Rachel Lui Win Tan

Natural Language Processing for the Social Sciences

Final Report

QMSS GR5067 Final Project Report: Sentiment Analysis using Natural Language Processing in Python on #Brexit Tweets

Abstract

This project uses natural language processing in Python to analyze tweets containing the hashtag '#Brexit' posted on 21 June 2016, in order to better understand twitter sentiment two days before the United Kingdom European Union membership referendum. The project conducts exploratory data analysis and cleaning, before training and testing an initial Random Forest Classifier on a general corpus of tweets labelled according to 'positive' and 'negative' sentiment. Based on the test and validation performance, this project trains two additional classifiers – Random Forest classifier and Support Vector Machine, both using Principle Component Analysis dimension reduction and a grid search. The models are then used to predict on the target dataset of #Brexit tweets. The predictions classified a large majority of #Brexit tweets as positive, and possible explanations for this result is detailed in the analysis.

Introduction

The United Kingdom European Union membership referendum, or better known the Brexit Referendum, took place on 23 June 2016. The vote result saw 51.9% of the country choosing to leave, starting a lengthy, ongoing process of Britain's withdrawal from the EU. In the weeks leading up to Brexit, both Leave and Remain campaigns used social media such as twitter as platforms for advocacy. Twitter data is suitable for natural language processing (NLP) since it consists of tweet text (maximum 280 characters in length), while including elements such as hashtags (#) or mentions (@) that can be extracted to identify key issues and people of interest. This project uses NLP in Python to analyze tweets containing the hashtag '#Brexit' posted on 21 June 2016, so as to better understand twitter sentiment two days before Britain's decision to leave the European Union.

The overall architecture of the analysis includes the following steps: pre-processing, tokenization, lemmatization with parts of speech tagging, vectorization, dimension reduction, training, testing, validation and classification. I use training data and target #Brexit data obtained from Kaggle.com and conduct exploratory data analysis with visualizations through word clouds. I then proceed to train a simple Random Forest classifier, followed by two improved models – an adjusted Random Forest classifier and Support Vector Machine, both using PCA dimension reduction and a grid search. The models are used to predict on the target dataset of #Brexit tweets.

The project is useful as an investigation into whether twitter users were using positive or negative words in tweets related to Brexit just before the vote. Moreover, it serves as an effective application of natural language processing and machine learning on real-world data.

Data

Training Dataset: Twitter Sentiment Analysis

This dataset was retrieved from a Kaggle competition (Kaggle, 2017). I used the train.csv file that contains a corpus of 100,000 tweets, not-specific to any topic, labelled either positive (1) or negative (0) in terms of sentiment. Due to technical and time constraints, I used a sample of 10,000 of the tweets as the training dataset.

Target Dataset: Brexit Opinion Data

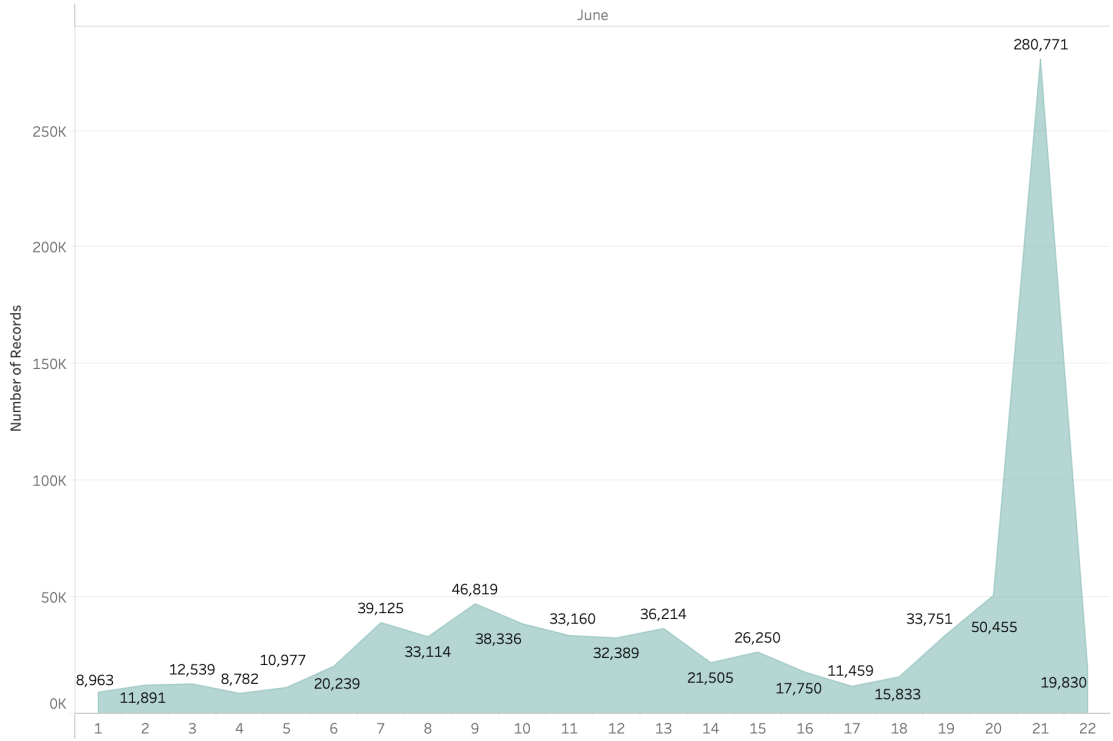
This dataset was a user-created dataset from Kaggle. (Dossa, 2018). I used the total2.csv file that contains tweets scraped from December 2009 to 22 June 2016, the day before the Brexit referendum.

