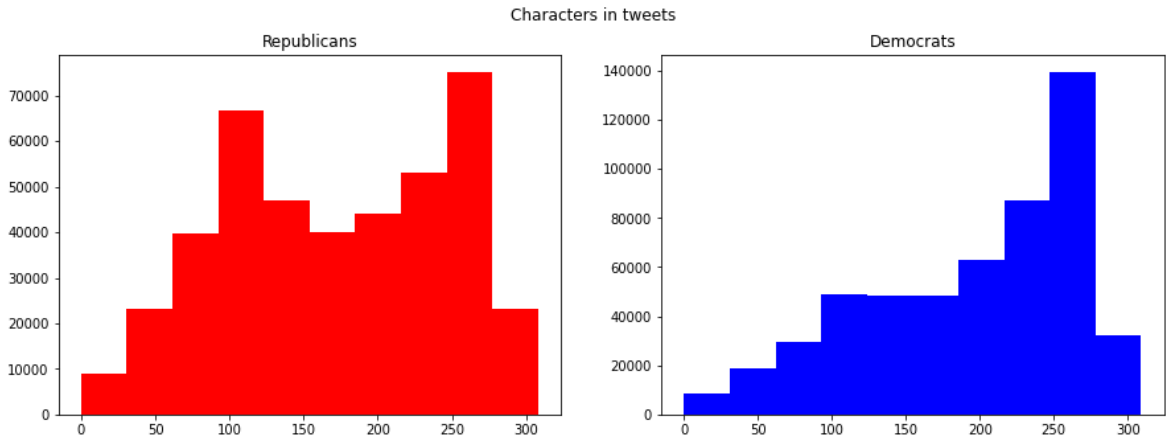
- Transform label column to integer values.

4 - Data visualization

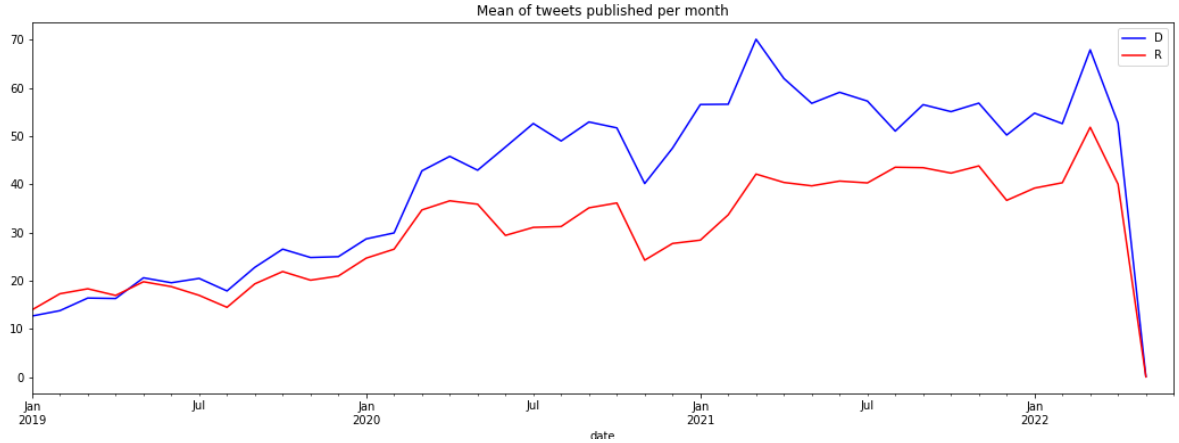
First, we must bear in mind that there are 421232 tweets of 250 republican users and 524016 tweets of 270 democrat users, so in some visualization we shouldn't be focusing on the numbers but on the tendency.



Democrat users tend to write longer tweets than republican users.

