Twitter sentiment analysis

A. Soutif, T. Lambert, W. Didier, A. Kulakova, J. Lindsay

Sunday February 3, 2019

Under the supervision of:
Prof. Michael Blum

Abstract

Brexit is the impending withdrawal of the United Kingdom from the European Union, which followed the referendum of 23 June 2016 when 51.9 per cent of those who voted supported withdrawal.

Being a controversial issue, it caught much attention of the authorities willing to estimate the number of each campaign's supporters. Consequently, a large number of opinion polls were conducted by research and political institutions and also independent media. In addition to that, it caused a lot of discussion in social networks, especially, Twitter — a micro-blogging platform which proved itself as a powerful tool for opinion-mining.

The aim of our project was to use sentiment analysis of Twitter posts in order to predict the popularity of politicians or political stances. We chose to gauge the popularity of each sides of the 2016 referendum about Brexit. This choice is motivated by high level of social impact of this subject and the huge amount of tweets it generated.

The initial scope of our research included creating a time series representing the popularity of the concept of Brexit among the British population along the period covered by validation data [5]. However, due to some limitations caused by the dataset used our analysis is restricted to a two months period and the prediction is available for one week ahead. Although the research in this direction has been already been done previously (see [6]), we managed to reveal new facts on how each side functioned and manifested on social media.

The report is organised as follows. In chapter 1 we describe the data collecting part of the work and provide the statistics on Twitter publications that we considered. Chapter 2 is dedicated to overview of models which can be used for sentiment analysis, including the possible limitations they encounter and the reasons for multi-layer perceptron's use in our work. In chapter 3 we demonstrate the results obtained by aforementioned algorithms run purely on Twitter data. Finally, in chapter 4 we summarize our approach and discuss the possible extensions of current work.

Contents

— 1 —
Data access and retrieval

1.1 EU Referendum Poll of Polls

The source which we used, What UK Thinks: EU Poll of Polls, is a cumulative result of relatively large polls of voting intentions in the EU Referendum. Figure 1.1 shows how the Poll of Polls changed from 22 of September 2015 when the first poll was conducted up to the day before the voting itself. It is worth mentioning that the series includes both polls that were only conducted in Great Britain and those that were conducted across the United Kingdom as a whole.

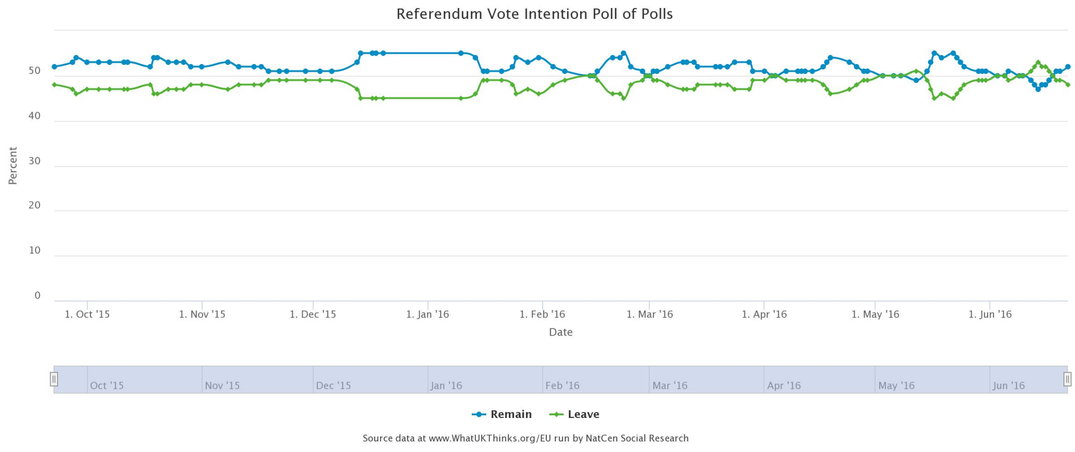


Figure 1.1 – *What UK Thinks: EU Poll of Polls results*

From now on we will focus on the percentage of Brexit supporters which can be also referred as "Leave". We notice that the curve for "Leave" is nearly flat, having the mean of approximately 48.2% with variance of just 2.9%. This data can also be visualized using a method called time-series decomposition that allows to decompose the original time series into three distinct components: trend, seasonality, and noise.

We conclude that no seasonal component is observed, which causes difficulties in extracting features of it and building up the forecasts. Although we can train one of the commonly used time-series model, known as ARIMA (Autoregressive Integrated Moving Average), it shows poor performance.

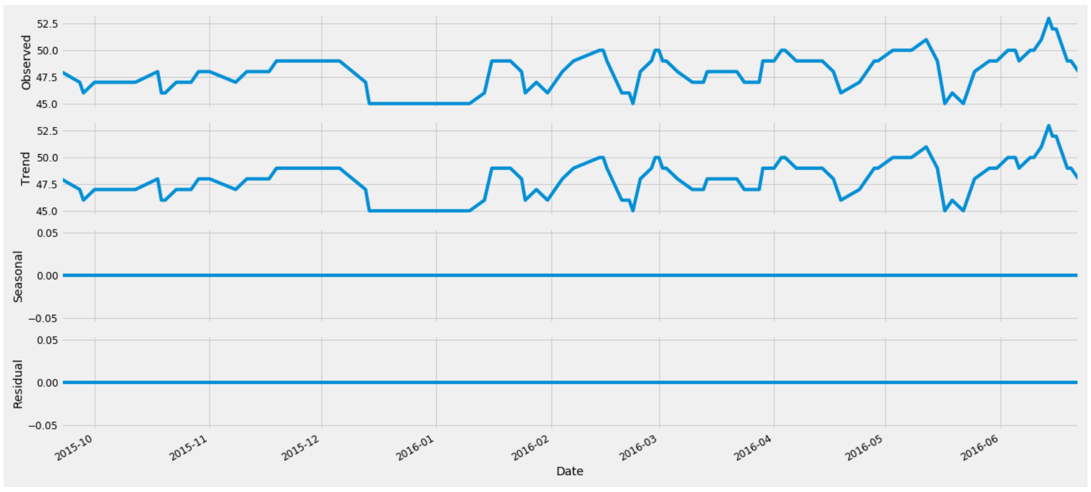


Figure 1.2 – *Time-series decomposition of the original time series*

Figure 1.3 shows that standardized residual series seems to be stationary with mean centered around 0. KDE follows closely with the normal distribution $N(0,1)$. Q-Q plot is linear, as it should be for normally distributed residuals. The correlogram indicates no correlation between time series residuals and previous data points since all points are within the light blue confidence interval.

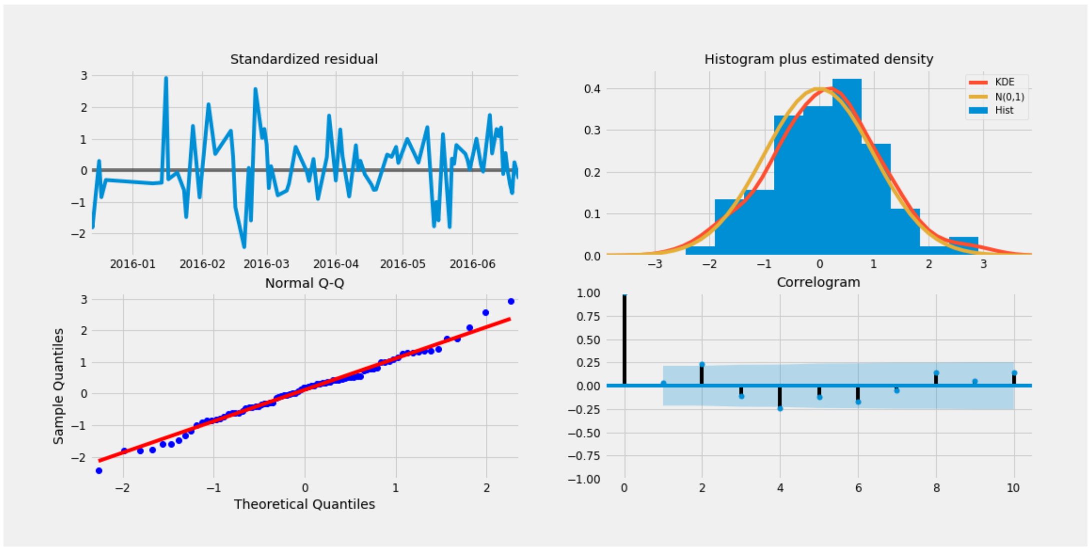


Figure 1.3 – *Arima model diagnostics*

Indeed, it can be seen that despite fitting the original data during the validation, when the whole time-series is available (Figure 1.4), it fails to provide a sufficient forecast when information from the time series is available up to a certain point, and after that, forecasts are generated using values from previous forecasted time points (Figure 1.5).