The full dataset contains

1. Tweets that all contain the hashtag #Brexit (variations for upper and lower case inclusive)
2. 1,512,728 tweets organised into 10 columns with corresponding information about the tweet:
    - Twitter username (username)
    - Date the tweet was posted (date)
    - Number of retweets (retweets)
    - Number of favorites (favorites)
    - The text of the tweet (text)
    - Location of tweet (geo) – empty column
    - Number of mentions symbolised by @ (mentions)
    - Number of hashtags symbolised by # (hashtags)
    - Twitter id tag of the tweet (id)
    - URL link to tweet (permalink)

In general, the number of #Brexit tweets posted per day before 2016 were relatively low as Brexit had not gained as much attention then, and there were more #Brexit tweets for the days after February 2016. This is because the Brexit referendum date was announced on February 22 (Blitz, Brunsden & Hughes, 2019). In total there were 919,662 tweets about Brexit posted in 2016. I looked at tweets dated June 2016 (the month of the Brexit referendum) and visualized the number of tweets gathered in a graph below:

Number of #Brexit gathered tweets per day in June 2016

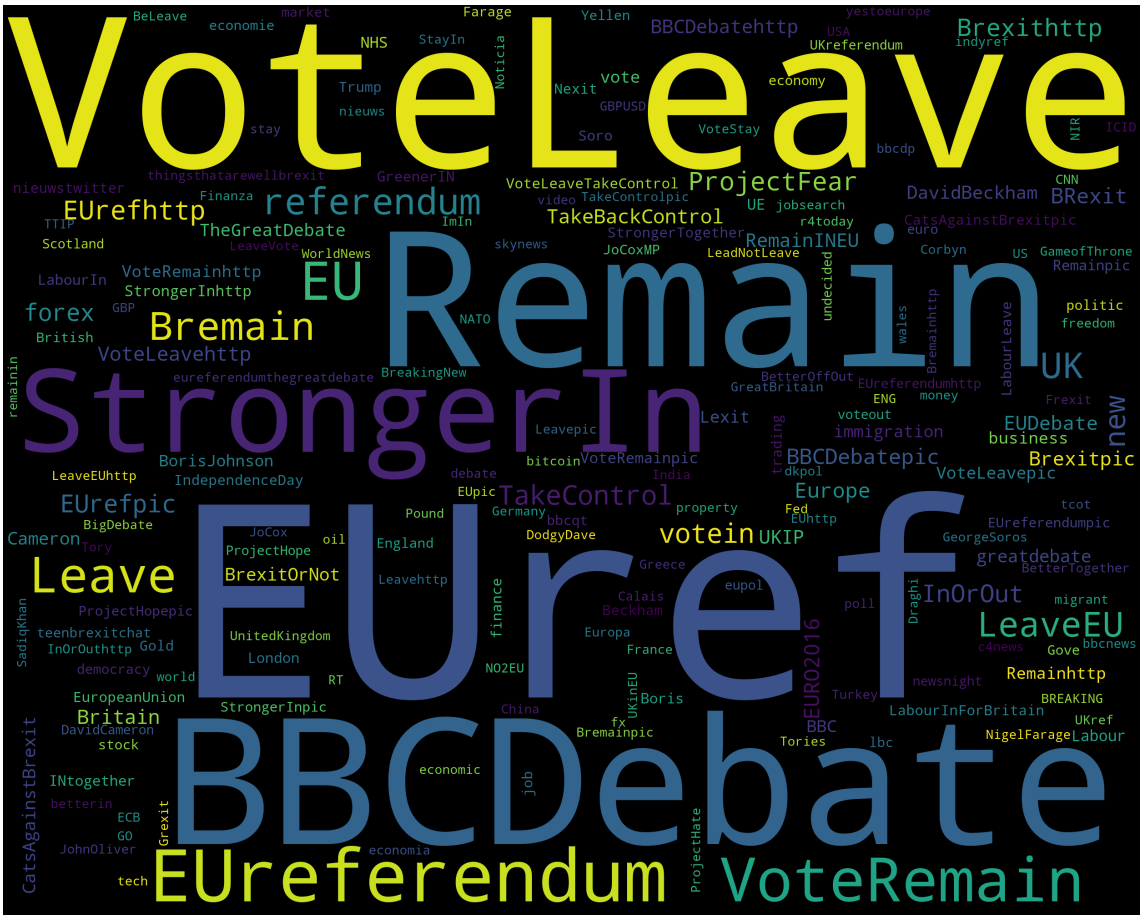


There is an increase in the number of #Brexit tweets from 18 June onwards, with the number rising sharply between 20 and 21 June. Tweets were only gathered until 02:00 AM on the 22 of June, hence only 19,830 tweets were gathered on that date. For this project, I chose to focus on the tweets gathered on the 21 June 2016, the date with the most tweets posted about Brexit.

Word Cloud Visualizations

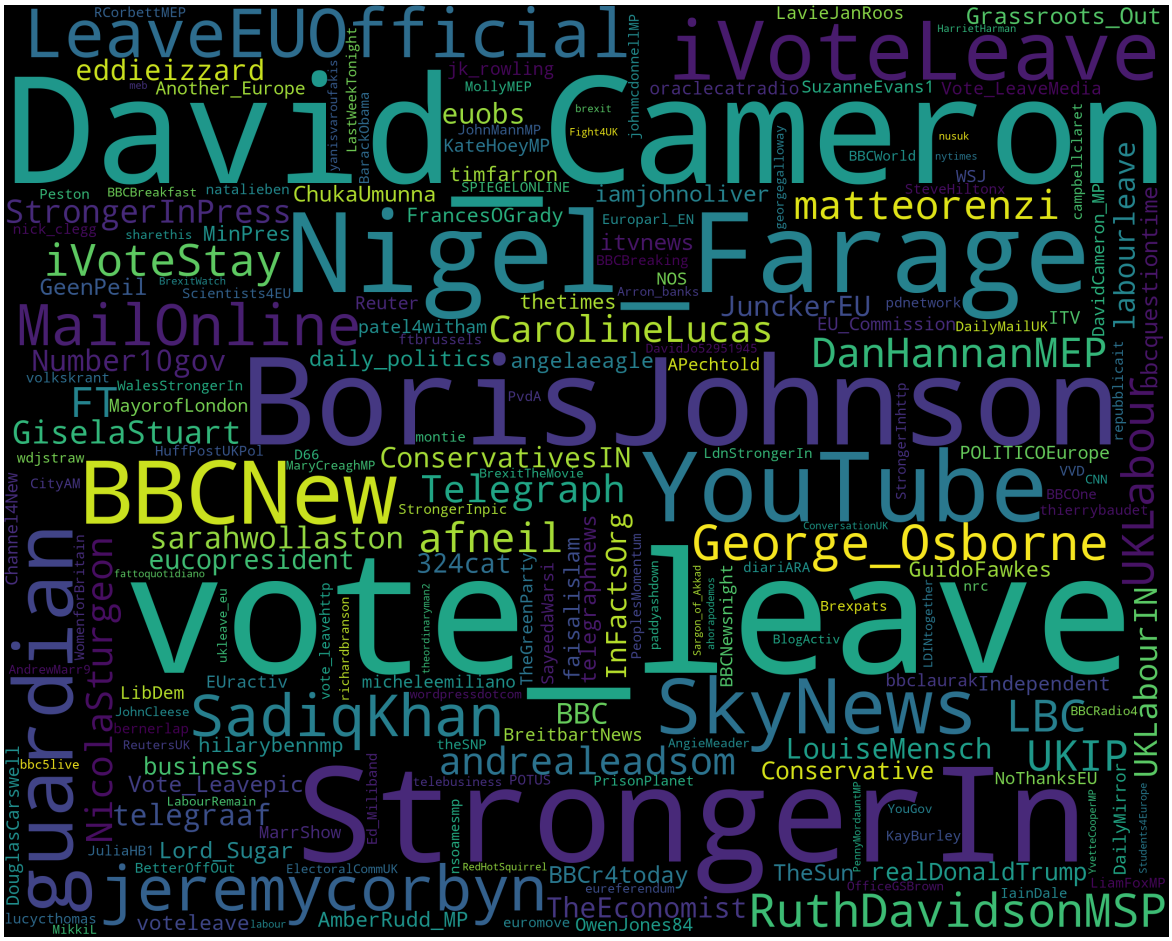
From the tweet text, I extracted all the hashtags (#) using re.findall() and regex formula. I then plotted a word cloud of the most popular hashtags used. I removed the hashtag #Brexit and its variations since that hashtag would be found in all the tweets.

