Assignment 1: Studying 2016 US Elections Through Analysing Twitter Data

António Mendes
17amendes@gmail.com
11925051

Judit Győrfi
judit.gyorfi@student.uva.nl
13209647

Orlando Scarpa
orlando.scarpa@student.uva.nl
13266918

Adam Horvath-Reparszky
adam.horvath-reparszky@student.uva.nl
13326481

Miklos Kosarszky
miklos.kosarszky@student.uva.nl
13242857

Abstract

As social media becomes a larger part of our day to day lives, it also becomes an important tool for discussing and expressing political opinions. Using various Machine Learning and Data Science methods, we strove to determine whether or not the sentiment expressed on social media can be used as a reasonable estimation of the results of an election.

1 Introduction

In this project, our group analysed a Twitter dataset about the 2016 United States presidential election. To prepare a reasonable conclusion and correlation between the Twitter posts and the final results in each state, we decided to use sentiment analysis. The importance of sentiment analysis is to recognize and monitor the feelings and emotions of society about a topic discussed in social networks. For this objective, we decided to use a set of different sentiment analysis approaches and tried to correlate the results we found with the effective results of the 2016 US elections.

2 Data Cleaning

One of the most important steps in analysing any data and especially text data is data cleaning, so a large amount of our effort was spent in this section. After extracting all the json objects and flattening all the fields the DataFrame had 35 columns, Of these, 10 were selected as valuable insights into the data. Using the columns relating to the language and country code, the DataFrame was reduced to only tweets in English and posted from the United States. After this, the column containing the full name of the location of origin of the tweet was used to extract the two-letter code of the state of origin, keeping in mind to exclude territories whose inhabitants don't vote for the president, like Puerto Rico and Guam. Of the 517,724 tweets in English from the United States, less than 5000 tweets were excluded in this manner. To clean the text and prepare it to be tokenized, all text in the tweets was stripped of punctuation, special characters, new lines, hyperlinks and trailing spaces. Furthermore, stop words were removed and select hashtags and mentions were counted to give a basic estimate of the candidate being talked about in each tweet.

3 Polarity

The already cleaned data provides us with many possibilities in terms of data analysis. One of the main focus of the analysis is on sentiment analysis of the Twitter texts. Python has a wide variety of Natural Language Processing (NLP) tools that are helpful when looking for the sentiment value of a particular text, and in our case, tweets. Some of the packages, such as NLTK and TextBlob can deal with strings mathematically. To find out the polarity of our tweets, we have used the TextBlob tool which classifies the text into negative, neutral and positive tweets between the values -1 and 1, where negative numbers tend to refer to tweets with negative wording and positives refer to tweets with positive words. TextBlob also creates a second output called subjectivity, with values between 0 and 1 where 0 is an objective tweet and 1 is a subjective one. However, in this assignment, we are only focusing on polarity as this gives us a better overview of the tweet.

4 Sentiment Analysis

4.1 Multinomial Naïve Bayes

Our first approach to predict the sentiment of the tweets was to use the Multinomial Naive Bayes. It is generally used where there are discrete features (e.g. - word counts in a text classification problem). The multinomial distribution describes the probability of observing counts among several categories, this was the reason why we chose this first. To feed the machine learning algorithm, we vectorized our string data to numerical values, for this we used CountVectorization and Tf-IDF transformation. Moreover, we used pipeline technique which is a built-in function of scikit-learn to pre-define a workflow of the algorithm. After selecting our model, we had to validate it with the existing training data set, so we used the conventional `train_test_split` technique to split the training data set with the test size of 20% and let our pipeline model be validated on the split data sets. The model reached an accuracy of $75,8\%$, which is relatively low, so we decided to look for another, hopefully more accurate, approach.

4.2 Neural Networks

To attempt to supplement the relatively low accuracy of the Multinomial Naïve Bayes, three different Neural Network approaches were tested. All Neural Networks were developed using Tensorflow 2.0 + Keras, and were trained, validated and tested using the Sentiment140 dataset containing 1.6 million labelled tweets.

The first approach was a simple sequence of three Dense layers, interleaved with dropout layers with a rate of 0.2 to avoid overfitting. The input tweets were first tokenized and converted into a matrix representing the count of how many times a given word appeared in a tweet. The model was trained on 80% of the data, with the remaining 20% being used as the test input. The fit function also used a 25% validation size, to guarantee equal dimensions between the test and validation dataset. The model was fitted for 10 epochs with a batch size of 32, then tested on the test dataset, reaching a final accuracy of 84.7%. While this was considerably higher than the Naïve Bayes approach, we decided to try some other more sophisticated approaches to attempt to further optimize the results.