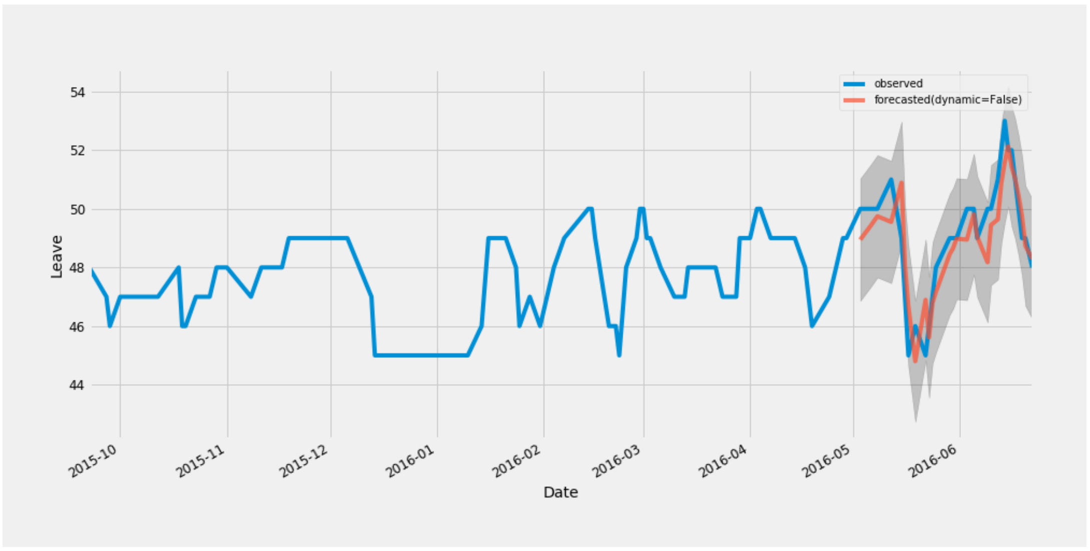


Figure 1.4 – *Validation of the model: red curve is close to the original time-series.*

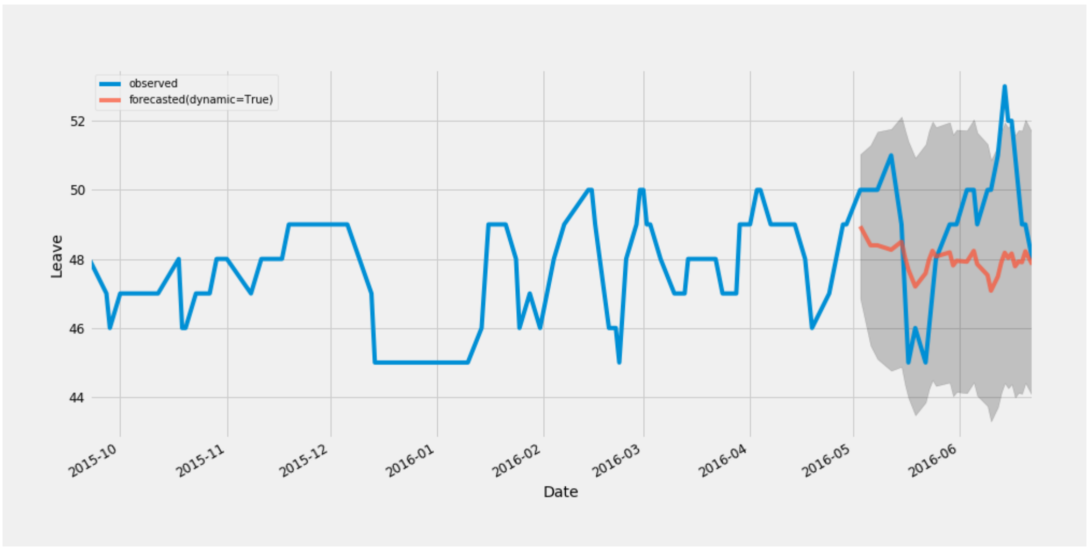


Figure 1.5 – *Forecast: predicted values significantly differ from the original data.*

The predicted values for the last month of the campaign nearly compose a plateau around the mean and the confidence interval is approaching $[-\sigma^2, +\sigma^2]$, which makes the forecast uninformative and unreliable. The most important issue is that the model relies on seasonality and trends, but does not account for the political background and news and fails to capture the changing public intentions.

The aforementioned drawbacks of the traditional time-series models provoke to consider completely different approach to prediction the public attitude towards Brexit.

1.2 Twitter API

Nowadays the potential of online social media for analyzing the public sentiment has been shown in several work (for example, see [2]). With its increasing importance in political discussions, we were suggested to use Twitter to collect data on public attitude towards Brexit.

The most common way to gather Twitter data is to use the API from the website, however, huge drawbacks are involved. Indeed, it is impossible to get tweets that are more than one week old using the API filtering. For example, if one wants to download all tweets containing "Brexit" it could be possible only for tweets of the passed seven days. Older tweets will not be returned by the API.

The second limitation is about data volume: the API only allows 180 tweet downloads every 15 minutes. Consequently, it is supposed to be a time-consuming procedure to build a dataset, that would satisfy our needs in this research — currently more than 100 tweets about Brexit appear per minute.

Due to large volume of data and Twitter API limitations, we had severe constraints and could not build a dataset covering tweets from the vote announcement until present day. We decided to use dataset [1] including all #brexit tweets collected from the 5th of May to the 24th August 2016.

The data is stored as a collection of IDs which then can be downloaded and turned into the actual `json` file using both Twarc and Poultry Python libraries.

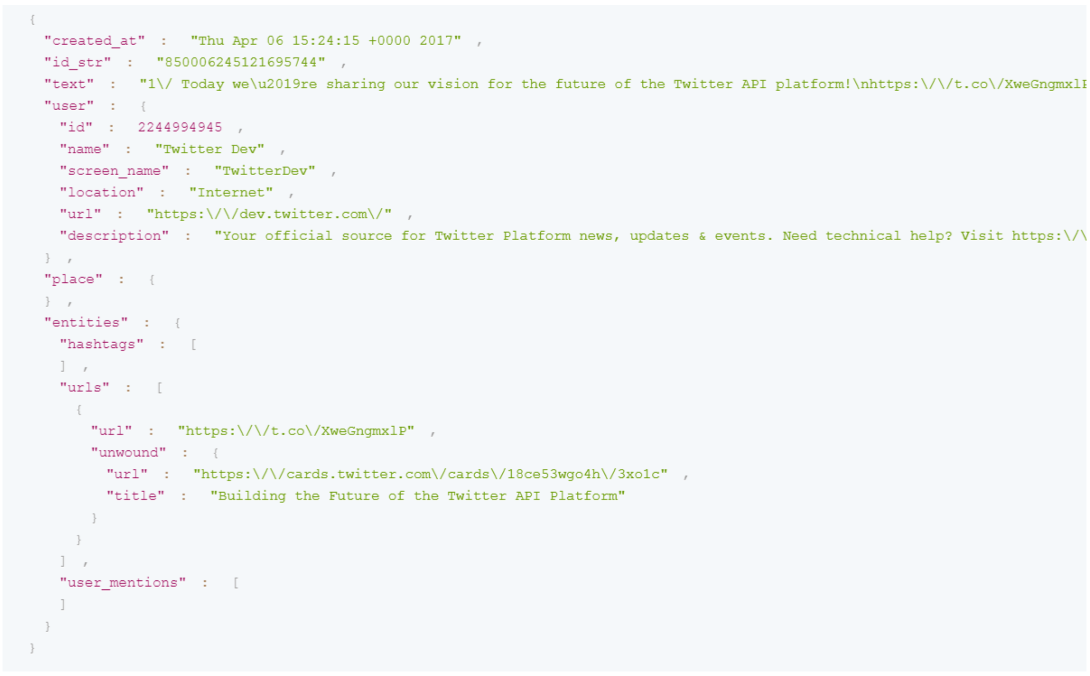
1.3 Data pre-processing

In this section we summarize our approach to Twitter data pre-processing. Indeed, the obtained 56GB jsonlines file, containing plenty of attributes ranging from the profile description of the author of the tweet to the retweets counter of the particular tweet (see Figure 1.6 for details), is not convenient for our research purposes.

