US Presidential Elections 2020

Seif Eddine Mejri

Ecole Supérieure de la Statistique et de l'Analyse de l'Information, Université de Carthage, Tunisie

mejriseifeddine@gmail.com — +216 27 122 788

Introduction

This is not not probably the first or the last project about the US elections you are going to see today,But I have tried my best to make it the most data exhausting one.

For this reason, I scraped, stored, cleaned, vectorized and analyzed the largest mega thread on reddit about us elections "Joe Biden elected president of the United States"(the equivalent of a Facebook post with tens of thousands of comments) containing more than 352k comments.

So basically, for those who didn't encounter Reddit yet, it is an American social news aggregation, web content rating, and discussion website.

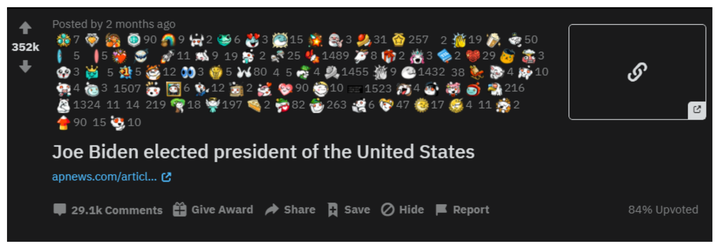

FIGURE 1 – Reddit thread

Presentation parts

I)Web Scraping and database building

II)Text Tokenization and cleaning

III)Sentiment analysis

IV)Vectorizing text

V)Building the Machine learning model, Evaluation and prediction

I) Web Scraping and database building

While Facebook and twitter made web scraping a baffling task, Reddit proposes its own scraping package PRAW, which made collecting comments an easy task through the reddit API.

Unfortunately, due my internet I was just able to collect only 22k comments from the whole 352k, but still a decent amount of data. Which was stored into a Pandas data frame.

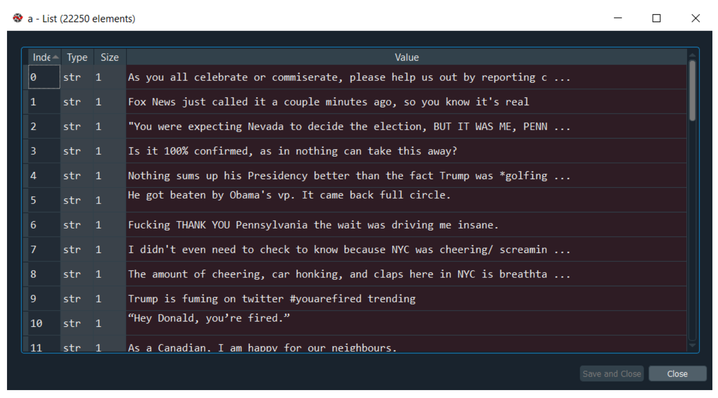

FIGURE 2 – Collected data set

II) Text Tokenization and cleaning

The natural language processing package (NLTK) multiple functions methods have been used to build cleantext(), a faction that takes a string as input and : lowers text, removes punctuation, stop words and empty tokens ,one letter words and finally lemmatizes text .

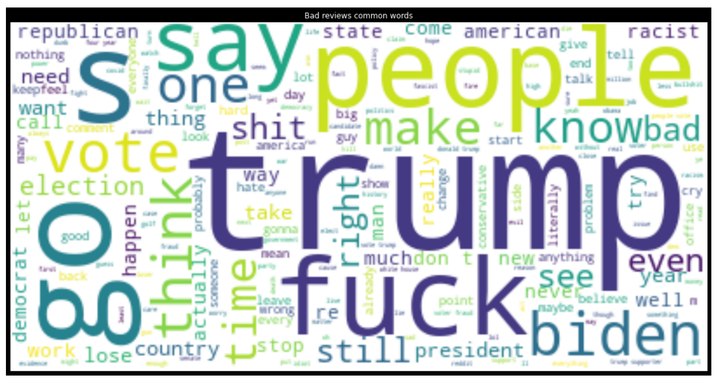

FIGURE 3 – Exploring most used words

III) Sentiment analysis

For the sentiment analysis i used the sentiment.vader from NLTK Package specifically the SIA function that produces 4 different values for each sentence (Positive, negative, neutral score and a compound score that combines all 3 scores)