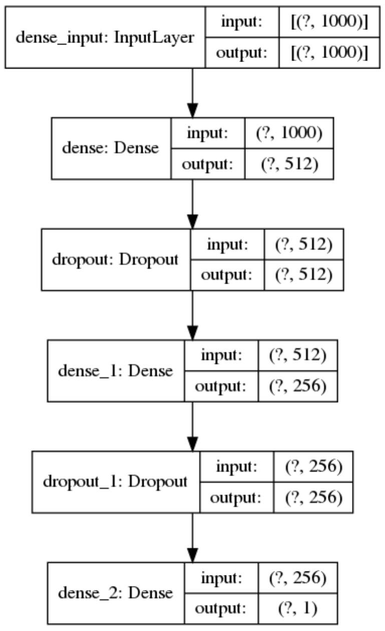

Figure 1: Diagram of Densely Layered Neural Network

The second approach used a 1-Dimensional Convolutional layer in the hope that the net would be able to recognize patterns in sequences of words. For this approach, tokenization was achieved with text sequences to maintain word order. The same $80 - 20$ split was used to separate the input data, with a 0.25 validation split of the train data. Before the 1D Convolutional layer, an embedding layer was added to increase the density of the sparse vector inputs. The Convolutional layer was followed by two Dense layers. Dropout layers were appropriately inserted to avoid overfitting. Once again, the net was trained on the same dataset as the previous approach with 10 epochs and a batch size of 32. Once tested, the model reached an accuracy of 79.4% that, while higher than the Naïve Bayes approach, was still lower than the simple dense model.

The final approach was to use a Recurrent Neural Network to make use of the memory state of these networks and better identify patterns in the tweets. Long Short Term Memory (LSMT) was chosen as it's more sophisticated than basic RNNs and avoids the vanishing gradient problem caused by these. The Train-Validation-Test split was identical to the previous approaches, and tokenization was achieved using the same method as the Convolutional Approach. The LSTM layer was preceded by an embedding and dropout layer, and followed by a Dense layer. The model was once again fitted on a batch size of 32 for 10 epochs, achieving a final accuracy of 82.1%.

All three approaches reached a higher accuracy metric than the Multinomial Naïve Bayes approach, but to our surprise, the simplest model achieved the best results. After using the examined 2016 tweet data as the input for the predict functions of all three models and manually checking samples of tweets where the models disagreed on the sentiment, we found the Convolutional model to be far less accurate than the other two in predicting the sentiment of the tweets, and the Simple model to be very slightly more accurate than the recurrent one. For this reason, we decided to finally use the simple model's predictions of sentiment for the twitter data.

5 Topic Modelling

In addition to sentiment analysis, an attempt was also made to model topics for the tweets. The hope here was to separate the two political candidates into two topics. The main approach explored is a Latent Dirichlect Allocation model (LDA) trained on the same data as was used before. LDA is an unsupervised soft clustering approach, so it allows for some flexibility between the topics. To optimise the model, the α and η hyperparameters are turned according to the gensim python package's coherence measure. The coherence measure ”rate[s] topics according to their understandability”(Röder, Both, & Hinneburg, 2015, p. 399). It is bounded between 0 and 1, with 1 being the most optimal score and 0 being the least optimal score. The highest coherence value is 0.498121 and the corresponding hyperparameters are values are $\alpha = 0.01$ and $\eta =$ ”*symmetric*”. A coherence of ≈ 0.500 is still not ideal, and it possibly suggests that the topics might not be coherent enough. After tuning the lda model, though, closer inspection at the results reveals that indeed the topics are not very coherent. Below is a list of filtered tweets in the first topic:

- Words in Tweet #30524: you picked him now you are staring straight into the face of the base of your party not so pretty is it uglygop nevertrump
- Words in Tweet #33233: realdonaldtrump you obviously have no idea how things work dummy
- Words in Tweet #99920: seanhannity newtgingrich realdonaldtrump he s cooke sean it s over trump is a loser
- Words in Tweet #34744: she is terrified of people trump will rip her a new one at the debates
- Words in Tweet #29191: cnn why do u put on these brain dead individuals the world is watching america usually keeps these morons off tv

While the first three tweets in the list can be interpreted as ”critical of Trump”, the same interpretation cannot be applied to the other two tweets. Too many of the tweets and this topic are similarly arbitrary. A similar problem can be found in the tweets in the second topic. Because of the poor performance of this lda model even after tuning, we fell back on using the hashtags found in the data preparation as an estimate of the topic for the other analyses in this paper.