We will further consider the text of the tweet, its hashtags, the user ID and the date when the actual publication was posted. For this purpose a script was created which enabled us to reduce the size of the dataset to 2.5GB. Each tweet is represented by the following dictionary:

```
{
'created_at'
'full_text'
'user_id'
'hashtags': list of {'text'}
'symbols'
}
```

We did not keep the favorites and retweet count, even though it was available information, because even though it could be a measure of how influential a tweet is among a community, our goal is to classify every user's opinion about Brexit and not how influential could a tweet supporting a side or the other is.



```
{
  "created_at"  :  "Thu Apr 06 15:24:15 +0000 2017" ,
  "id_str"  :  "850006245121695744" ,
  "text"  :  "1\/ Today we\u2019re sharing our vision for the future of the Twitter API platform!\nhttps:\/\/t.co\/XweGngmxlP
  "user"  :  {
    "id"  :  2244994945 ,
    "name"  :  "Twitter Dev" ,
    "screen_name"  :  "TwitterDev" ,
    "location"  :  "Internet" ,
    "url"  :  "https:\/\/dev.twitter.com\/" ,
    "description"  :  "Your official source for Twitter Platform news, updates & events. Need technical help? Visit https:\/\
  } ,
  "place"  :  {
  } ,
  "entities"  :  {
    "hashtags"  :  [
    ] ,
    "urls"  :  [
      {
        "url"  :  "https:\/\/t.co\/XweGngmxlP" ,
        "unwound"  :  {
          "url"  :  "https:\/\/cards.twitter.com\/cards\/18ce53wgo4h\/3xo1c" ,
          "title"  :  "Building the Future of the Twitter API Platform"
        }
      }
    ] ,
    "user_mentions"  :  [
    ]
  }
}
```

Figure 1.6 – *Introduction to Tweet JSON from Twitter developer portal.*

We also wanted to make sure that tweets were coming from the UK because only British citizens could vote, so we didn't want non voting people to influence our predictions by expressing their feelings about the referendum. One option is to discard publication with non-UK geotag. Unfortunately, less than 1% of tweets had localisation information available among randomly picked 100000. Finally, we only discarded tweets that were marked as non English language by the Twitter algorithm. This would have anyway been an essential step before applying language based models.

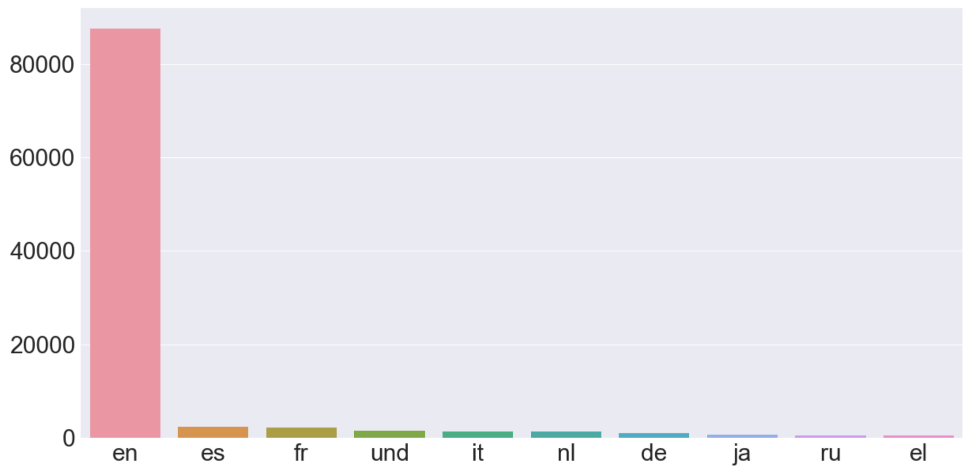


Figure 1.7 – *Languages distribution.*

The aforementioned script treats lower- and upper-case hashtags as equal due to transformation performed in order not to encounter case sensitivity-related problems during the analysis. After this procedure some descriptive statistics can be revealed.

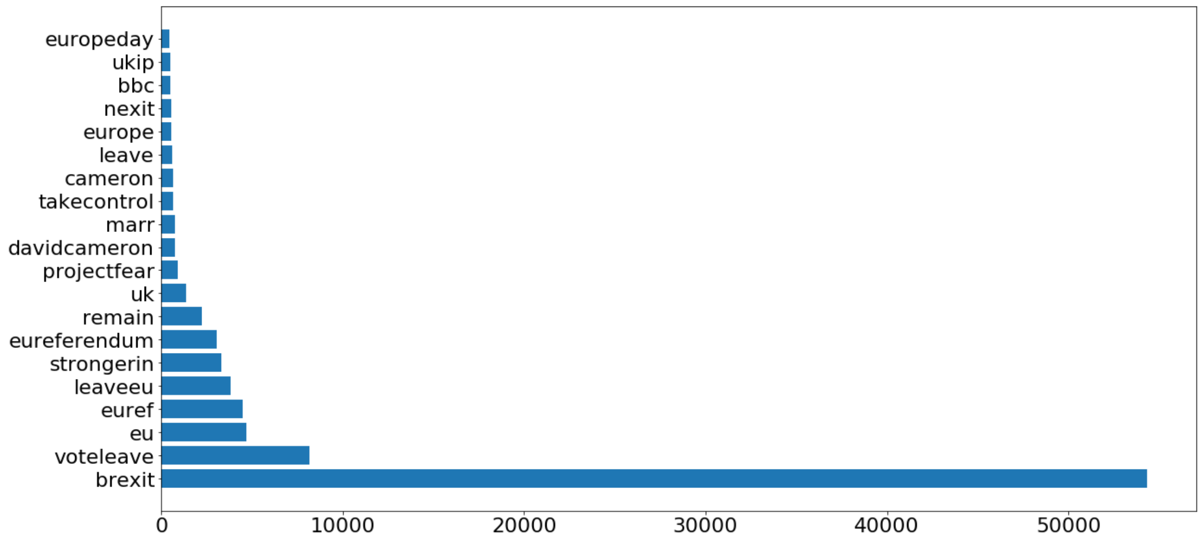


Figure 1.8 – *Hashtags distribution.*

Figure 1.8 shows a predominance of pro-Brexit hashtags, like "#voteleave" and "#leaveeu" over the pro European hashtags like "#strongerin" and "#remain". Unsurprisingly, #Brexit is by far the most used hashtag in the dataset.

1.4 Main challenges

Before moving on to methods, we would like to list the main challenges the models, which will be discussed in details in chapter 2, will have to deal with:

- Bias towards pro-Leave tweets
- Length of the tweets
- Content of the tweets

The bias towards pro-Leave tweets. After performing the procedure briefly described in section A.1 of the appendix chapter, which enabled us to work in the supervised setting, we noticed that 76% of the tweets in our collection were labeled as pro-Leave and the remaining 24% of them as pro-Remain. We consider it an important issue causing the bias, which should be taken into account while consructing the metrics and discussing the results.

The length of the tweets. In addition to that, the length of the tweet which is less than 140 characters severely restricts the volume of data available to perform the sentiment analysis.

The content of the tweets itself is quite uninformative for natural language processing due to its overall neutral purpose and additional information such as links, pictures or videos that we can not analyze.

— 2 —
Methods and algorithms

The considered machine learning problem is a hybrid problem since the results of opinion polls on Brexit over time are available, but not the tweets labelled as for- or against-Brexit. Therefore, the variety of approaches to the problem can be taken. In the following part we will review three approaches that could be relevant to classify a user's sentiment concerning Brexit.

2.1 Louvain algorithm

The fundamental idea behind unsupervised learning approach is to detect some regularities in the data to enrich the labeled dataset we had. If we find some clusters based on hashtags or users that can be classified using 'expert' knowledge (in the sense of a prior knowledge of the field), we could add the related tweets to the pro-Leave and pro-Brexit categories. To give an example, if we find a coherent cluster centered around 'Leave', we can classify other tweets from the cluster to extend the size of the training set.

Being rich and diverse, the Twitter data enabled us to perform an exploration of the dataset and draw conclusions that could help interpreting the results given by other techniques.

Our approach is based on similarity between objects, such as hashtags or mentions[1]. Given a set χ, a similarity measure on its elements is a function $f : \chi^2 \to \mathbb{R}^+$ that computes how similar elements are. The higher it is, the closer the elements are. On the other side, a similarity measure of 0 means that the elements have nothing in common. Getting back to the example of hashtags co-occurrences, the similarity measure would associate to two hashtags the number of times they appear simultaneously in a tweet.