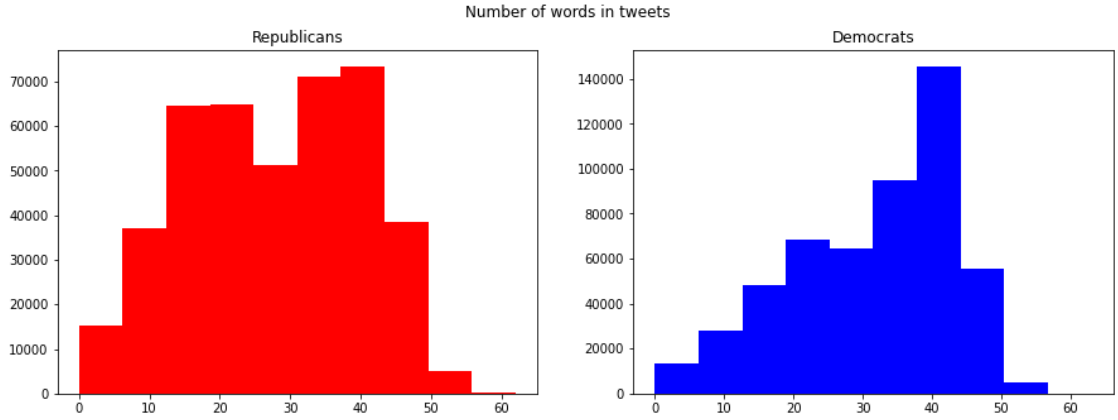


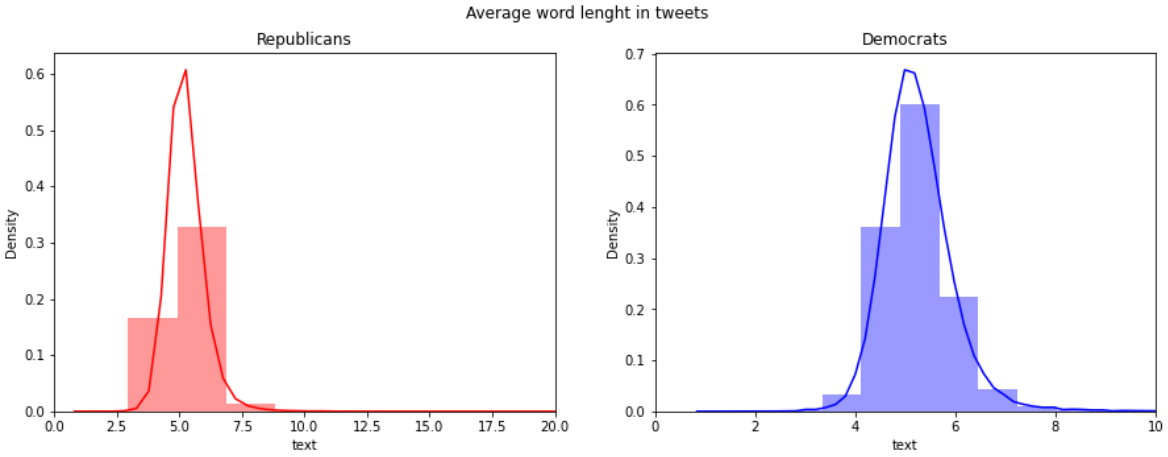
As we could expect, Democrats tweets contain more characters than republican tweets.



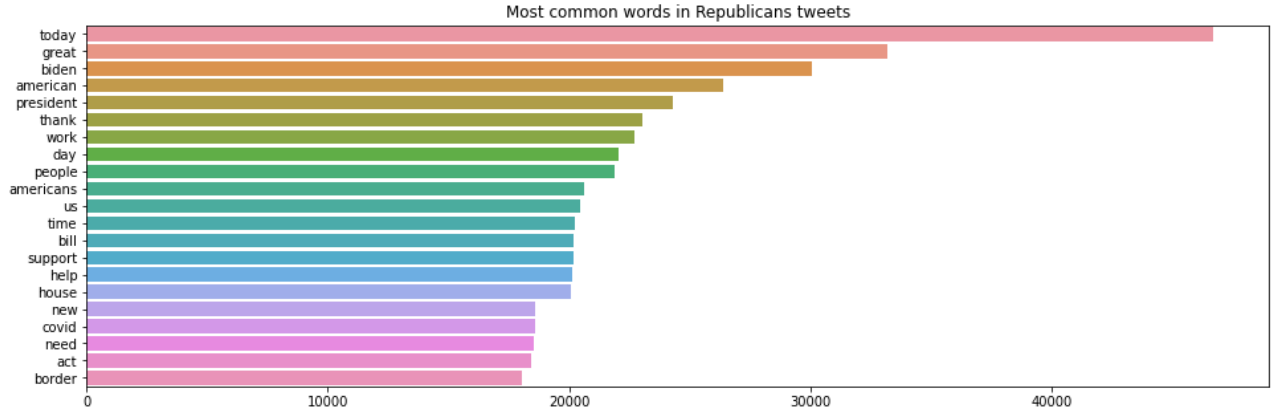
As we are extracting the most recent tweets, it's normal to see an increase in the number of tweets published. However, this may not be true, as users with more recent activity could dismantle this theory.