6 Results

6.1 Polarity Results

The first step in our analysis is to create a new column called “polarity” in the already cleaned data with the TextBlob package based on the tweet texts. The new column will contain the values of the polarities of particular tweets. Next, we are grouping the data by states and taking the average value of the polarities per state. As a result, we get the sorted polarity values for the 50 states. We can observe that the average polarity values of the tweets, in general, are between -0.1 and 0.2. We can observe that tweets in general (not considering that Hillary or Trump is included in the tweet text) are the most positive in Kentucky, while Mississippi has the lowest average polarity score.

To get a better insight on how tweets can be related to votes results, we have split up our dataset to tweets which are mainly about Trump and to tweets about Hillary. After that, we follow the same procedure as before and group the average polarity scores of the two candidates per state. Next, we load an external dataset from the 2016 U.S elections (“usa-2016-presidential-election-by-county.csv”) to compare the results with the polarity of the tweets. The results state that Trump got the most positive comments from states such as Montana, Wisconsin and Kentucky. These are states where Trump also got more votes than Hillary Clinton.

state	polarity	name
NE	0.0819	Nebraska
MT	0.0825	Montana
WI	0.1080	Wisconsin
KT	0.1120	Kentucky

As for Hillary, we can conclude that she received negative tweets in states which are Republican bastions such as Idaho, North Dakota and Wyoming. Correspondingly, she received positive tweets from states such as Washington D.C and New Hampshire which indeed voted for Hillary. We have calculated a correlation score between polarity and the winner of a state (1 : Winner is Trump, 0:Winner is Hillary). The resulting correlation for tweets about Trump and "winner" : Trump is 0.257. The result states that the more positive the tweet about Trump is, the more likely that state has voted for Trump. Vice versa, if the tweets are more positive about Hillary, it is less likely that people in that state voted for Trump. (Correlation: -0.189). However, these correlations are not high enough (smaller than 0.3) to say that the polarity of a tweet has a decisive impact on the vote results.

state	polarity	name
ID	-0.0236	Idaho
ND	-0.0158	North Dakota
WY	-0.0148	Wyoming
UT	-0.0133	Utah
DC	0.0786	District of Columbia
NH	0.0822	New Hampshire

To get a better overview, Figure 2 reflects whether the tweets in a state were positive or negative about Hillary or Trump.

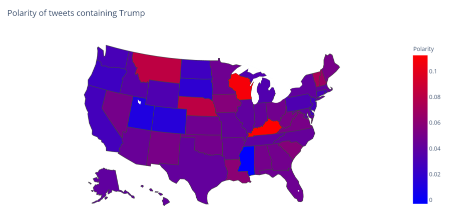

Figure 2: Polarity in Each State

6.2 Sentiment Results

After analysing the Polarity data, a similar examination was performed on the sentiment prediction of the neural network model. When listing the average sentiment for each candidate in each state in descending order, these were the results:

top trump States	top hillary states
Montana	Hawaii
Louisiana	New Hampshire
Wisconsin	District of Columbia
Alaska	Oregon
Arkansas	Kansas

Trump's top 5 states were all states where he did in fact win, albeit in Wisconsin by a very slim margin, while for Hillary Kansas appears in 5th place despite her losing handily in that state, and another swing state in the form of New Hampshire appears.

Further down in the list the results appeared to be more incorrect, and for this reason we decided to compare the difference between the average sentiment in each state for each candidate. This resulted in us having a single metric that represented whether the average sentiment in the Twitter data for each state was more positive towards Trump or Hillary. An initial analysis of this metric showed evidence of some problems in correlating the sentiment data we predicted with the actual results. The top five states for the candidates were:

top trump States	top hillary states
Louisiana	Kentucky
Montana	Mississippi
Arkansas	Oregon
Delaware	District of Columbia
North Dakota	New Hampshire

According to this metric, The states where Hillary found more support than Trump included Republican strongholds Kentucky and Mississippi, while Trump's list includes Delaware, where he lost. To further confirm that this metric, and by association the calculated sentiment derived from the Neural Network, held no relationship with the results of the election, a correlation score was computed between the difference in average sentiment per state and the results in each state. The correlation score was: -0.081, showing no correlation.