The general workflow is the following:

1. Filter the data to select only information we want for clustering. For example list of mentions or hashtags.
2. Given a similarity measure f, usually cosine or χ_2, and a set of observations $(x_i)_{1 \leqslant i \leqslant n}$, we build the matrix A such that $[A]_{i,j} = f(x_i, x_j)$.
3. We construct the weighted graph with vertices $(x_i)_{1 \leqslant i \leqslant n}$ and edges $e_{i,j} = f(x_i, x_j)$.

[1]Mentions occur when a Twitter user mention another one using the '@' symbol

4. We apply a community detection algorithm to the graph such as the Louvain algorithm.

The cosine similarity between 2 numerical vectors is defined such that it ranges in $[-1,1]$. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors oriented at 90° relative to each other have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. We have the following formula for the cosine similarity :

$$\cos(A,B) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Cosine similarity then gives a useful measure of how similar two documents are likely to be in terms of their subject matter and can therefore be interesting to measure similarity among full tweet text.

The modularity measures the ration of the density of links inside communities compared to links between communities of a graph is defined by

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i,c_j)$$

where :

- A_{ij} represents the edge weight between nodes i and j;
- k_i and k_j are the sum of the weights of the edges attached to nodes i and j, respectively;
- $2m$ is the sum of all of the edge weights in the graph;
- c_i and c_j are the communities of the nodes;
- δ is a simple Kronecker delta | delta function.

The Louvain algorithm was used to detect communities among the graph that we built using the similarity measures. It is a greedy algorithm that iteratively maximises the modularity of the graph. It is a useful algorithm to cluster social media data [4].

Cortext web platform

While looking for literature on the subject, we found a web platform called *Cortext*, that fitted our needs. It is a scientific tool developed by the INRA and the LISIS lab, that offers a wide set of methods to perform textual analysis.

One can import its dataset on Cortext servers and build some pipeline to go from raw data to relevant insights using pre-implemented algorithms. We mainly used the *network mapping* section to do what we explained above. Figures of outputs produced by the platform are presented in the devoted section.

This platform had many advantages over a homemade implementation of the Louvain algorithm. Indeed, it was brought to a knowledge through an explanatory article [3] addressing the exact same issue in the context of the Trump 2016 election. The platform is designed to handle raw `jsonl` files, and then allows the user to produce network outputs while only dealing with important hyperparameters such as the similarity metrics and number of terms to extract from the database. Also, it provides visual and interpretable outputs in a clear fashion. This platform allowed us to be extremely efficient on this part of the project.

2.2 Naive Bayes classifier

Applying a supervised learning approach implies having a set of labelled set of tweets that can be used as a training set. The problem of obtaining the mapped data is referred to section A.1 of the Appendix. In the following sections, we will specify two baseline binary classification methods and suggest some improvements that can be done based on the problem considered.

Naive Bayes classifiers are a family of simple "probabilistic classifiers" based on Bayes' theorem with assumptions about independence of features.

Let $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$ be a vector of observation which is described by n features. Then, being a conditional probability model, naive Bayes assigns probabilities $p(C_k | x_1, x_2, \dots, x_n)$ for each of K possible classes C_k to the observation.

Using the above mentioned Bayes theorem, the conditional probability is written down as follows:

$$p(C_k|\boldsymbol{x}) = \frac{p(C_k)p(\boldsymbol{x}|C_k)}{p(\boldsymbol{x})}.$$

Taking into account the independence assumptions, we have

$$p(C_k|x_1, x_2, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k),$$

where $Z = p(\boldsymbol{x}) = \sum_k p(C_k)p(\boldsymbol{x}|C_k)$ is a factor which depends only on $\boldsymbol{x}$ that is a constant if the values of the feature variables are known.

Finally, using the maximum a posteriori (decision rule) the most probable class is given by

$$\hat{y} = \text{argmax}_{k \in K} p(C_k) \prod_{i=1}^{n} p(x_i|C_k).$$

In sentiment analysis setting, the observations are tweets and the problem statement implies two classes: pro-Leave (1) and pro-Remain (0). The question about the feature extraction and representation of text as a vector will be discussed in the Appendix section of the report.

2.3 Multi-Layer Perceptron

The Multi-Layer Perceptron is a mathematical model consisting of an acyclic graph of formal neurons fully connected with each other. The input signal is propagated through this graph from the input layer to the output layer. In the case of binary classification, the output layer can consist of a single neuron, taking values from range 0 to 1 (sigmoid activation). The classification is then made by rounding up the number that this layer outputs. If it rounds up to 1, the tweet is pro-Leave, but if it rounds up to zero, the tweet is pro-Remain.

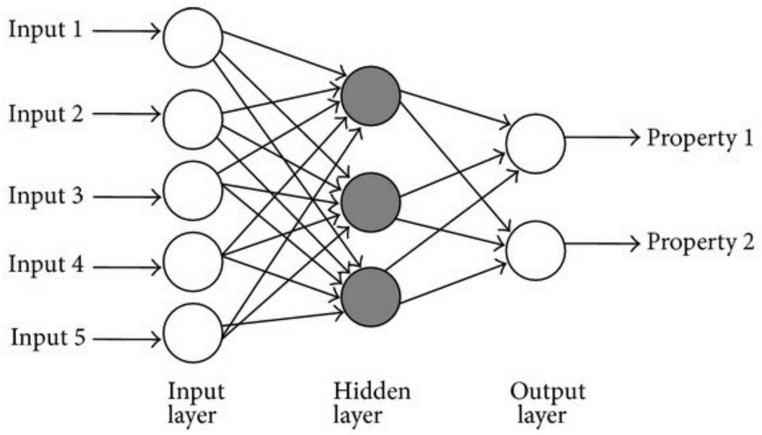


Figure 2.1 – Scheme of a multilayer perceptron

The computations made for one neuron j on a given layer l are the following:

$$a_i^{(l)} = w_{.,i}^{(l)T} z^{(l-1)} + b_i^{(l)} \tag{2.1}$$

$$z_i^{(l)} = \sigma(a_i^{(l)}) \tag{2.2}$$

σ can be any activation function that you can derive. In our case, we chose the ReLU function for the hidden layer and the sigmoid function for the output layer.

The ReLU activation:

$$\sigma(x) = max(0, x)$$

The sigmoid activation:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

The training of this network is made with Stochastic Gradient Descent. When an input from the training set is propagated, it results in an output value which can be compared to the label we actually want for this input. This comparison is made via a loss function. Below the update rule is described. We denote $w_{ij}^{(l)}$ the weight from unit i of layer $l-1$ to unit j from layer l, η the learning rate.

$$w_{ij}^{(l)} = w_{ij}^{(l)} - \eta \frac{\partial l(h(x), y)}{\partial w_{ij}^{(l)}}$$

Using the chain rule:

$$\frac{\partial l(h(x),y)}{\partial w_{ij}^{(l)}} = \frac{\partial l(h(x),y)}{\partial a_j^{(l)}} \frac{\partial a_j^{(l)}}{\partial w_{ij}^{(l)}}$$

We have $\frac{\partial a_j^{(l)}}{\partial w_{ij}^{(l)}} = z_i^{(l-1)}$.

If l is the last layer:

$$\frac{\partial l(h(x),y)}{\partial a_j^{(l)}} = \sigma(a_j^{(l)})' \times (z_j^{(l)} - y_j)$$

otherwise:

$$\begin{aligned}\frac{\partial l(h(x),y)}{\partial a_j^{(l)}} &= \sum_{k \in Af(l)} \frac{\partial l(h(x),y)}{\partial a_k^{(l+1)}} \frac{a_k^{(l+1)}}{a_j^{(l)}} \\ &= \sigma(a_j^{(l)})' \sum_{k \in Af(l)} \frac{\partial l(h(x),y)}{\partial a_k^{(l+1)}} \times w_{jk}^{(l+1)}\end{aligned}$$

Technical details

Memory: Training the network requires some time and computational resources. One issue was that the whole training set turned into vectors of size 40000 (size of the vocabulary) could not fit into the computer memory. Some models could handle a sparse format for these input that was really convenient to use in our case (each tweets has typically less than 20 entries set to 1, others are all equal to 0) but `Keras` required a dense input of size 40000 (note: `Keras` is the Python framework we used for implementing neural networks). So we used a training method called `fit_generator` that allowed us to expand the batches one by one rather than transforming the whole data into a dense representation at once.