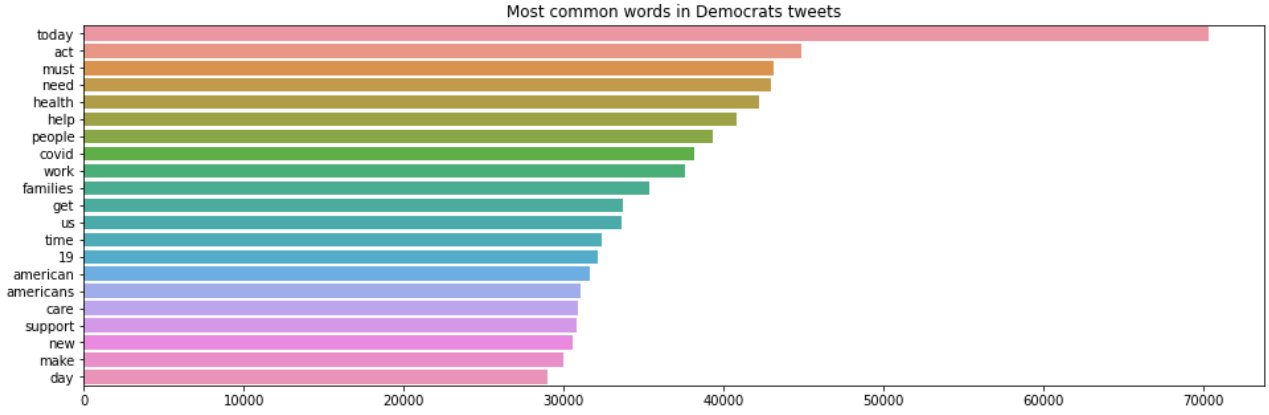
There is a clear increase in the number of tweets in the beginning of 2021. This matches with 2021 EEUU elections and everything related to it, such as the Capitol riot and conflicts with the ex-president Donald Trump.





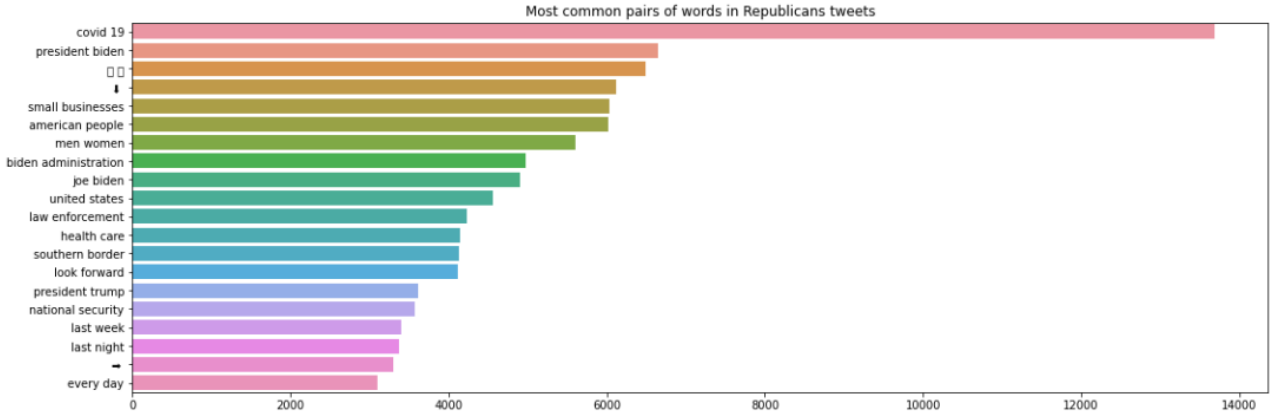
These two last plots only strengthen the affirmation that republican users write shorter tweets than democrat users.