6.3 Visualisations, Result Analysis and Insights

First, we created Figure 3 with the help of Plotly library to show the winners of each state. One of the most useful parts of Plotly is that it's interactive which advantage unfortunately gets lost in an offline document.

Meanwhile, Figure 4 concludes two of our most important findings of sentiment and topic analysis. On the left bar plot, you can see the polarity of those tweets that were posted about Trump. The added colour represents the winner of the election in that state. Surprisingly, there was one state with negative polarity about Trump where he actually won. Despite the data cleaning the range of the polarity varies on a small scale, roughly between -0.003 and 0.1. Deciding whether the posts were about Trump or Hillary and whether they were positive or negative per state would give us a very complicated table that does not really help us to show to the final conclusion. In order to understand these results we added +/- signs, so the positive numbers mean the average sentiment is more positive about Hillary and vice versa. E.g. Wisconsin provides us with an interesting example because even though Trump won the elections there, the tweets more tweets were negative when he was the topic.

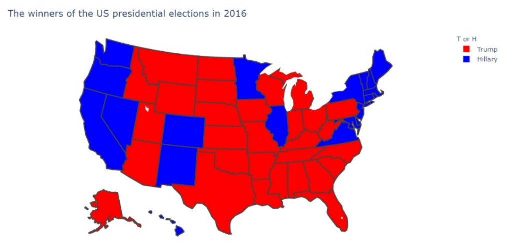

Figure 3: Majority Choice in Each State

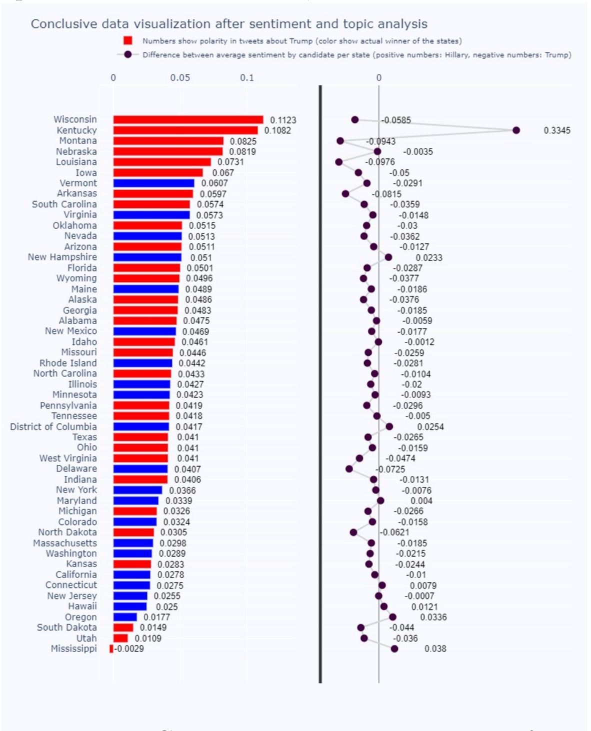

Figure 4: Conclusive Data Visualisation after Sentiment and Topic Analysis

7 Conclusion

The approach with TextBlob's polarity resulted in a slight correlation between the tweets and the vote results. However, the correlation of tweets about Hillary / Trump with the actual results was not decisively high enough to make out a clear connection between them. The Sentiment analysis found no correlation between the predicted sentiment and the actual results, while this could be a result of incorrect methodology or flawed training dataset, it could also show that social media posts aren't a good metric to use to predict the results of an election.It is also worth noting that the amount of tweets analysed is dwarfed by the total amount of tweets posted during the final stretch of the 2016 presidential campaign, and possibly the sample was not representative of the overall opinion of Twitter users.

8 References

DeGrave, K. (2016). A naive bayes tweet classifier. Retrieved from https://www.kaggle.com/degravek/a-naive-bayes-tweet-classifier

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *Processing*, *150*.

Kleppen, E. (2020). Simple sentiment analysis for nlp beginners and everyone else using vader and textblob. Retrieved from https://medium.com/swlh/simple-sentiment-analysis-for-nlp-beginners-and-everyone-else-using-vader-and-textblob-728da3dbe33d

Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. *WSDM*, 399–408. doi:10.1145/2684822.2685324