Hashtag Word Cloud



I also extracted all the mentions (@) and plotted a word cloud of the most popular people mentioned in the tweets.

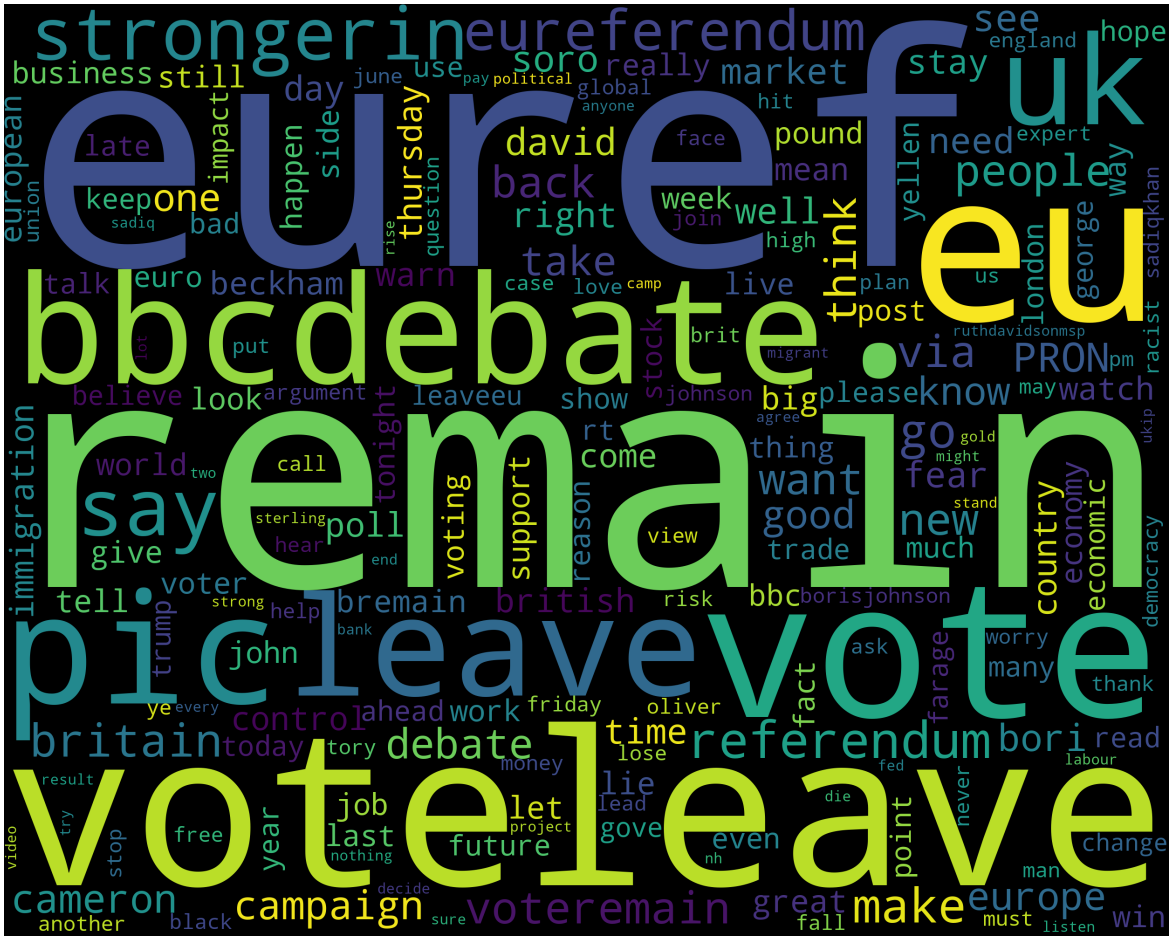
Mentions Word Cloud



After cleaning the tweets (see methodology), I plotted a word cloud of the most common words used in tweets on 21 June 2016. Due to the format of tweets, where users tend to use hashtags and mentions in place of nouns, I did not exclude hashtags and mentions from the word cloud. An example of a typical tweet can be seen below:

> “Good read frm @VanessaBNI @newint My #Brexit anxiety grows every night, seems I'm not alone #BeAfraid & vote #Remain https://newint.org/blog/2016/06/20/eu-referendum-fear-is-the-right-thing-to-feel/ …”
>
> - @DanversKate

Common Words Word Cloud



Methodology

Initial Data Preparation

Target Dataset:

Removing non-useful data

I removed the columns that were unnecessary for my sentiment analysis, keeping only the text column containing the main text of all the tweets. After removing duplicate tweets and missing data, there were 207,449 tweets in my target dataset.

Non-English tweets

Upon looking closely at the target dataset, I then realized that there were tweets in other languages about Brexit. As such I attempted to remove most of the non-English tweets, which seemed to be mostly Spanish, French and German language tweets. I created a list of common words or phrases (' la ', ' le ', ' el ', ' que ', ' del ', ' cette ', ' cet ', ' es un ', ' es une ', ' dans ', ' du ', ' de ', ' ist') and removed the tweets that contained these words. This left me with 191,344 English tweets.

Training Dataset:

I used a sample of 10,000 tweets from the train.csv file and converted the "Sentiment" and "SentimentText" columns into arrays, as y_data and X_data respectively.

Pre-processing of both Target and Training Datasets (columns containing the text of tweets)

I cleaned the tweets with a for loop that

- Made all the tweets lowercase
- Removed all the URLs mentioned in the tweets
- Removed special characters and numbers

- Removed single characters in the tweets
- Replaced multiple space characters with one space
- Removed stop-words in the tweets

Tokenization and Stemming, Lemmatization, POS

I tokenized the tweets using the .split() function and appended them to list, before using spaCy to lemmatize the tweets. I used an nlp.pipe with batch_size = 20, disabling “ner” and “parser” in order to run the lemmatization more efficiently, while keeping the “tagger” component to enable POS-tagging.

Vectorization

I used TF-IDF Vectorizer to transform the the tweets into arrays, extract the feature names and transform the tokenized tweets into X-matrices so that they could be used with my machine-learning models.

Dimension Reduction/ Train/ Test/ Validation

I trained 3 different models on my vectorized training data (10,000 tweets).

Model 1: Random Forest Classifier (n_estimators = 200)

At first, I used train-test-split 80/20%, training the model on X_train and y_train set and predicting X_test. Upon comparing the predictions with y_test I got an accuracy of 72.0%. I also generated the following classification report:

	precision	recall	f1-score	support
0	0.73	0.58	0.64	880
1	0.71	0.83	0.77	1120
accuracy			0.72	2000
macro avg	0.72	0.70	0.71	2000
weighted avg	0.72	0.72	0.71	2000

However, I acknowledged that train-test-split may not be the most thorough way of testing the model and therefore tried a 5-fold cross validation (CV = 5) on the training data and got an average accuracy of 72.5%, which verifies the accuracy of my predictions.

Model 2: Random Forest Classifier with PCA dimension reduction and a grid search

The target variance used for this model was 95% with a 5-fold cross-validation and the following grid search:

```
param_grid = {'max_features': ['auto', 'sqrt'],
              'max_depth': [1, 2, 5, 10, 20],
              'n_estimators': [100, 200, 500],
              'random_state': [0]}
```

The minimum number of PCA components was 91, with an explained variance ration of 0.138. The best accuracy from the gridsearch was 63.6% with the optimum parameters being max_depth = 20, max_features = 'auto', n_estimators = 200.

Model 3: Support Vector Machine with PCA dimension reduction

The target variance used for this model was 95% with a 5-fold cross-validation and the following grid search:

```
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10] , 'gamma' : [0.001, 0.01, 0.1, 1]}
```

The minimum number of components of the PCA was 91, with an explained variance ration of 0.138 (using the same dimension reduction function as for Model 2). The best accuracy from the gridsearch was 69.8% with the optimum parameters being C = 10, gamma = 1.