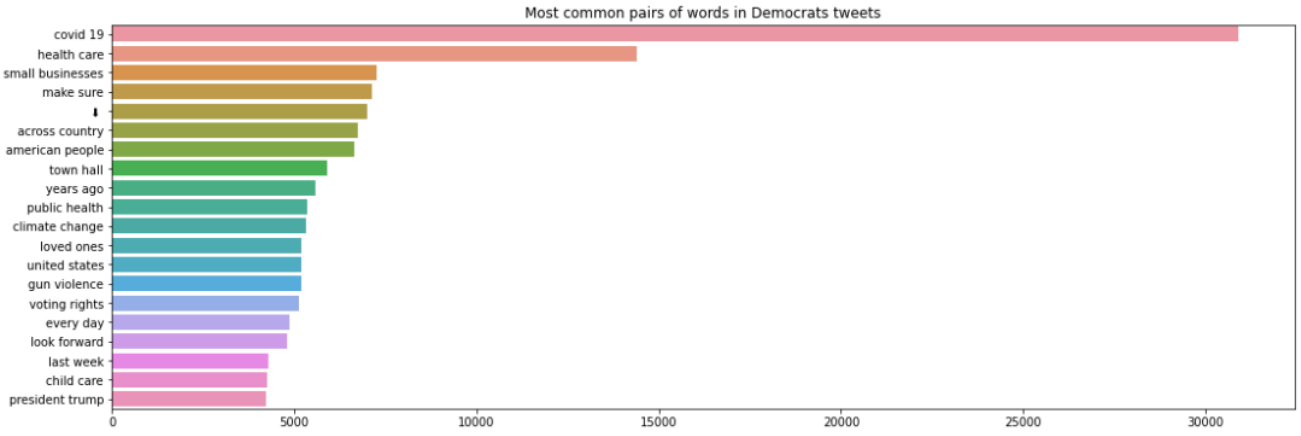




We can see some interesting stuff from these plots:

- Republicans write a lot about President Joe Biden
- Democrats write much more about health than Republicans.





- Most written bigram is covid 19, which is completely normal considering the period we are analyzing and the troubles covid 19 has caused.
- Republicans talk more about President Biden and Ex-president Donald Trump than democrats.

- Both groups talk about health care but only democrats talk about public health. Actually, it's the party that fights for a free public health and for a change in EEUU's health system.
- Democrats talk a lot about gun violence and climate change, while Republicans don't. Speaking about the first one, The Second Amendment to the United States Constitution protects the right to keep and bear arms. It is a very controversial topic nowadays, above all among democrats, and strongly supported by republicans.
- Those squares we can see on third place on the most common pair of words in republican tweets are 'U S', we can see it in the notebook. It does not appear in the Democrats top 20. Republicans tend to be more patriotic than democrats, so it makes sense.

In fact, these conclusions in general are the expected ones. Politics are a sensitive subject, so I let the reader come to subjective conclusions from those results.

5 – Classification algorithms

5.1 - Feature extraction

TF-IDF is the first step that must be executed before trying out model performance.
TF-IDF stands for term frequency-inverse document frequency. It is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.
It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling.
The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

- Term Frequency: This summarizes how often a given word appears within a document.
- Inverse Document Frequency: This downscales words that appear a lot across documents.

The formula is as follows:

$$tfidf_{i,j} = tf_{i,j} * \log\left(\frac{N}{df_i}\right)$$

being:

- $tf_{i,j}$ = total number of occurrences of i in j
- df_i = total number of documents containing i
- N = total number of documents

5.2 – Model comparation

Machine Learning Classifiers Algorithms used in the project are:

- Logistic regression
- Random forest
- XGBoost
- BERT

SVM (support vector machine) models were discarded as they tend to be really slow with big number of samples.
Every model except from BERT has been executed with a grid search for better hyperparameter selection.