Dropout: The number of parameters in the network was huge because of the size of the input layer. Typically it was of the order of 4 million parameters. Because of this number of parameters, we had to be really careful about over-fitting. A practical way to deal with such a problem in neural networks is to add a Dropout layer. It consists in deactivating randomly a fixed number of neurons at each step.

Unbalanced Classes: As mentioned earlier, the proportion of training samples on pro-Leave tweets was higher than the one for pro-Remain. Training an MLP with such unbalanced classes could create a bias towards predicting the overpopulated class. Several options are available to try to eliminate such a bias. The one we choose is to put weights on the training samples when computing the loss. These weights are computed as a function of the proportion of each class in the training set. Here, we gave more weight to the pro-Remain samples and less weight to the pro-Leave samples.

The model in `Keras` is defined as follows:

```
# Keras model
model = Sequential()
model.add(Dropout(0.3, input_shape=(input_shape,)))
model.add(Dense(100, activation="relu"))
model.add(Dropout(0.3))
model.add(Dense(1, activation="sigmoid"))
model.compile(optimizer="adam", \
loss='binary_crossentropy',\
metrics=['accuracy'])
```

Discussion and improvements

One big weakness of the resulting model comes from the initial encoding of the tweets. Indeed, the one-hot encoding procedure is trained only on the training set and thus does not contain every possible word and declination of word one can imagine. This weakens the generalization power of the model. For instance, if the model has learned that "EU is fragile" indicates a pro-Leave tweet it will not be triggered by another tweet saying "EU is weak", even if those two words are 'close' in meaning in this context.

A first work-around to limit this weakness is to try to reduce the amount of possible words a tweet can have by turning every word into its principal part (*lemmatization*). Another more powerful technique could be to use Word Embedding, which reduces the dimension of the input a lot more than lemmatization and captures the semantic meaning of a word in some way. Such a model can be found pre-trained on the internet or trained by hand on a huge dataset in an unsupervised manner. The first method is the simplest one to implement. One can lemmatize a word by using existing python libraries like spacy. However, this is only working for correct English words. This approach was not implemented in current research due to the features of the original data, which mainly contained hashtags and citations and lacked the determinant words (like Brexit).

Another weakness is that the model does not take into account sequential dependencies between words inside the tweets, except from one step ahead dependencies with the use of bigrams. A way to capture those dependencies is to use recurrent neural networks.

— 3 —
Results

3.1 Clustering analysis

This section is devoted to the analysis of the clustering we applied using the Cortext platform. It provides interesting insights about the dataset even though the results aren't really useful to enrich the training set as we were expecting. Our goal was to identify clusters of information (hashtags, mentions, terms) used by a side or the other to enrich the training set in prevision of the supervised part of the project. The first clustering was applied to the hashtags of the dataset. We decided to exclude the #Brexit from the analysis because of its weight. Indeed, #Brexit being the central hashtag for discussions regarding this topic, the node corresponding to this hashtag was extremely central and was preventing a proper clustering of the other nodes. We applied the Louvain algorithm on the list of 300 most used hashtags except #Brexit, their similarity being estimated by χ_2 similarity applied to co-occurences. The results are provided in figure 3.1. Closer screenshots are provided in the appendix section. The results show that even when removing #Brexit from the analysis there is an important correlation between the most used polar hashtags : #Strongerin and #VoteLeave.

Analyzing the content of the clusters, we can see that the red cluster basically groups all the most active and polar hashtags together, without distinguishing pro-leave and pro-Remain hashtags.

Running the same kind of analysis on the mentions co-occurences did not give much better results as it can be observed in Figure 3.2. The official accounts @vote_leave and @strongerin are highly correlated which indicates that they have been used concurrently in any tweets.

Both results do not allow us to identify communities or hashtags used by a side or the other because the polar information is clustered all together. Though, a finer analysis of the results showed us that an active far-right populist section used correlated hashtags such as #Trump, #maga (Make America Great Again) or #pegida. See figure A.3 for a zoom on this cluster.

Also, in figure 3.2 we can see that @Nigel_farage is very often mentioned when official media accounts are. This tends to confirm the bias towards pro-Brexit activity on Twitter, as we can suppose, using our knowledge, that tweets supporting Nigel Farage also support Brexit.

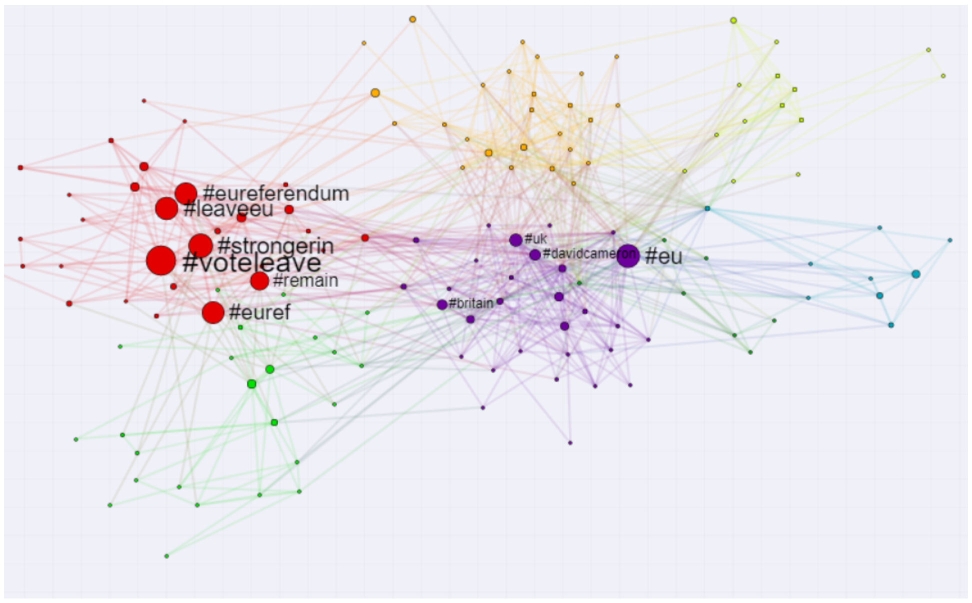


Figure 3.1 – *Overview of clustering using hashtag co-occurences and χ_2 similarity, excluding #Brexit*

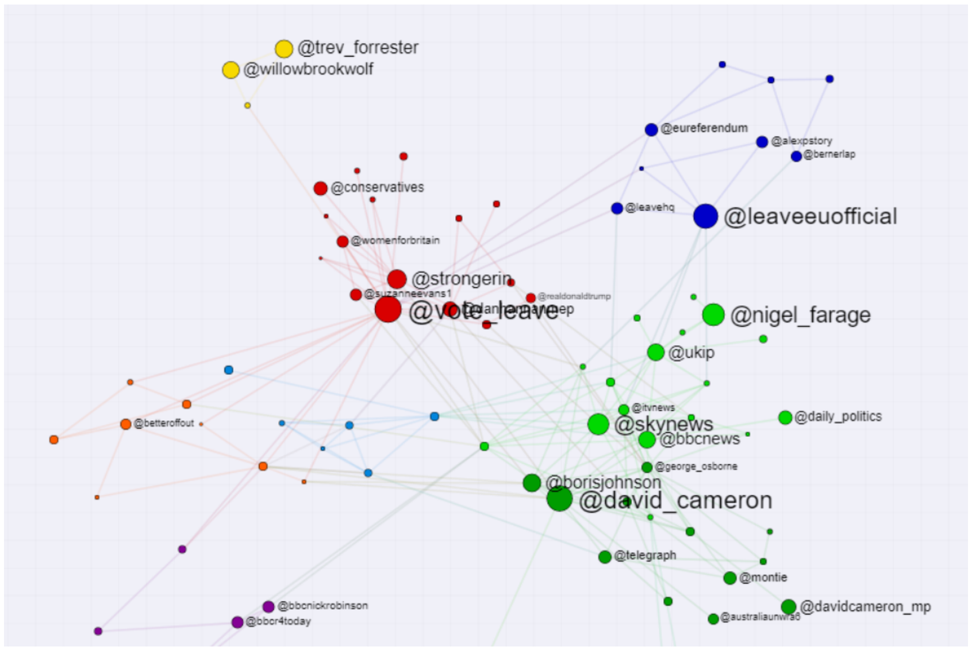


Figure 3.2 – *Overview of clustering using mention co-occurences and χ_2 similarity*

3.2 Classification assessment

In this section, we present two metrics that are really useful to the comprehension of the results. These metrics are defined for each class. Intuitively, the precision corresponds to the percentage that a prediction is true for the given class, and the recall corresponds to the percentage of samples the classifier spotted amongst all samples from that class. Denoting the true positive as tp, the false positive as fp, and the false negative as fn, we have the following definitions:

$$Precision = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

The results below shows us that even a simple model like Naive Bayes is able to perform well on sparse data like text data. Its recall metrics are quite good and it even catches more pro-Remain tweets than the multi-layer perceptron. However the results are not precise enough for pro-Remain prediction. So this model has a tendency to predict more often than it needs the pro-Remain. On the contrary, MLP with unigrams has a tendency to always predict pro-Leave and catches a really low percentage of pro-Remain, but its predictions have a good precision. The MLP with bigrams is a good compromise, able to catch a decent number of pro-Remain tweets by staying precise on it's predictions.