Evaluation

The difference in accuracy between Model 1 and Model 2 could be due to the fact that PCA does not guarantee better performance of the model, although it helped cut down on processing time. Using a 0.95 explained variance inevitably resulted in some information being lost from the input. Moreover, the parameter 'max_depth' was

not present in Model 1 and may have limited the model's performance. Finally, given that the SVM Model 3 also used PCA dimension reduction, this may have resulted in information loss. However, Model 3 yielded a slightly more accurate performance than Model 2. However, it is important to acknowledge that although the accuracy of the Model 2 was the lowest, it could still be useful because limiting max_depth could prevent overfitting of the model.

Classification

Based on the models with the highest accuracies I chose Model 1 and Model 3 and used them to predict on my cleaned target dataset (191,344 #Brexit tweets) I obtained the results as follows:

- Model 1: 24998 tweets labelled negative and 166346 labelled positive (13% negative, 87% positive)
- Model 3: 16245 tweets labelled negative and 175099 labelled positive (8.5% negative, 91.5% positive)

The different models do have an effect on the performance on the predictions (a 4.5% difference in the negative and positive labels), however what is clear from the classification is that a large proportion of the tweets in the target dataset are positive as opposed to negative. The first possible reason for this result is that leading up to the Brexit vote, many of the tweets were phrased as encouragements for people to vote, either to leave or to remain. Hence rather than being classified as negative criticism or positive praise for or against Brexit, many of the tweets could be classified as positive encouragement for people to vote in general. Secondly, the dataset contains many tweets from either the remain or leave parties that are promoting their position in the debate. As such, the tweets could be classified as positive as they are

promoting the benefits of either the leave or remain campaigns. Thirdly, leading up to the vote, many people expressed excitement regarding the outcome and hence may use positive sentiment words in their tweets.

Conclusion

It is important to acknowledge that my analysis is subject to certain limitations. In terms of data quality, there was a lack of information about the datasets from Kaggle including how they were collected or whether it was a full sample or a partial sample. More over the quality of the data (filtering for only English tweets during data collection) could be improved. Furthermore, given the time and computing power, training the models on a larger corpus of text, even the full dataset with 99,989 tweets would probably improve the mode. The training dataset also consists of a general corpus of tweets, and the results of the analysis might change should the training dataset be limited to tweets specifically about Brexit or British politics. It would also have been interesting to gather tweets from 22 June onwards, or during different key points in the Brexit saga, in order to plot a time series of sentiment changes. Finally, although parameter fine tuning and model selection was not within the scope of the class, more research could have been done on feature engineering, selecting the hyperparameters used in the grid search or how to optimize the models. Time permitting, it would also be helpful to consider a wider range of machine learning models in order to conduct a comparison of their performances.

This project was definitely useful in expanding my understanding of an end-to-end natural language and machine learning project. The word clouds enabled me to better understand the data, and hence the concerns and interests of twitter users about #Brexit on 21 June 2016. The models then provided information on the

sentiment of tweets during that day. This provided a starting point in understanding whether twitter users were using positive or negative language to talk about Brexit, and inferring the reasons for the results.

Bibliography

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. Sebastopol: O'Reilly Media, Inc.

Blitz, J., Brunsden, J., & Hughes, L. (2019). Brexit timeline: key dates in the UK's divorce from the EU | Financial Times. Retrieved 18 December 2019, from https://www.ft.com/content/64e7f218-4ad4-11e7-919a-1e14ce4af89b

Dossa, O. (2018). Brexit Opinion Data. Retrieved 17 December 2019, from https://www.kaggle.com/natmonkey/brexit-opinion-data/activity

Koehrsen, W. (2018). Hyperparameter Tuning the Random Forest in Python. Retrieved 18 December 2019, from https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

Malik, U. (2019). Twitter Sentiment Analysis Using TF-IDF Approach - GoTrained Python Tutorials. Retrieved 10 December 2019, from https://python.gotrained.com/tf-idf-twitter-sentiment-analysis/

Reddy, V. (2018). Sentiment Analysis using SVM. Retrieved 17 December 2019, from https://medium.com/@vasista/sentiment-analysis-using-svm-338d418e3ff1

Twitter sentiment analysis | Kaggle. (2017). Retrieved 11 December 2019, from https://www.kaggle.com/c/twitter-sentiment-analysis2