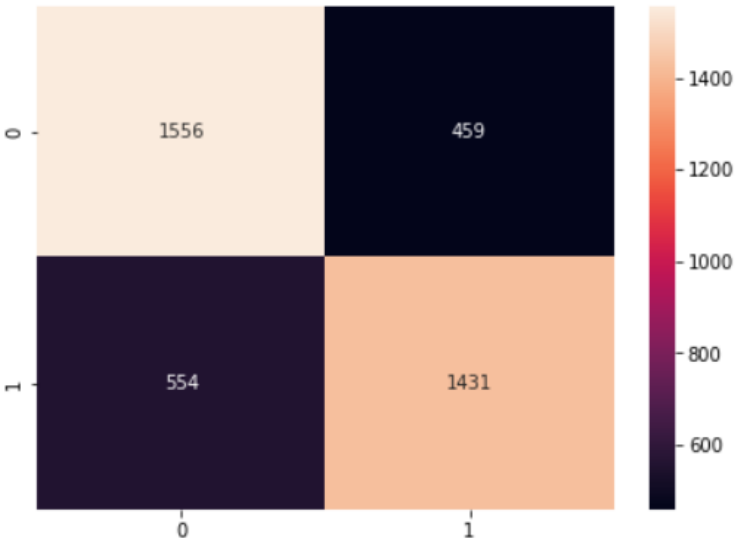
The next table represents the metrics obtained by each model, using a sample of the original dataset of 10000 tweets:

Model	Accuracy	Precision	Recall	F1	ROC AUC SCORE
Logistic regression	0.746	0.747	0.746	0.746	0.827
Random forest	0.699	0.7	0.699	0.699	0.784
Xgboost	0.699	0.7	0.699	0.699	0.78
BERT	0.75				

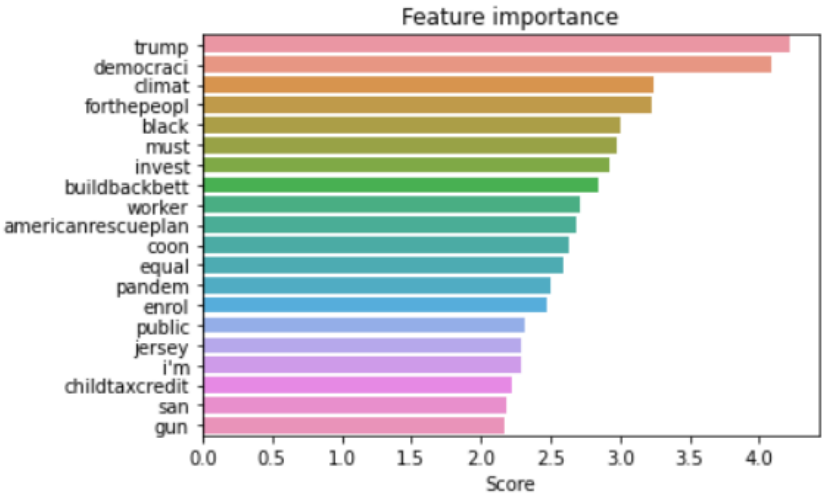
DISCLAIMER: This numbers may vary, because models are using a random subset of the dataframe of length 10000.

Considering that there is a 92% of correlation between the two groups, these are not bad results at all.

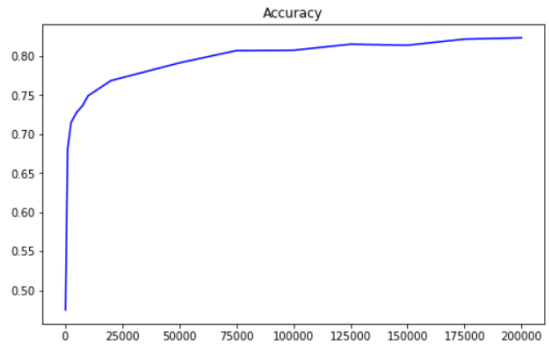
Logistic regression was the model with the best performance in all metrics, including execution time. It's confusion matrix shows that the proportion of false positives and false negatives is similar, which is also great as predictions are balanced.