The training curve depicted in figure 3.1 shows us that the multi-layer perceptron has an over-fitting issue even after having applied a decent amount of dropout. This is because they are too many parameters compared to the number of training samples and because the training data is imperfect, they are some neutral tweets that are labeled as pro-Leave of pro-Remain.

	Naive Bayes (bigram)	MLP (unigram)	MLP (bigram)
Test accuracy	0.755	0.814	0.817
Recall leave	0.748	0.963	0.909
Precision leave	0.850	0.813	0.823
Recall stay	0.767	0.434	0.653
Precision stay	0.633	0.819	0.802

Table 3.1 – Table showing the metrics obtained for several models

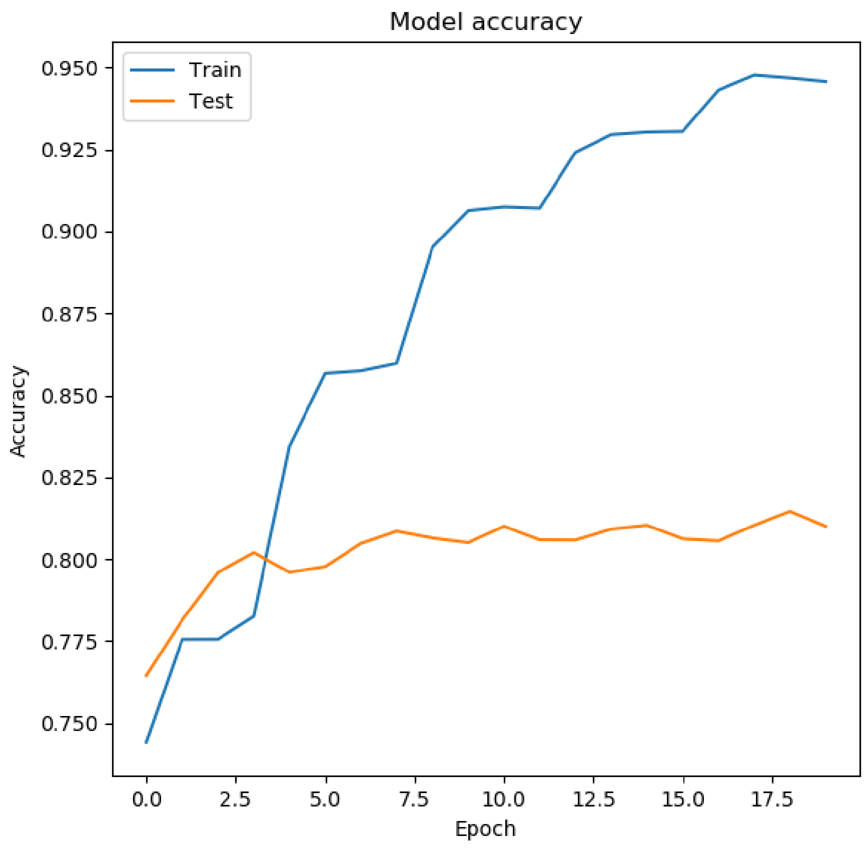


Figure 3.3 – Training curve for the multi-layer perceptron

Here we demonstrate some predictions that the model has made and that we judged by hand.
True Positive Leave

```
- "Why Parents should Vote to Leave the EU to Keep Our Children
  Safe #Parents4Britain #Brexit https://t.co/riq5IONsaU"
- "From where I'm standing the incompetent #EU has already broken
  #Europe @TheEIU - #Brexit can help help fix it.  https://t.co/W1MgGrnhkD"
```

False Positive Leave

```
- "Here is Why UK #Startups Should Fear Leaving The EU
  https://t.co/ZWfUVaDPa7 via @Mattermark #Brexit"
```

True Positive Remain

```
- "Brexit to make U.K. less attractive to investors:
  Japan https://t.co/OKFfGDUBX5 @japanherald"
- "RT @CER_Grant: If voters knew about the good work
```

```
  of Europol,they wouldn’t want to risk Brexit.
  @CER_London @CaminoMortera  https://t.co/â¦"
```

False Positive Remain

```
- "RT @bengoldacre: People say the Brexit camp is all cranks and racist
  pub windbags but here is some evidence to the contrary\n\nhttps://t.co/Lâ¦"
```

Interpretation

Such a model is quite hard to interpret since the number of units is chosen arbitrarily and different architectures with a different number of layers and a different number of units work well too. However, with only one hidden layer we can visualize which words are given the most weight between input and hidden layer and then which unit is given the most weight between hidden and output layer. We have the following results for the top 5 words of the top 5 units (units with most weight either negative or positive). Negative weights means the units pulls the prediction towards pro-Remain whereas Positive weights pull it towards pro-Leave.

```
unit 0 ; weight -2.1019089221954346:
veto brexit
voting because
rt britinfluence
gbpusd
vs brexit

unit 1 ; weight -1.803170084953308:
votein votestay
vs brexit
wolfgangtillmans
voted but
rt britinfluence

unit 2 ; weight 1.466126799583435:
rt bernerlap
subtel1
rt petenorth303
thomasbrake
strongerin no2eu

unit 3 ; weight 1.2806601524353027:
dexit
thomasbrake
retweet retweet
no2eu euref
strongerin no2eu
```

```
unit 4 ; weight 1.6732659339904785:
vote_leave ukip
no2eu euref
thomasbrake
rt bernerlap
strongerin no2eu

unit 5 ; weight 1.3731175661087036:
retweet retweet
rt sj_powell
dexit
strongerin no2eu
thomasbrake

unit 6 ; weight 1.4757411479949951:
rt bernerlap
thomasbrake
dexit
strongerin no2eu
rt petenorth303

unit 7 ; weight -2.0975162982940674:
in boost
brexit nonsense
gbpusd
voted but
vs brexit

unit 8 ; weight -1.9592608213424683:
voted but
wolfgangtillmans
grumpy_p_sloth
vs brexit
gbpusd

unit 9 ; weight -1.969815969467163:
euref in
voting because
rt britinfluence
wolfgangtillmans
gbpusd
```

As we can see, a lot of words with most weights are in the end hashtags and citations or retweets. This is quite a bad news for prediction because citations for example can appear during a certain period of time as a reaction to a popular tweet from the user. But after this period citations of this user may not mean the same thing. It is also interesting to notice that even if the user cited is pro-Remain, the network interprets the presence of his name as pro-Leave because sometimes most of the tweets referring to this user are a counter argument to what he stated before.

3.3 Time series prediction

Discussion Because the models found do not achieve really high accuracy, predicting a time serie based on this model can be really uncertain. However, because the initial goal was to estimate how much twitter users were pro-Leave/pro-Remain, if we gather all tweets from a given user, we can try to tackle this task. Note that in addition to better estimate the real proportion of brexit supporters, it also gives our model more tweets per user to decide whether a user is pro-Leave or pro-Remain. We did this by simply averaging the rounded prediction number for each tweet (i.e a user having tweeted three tweets predicted leave/remain/remain will be labeled as pro-Remain). So, if we are lucky enough and that a lot of users do write more than one tweet, we can hope that our model will be better at predicting the polarity of a user than the polarity of a single tweet.

Methodology After having trained a model, we tried to predict on new unlabeled tweets what the sentiment for each tweet is. Then we created a time series by averaging the user-sentiment and the tweet-sentiment around a given date for a list of dates covering the whole dataset we gathered. We chose to tune this time series by both choosing how much time to skip between two prediction (stride) and how much time around each date to take into account when predicting the sentiment for a date. We have to keep in mind that reducing the width will make the precision for user of lower quality because there will be less tweets to take into account, so also less tweets per user.