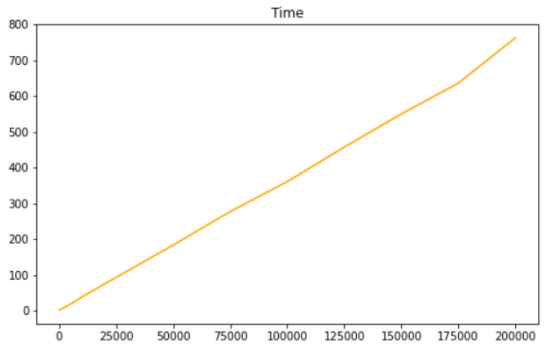


Evaluating the feature importance of the logistic regression, we can see the top words with the most predictive importance:



In terms of accuracy vs time of execution, while time grows linearly, accuracy starts to decrease its improvement after a few tweets:





With this in mind, maybe the best number of tweets for the dataset is around 25000, of course with the same computational capacity. Everything was run on google colab's free tier.

6 – Frontend

There is a simple frontend created with streamlit library. The code is inside the repository in frontend/. Streamlit is a pure python library that turns data scripts into shareable web apps in minutes. No front-end experience required.
The aim of this web is to try your own tweet and see the classification the model does with its confidence of the prediction.

This is done thanks to the models stored in the repository in model/, which are in pickle format. Pickle is a really useful Python tool that allows you to save your models, to minimize lengthy re-training and allow you to share, commit, and re-load pre-trained machine learning models.

In order to execute it, download the repository to a linux environment and executing the following commands in the CLI:

cd twitter-category-nlp/frontend
streamlit run app.py

It will open a web in your localhost like this:

There you can just play with your own tweets:

Political party classification with tweets

Try it yourself!

Enter your tweet

We must fight for a public and free health care system

Submit

Your tweet is Democrat with a confidence of 93.10%

Political party classification with tweets

Try it yourself!

Enter your tweet

Make america great again

Submit

Your tweet is Republican with a confidence of 87.24%

7 – Conclusion and future work

The execution of this project has been really interesting not only for the results themselves, but for everything I've learned about natural language processing and classification algorithms.

This case it's not as if we were trying to classify between big groups of users, such as music, politics, sport… (this was one of my first options for this project). That would have been much easier, as they use very different language and accuracy would have been higher. Generally, politicians talk about the same topics, and use a similar formal language, so it's normal to not expect a really big accuracy in the classification models.

Some future work I find interesting would be:

- Build a batch ETL in the cloud to train models at night with all the data we want and all the hyperparameters we would like to try in grid searches
- Spend a lot of time on deep learning algorithms as the must show better results than I could.
- Somehow extract location of tweets and show in a map the tendency per region in the EEUU

Finally, last weeks I've been thinking of a way I could use everything I've learned from this project for good. Recently, there has been many shooting in public places in the United States. It is a fact that many of these shooters use twitter to post their ideas related to guns, white supremacy... and show increasing radicalization through the years. So, it would be amazing to train a model to try to detect this kind of profiles online. I would have executed this idea for the project, but it was a really recent idea I came up with. It is what it is.

7 – References

- Congressional Social Media Handles » Triage Cancer | Finances-Work-Insurance
- twython · PyPI
- Use Cases, Tutorials, & Documentation | Twitter Developer Platform
- Random Forest python (cienciadedatos.net)
- Gradient Boosting con python (cienciadedatos.net)
- Machine Learning con Python y Scikitlearn (cienciadedatos.net)
- Creating a TF-IDF in Python. From scratch in python code | by Iftekher Mamun | Medium
- Dimensionality Reduction in Python with Scikit-Learn (stackabuse.com)
- https://www.tensorflow.org/text/tutorials/classify_text_with_bert
- https://www.geeksforgeeks.org/a-beginners-guide-to-streamlit/
- https://machinelearningmastery.com/save-load-machine-learning-models-python-scikit-learn/

…and a lot of stackoverflow, medium and Kaggle.