Time series obtained

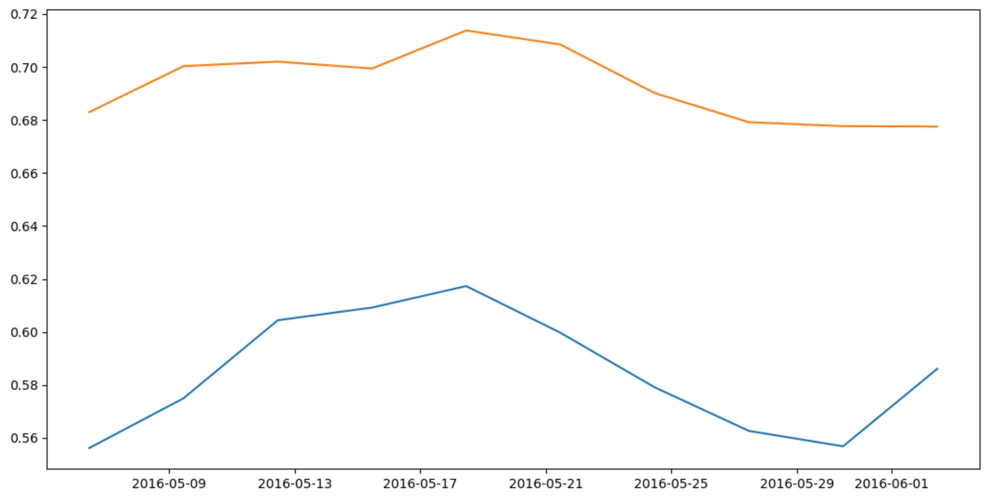


Figure 3.4 – Predicted time series for a period of one month

As we can see, the mean tweet sentiment (orange) is way higher than the mean user sentiment (blue) in general, suggesting that pro-Leave users tweet more than pro-Remain users. The vertical axis should be interpreted as the percentage of users supporting Brexit.

Finally, we compare the average user sentiment obtained by our sentiment analysis model to the original time-series of Brexit opinion polls. Here we note, that the polls are released with a lag and estimating the most probable window between these series can be regarded as a separate complicated task. We assume that it is around 12 days according to the paper [2].

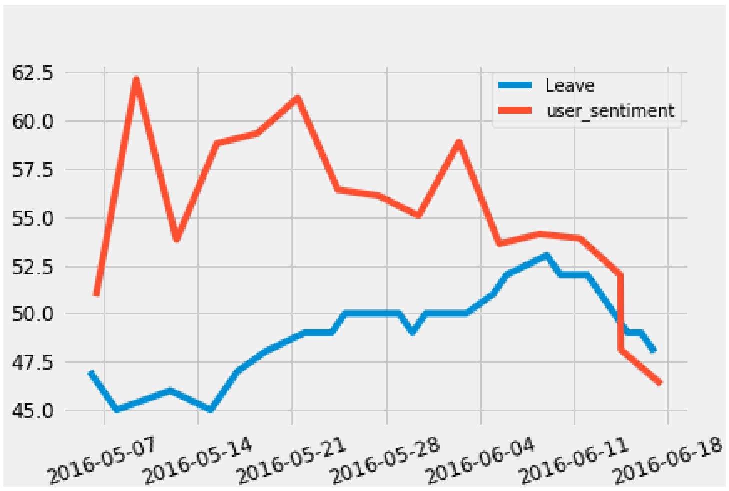


Figure 3.5 – The comparison of Brexit opinion polls time series and the user_sentiment obtained by algorithm scaled by 100.

Due to the limited period of both series' availability and their changing frequencies we can not conclude on the overall correlation between them. However, we note that the user sentiment time series was able to capture trends when the number of tweets generated around these days was high enough (see fig. 3.6).

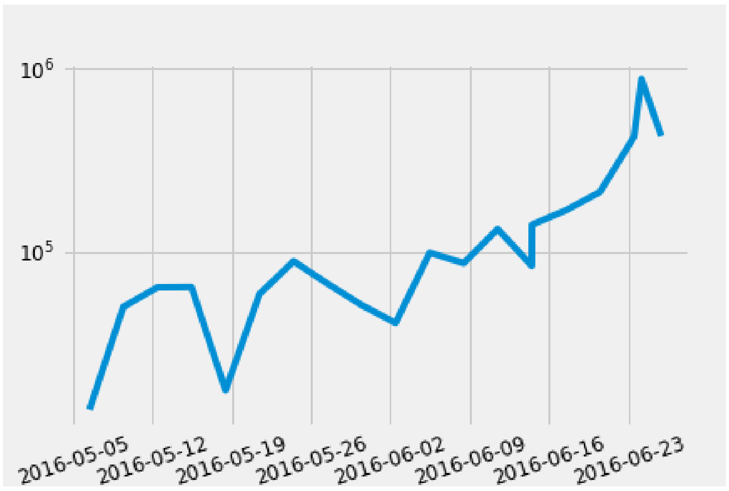


Figure 3.6 – Number of tweets with Brexit tag per day is significantly higher in the last month before voting.

In addition to that, the decline in Brexit supporters around the voting day was captured right. Another question is that according to the Referendum results, the percentage of Brexit-supporters was still higher than those who voted for "Remain". This brings us to the discussion of drawbacks of both opinion polls and social media analysis in estimation the voting intentions.

— 4 —
Conclusion

First of all, it is worth mentioning that the proposed problem is difficult to tackle, which was highlighted in various publications on the topic (see, for example, [2]).

Indeed, although Twitter sentiment analysis managed to reveal some new facts on Brexit campaign in social media and even to predict the dramatic decline in the number of Brexit supporters in the last considered week, it is still unreliable for a precise forecasting and based on doubtful assumptions. For example, we assume that the sentiment of tweet is directly associated with the voting intention of a particular person which is a hypothesis which should be tested carefully, i.e. a Twitter account is not necessary associated with one person and also the average sentiment of the tweets does not provide a reliable evidence that the user expresses his/her opinion.

Also the Twitter language itself is difficult to analyze: even the complicated natural language processing models encounter difficulties with irony and sarcasm tweets often contain. The same causes difficulties for manual labelling of the tweets by people who are not English native speakers. In addition to that, still a lot of the tweets posted by users are not expressing their opinion directly, but are rather neutral. For example, they can be news or tweets that point to a link where an opinion is expressed.

Nevertheless, the multi-layer perceptron model managed to capture the possible decline in number of Brexit supporters before it was shown by opinion polls. It might be due to the fact that the reaction of Twitter users to murder of member of Parliament Jo Cox on 16 June was clearly seen as a growth of tweets supporting "StrongerIn" campaign, while the opinion polls and the campaigns itself were suspended.

However, it seems to us that we more succeeded in capturing the public attention than the voting intentions.

One of the possible directions for improvement can be to examine the "influencers" — people who got lot of likes and retweets for their posts with a relatively high number of followers. Also it could be interesting to combine time-series analysis and regression on number of tweets for and against Brexit and construct the state-space model.

Finally, the project has enabled us to study such topics as binary classifications algorithms and statistics. We significantly improved our abilities to test the hypothesis and evaluate model's performance.

— A —
Appendix

A.1 Labelling of tweets

It is worth mentioning that no labelled data was available to us, consequently, we faced a problem of obtaining relatively large training training set of tweets marked either for or against Brexit based on their sentiment. In this subsection we are going to discuss the way of labelling tweets which makes this procedure less time-consuming than manual labelling.

We could start from the assumption that all tweets having typical pro-Leave hashtags ("Vote-Leave", "LeaveEU") are pro-Leave and those with typically pro-Remain hashtags ("Stron-gerIn", "Remain") are be pro-Remain. By doing this, we can perform an initial mapping. However, we still need to test this hypothesis, and the only way we know how to do that is by taking a small sample of tweets and manually labelĐ´ing them.

So, we start by taking the most popular hashtags. Then, for the most popular (see fig. 1.8) ones in descending order, we use a brute force method. First, we take a sample of tweets. We assume that this sample is representative of the set of tweets as large and we want to find a hashtag that is a reliable indicator of either pro-Leave or pro-Remain sentiment. So, we take every single tweet with that hashtag in it and check its contents in order to determine whether it is pro-Leave, pro-Remain, or neutral. If less than 95% of tweets are pro-Leave or pro-Remain, we discard this hashtag as unreliable. We then use the reliable hashtags as indicators for the polling data.

The first hashtag we check is "Brexit". A majority of the tweets with this hashtag support Leave, however, there are still quite a lot of tweets that do not (around 30%). Consewuently, it is not useful for us to rely on it.

The second most common is "VoteLeave". This hashtag appears to be much more reliable: amongst over 200 tweets, only 2 do not support the Leave campaign, which gives it a success rate of 99% according to our sample. We can thus assume that almost every single tweet with the "VoteLeave" hashtag supports the Leave campaign.

The next most common hashtags are "eu" and "eureferendum" which again yield no interesting results. Then, we have "StrongerIn", which is strongly associated with the Remain campain. However, unlike "VoteLeave", "StrongerIn" is not very reliable: out of around 200 tweets, over 60 supported Leave, which gives us 70% accuracy. However, the vast majority of these used both the "VoteLeave" and "StrongerIn" hashtags. If we exclude these tweets from the dataset, we still get, out of 160 tweets, around 40 supported Leave. This only gives us around 75% accuracy.

By doing this, we obtain a mapped dataset. In fact, we could easily notice a trend: while pro-Leave hashtags are highly reliably indicate pro-Leave tweets, pro-Remain hashtags are far less reliable. This highlights a massive pro-Leave bias among tweets.

A.2 Clustering insights

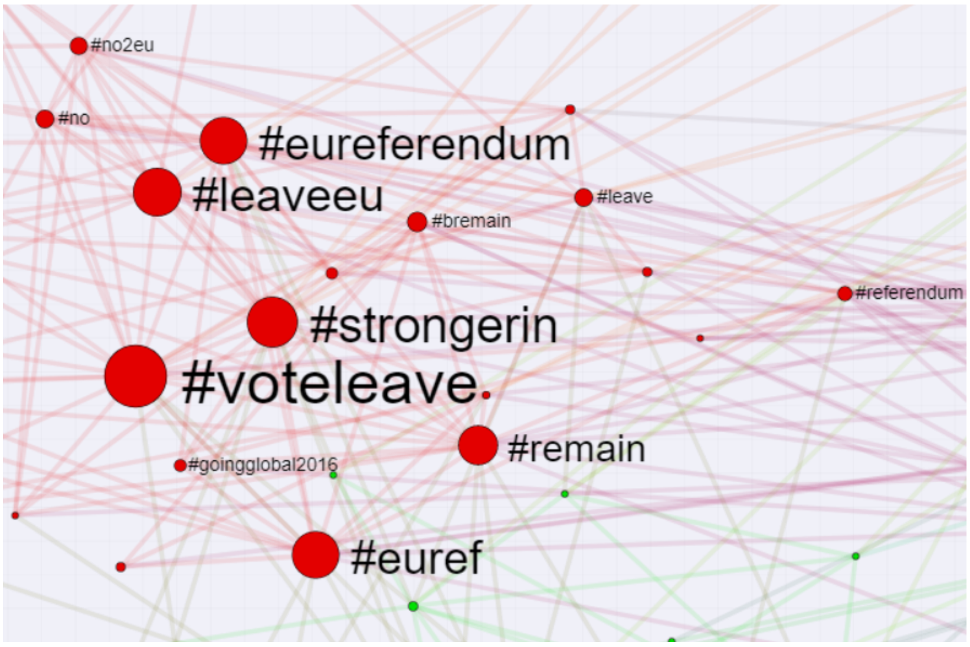


Figure A.1 – *Cluster containing polar hashtags*

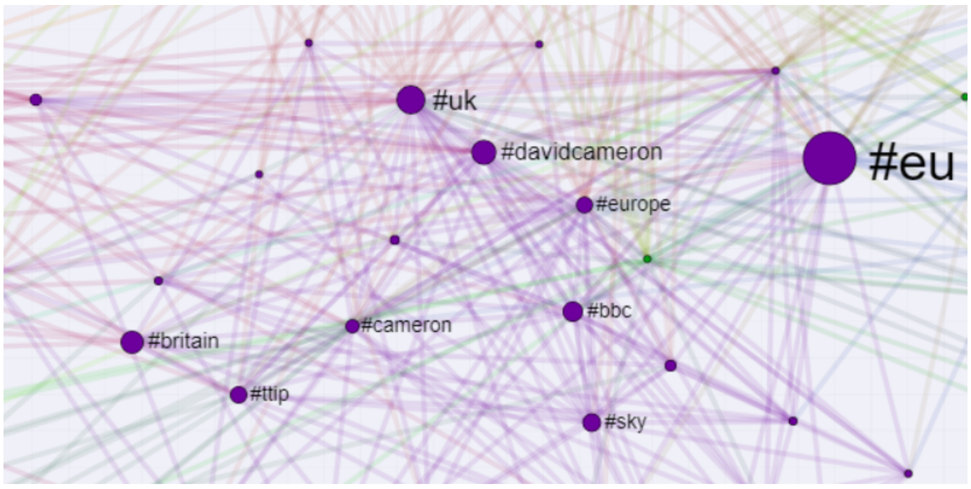


Figure A.2 – *Cluster containing neutral hashtags*

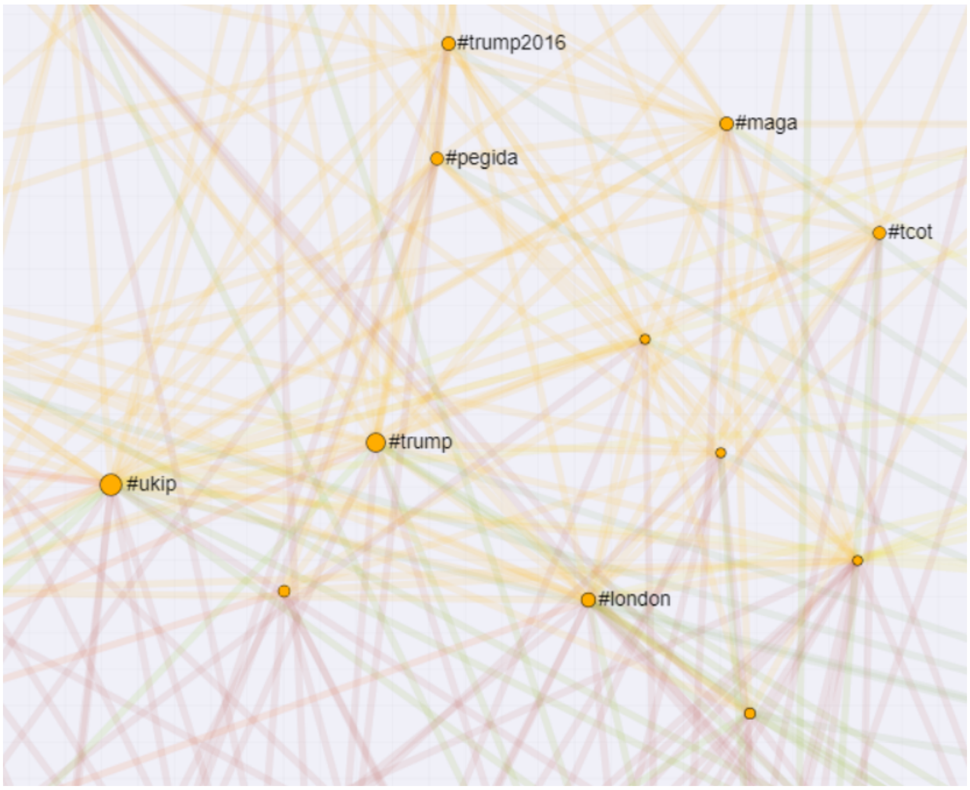


Figure A.3 – *Cluster containing populist hashtags*

Bibliography

[1] #brexit tweets collected from the 5th of may to the 24th august 2016.

[2] Flaviano Morone & Hernan A. Makse Alexandre Bovet. Validation of twitter opinion trends with national polling aggregates: Hillary clinton vs donald trump. *Nature Scientific Reports*, 2018.

[3] Cortext. Hashtag cooccurrence network of tweets about trump. 2017.

[4] Pablo Rodriguez Josep M. Pujol, Vijay Erramilli. Divide and conquer: Partitioning online social networks. 2009.

[5] NatCen Social Research. Eu referendum polls of polls. 2016.

[6] Ross McDermott Manel Zarrouk Manuela Hurlimann Brian Davis Tobias Daudert Malek Ben Khaled David Byrne Sergio Fernandez Andre Freitas Frederico Caroli Siegfried Handschuh Angelo Cavallini Vyacheslav Polonski, Laurentiu Vasiliu. In or out? real-time monitoring of brexit sentiment on twitter. 2016.