sentiments_dataset - DataFrame

Index	neg	neu	pos	compound	headline	review_type
0	0.168	0.656	0.175	-0.34	a facts all:	bad
1	0	1	0	0	fox news call couple minute ago know real	neutral
2	0	1	0	0	expect nevada decide election pennsylvan…	neutral
3	0	1	0	0	confirm nothing take away	neutral
4	0.231	0.769	0	-0.3412	nothing sum presidency good fact trump g…	bad
5	0.219	0.781	0	-0.4215	get beaten obama s vp come back full cir…	bad
6	0.53	0.256	0.214	-0.5719	fuck thank pennsylvania wait drive insane	bad
7	0.193	0.568	0.239	0.296		good
8	0	0.329	0.671	0.8807	amount cheer car honk clap nyc breathtak…	good
9	0.355	0.645	0	-0.296	trump fume twitter youarefired trend	bad
10	0	1	0	0	hey donald you re fired.	neutral

FIGURE 4 – Sample of sentimentally analyzed text

IV)Vectorizing text

Unfortunately, Neural Networks don't understand text data. To deal with the issue, i will use a Natural language preprocessing package.

Nowadays, pre-trained models offer built-in prepossessing that does all the dirty work for us.

For this task i have chosen The Universal Sentence Encoder (USE)that encodes sentences into word vectors. It has great accuracy and does all the work.

V) Building the Machine learning model, Evaluation and prediction

0.1 Model Architecture

The model is composed of 2 fully-connected hidden layers and 2 Dropout layers.

the input layer is a dense layer with 512 inputs and 256 neurons. then comes the second one with 128 neurons and finally an output layer with 3 outputs 'good review', 'neutral review' and 'bad review'. and 2 more dropout layers for over-fitting prevention

```
##################### Keras Model
model = keras.Sequential()
model.add(keras.layers.Dense(units=256,input_shape=(X_train.shape[1], ), activation='relu'
model.add(keras.layers.Dropout(rate=0.5))
model.add(keras.layers.Dense(units=128,activation='relu'))
model.add(keras.layers.Dropout(rate=0.5))
model.add(keras.layers.Dense(3, activation='softmax'))
```

FIGURE 5 – Model Architecture

0.2 Model Compiling and Training

The model was compiled whith 'categorical-crossentropy' as loss function and 'Adam' as Optimizer.

The model was trained on 10 epochs, and each one of them with batch size of 16.

```
#### Compiling and fitting
model.compile(loss='categorical_crossentropy',
              optimizer=keras.optimizers.Adam(0.001), metrics=['accuracy'])
```

FIGURE 6 – Model Compiling

0.3 Model Evaluation

The model reaches just from the the 4th epoch takes a steep learning curve and does not improve much.

the model reaches accuracy of 0.7891, after 10 epochs.

which is confirmed by the test data : Accuracy : 0.7739

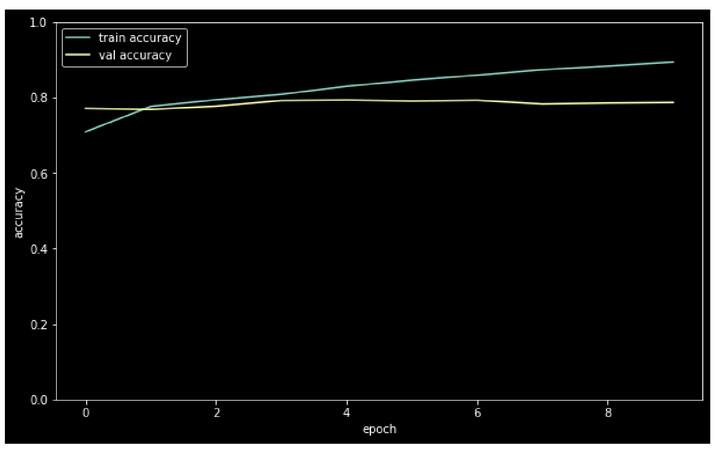

FIGURE 7 – Model Evaluation

Références

https ://curiousily.com/posts/sentiment-analysis-with-tensorflow-2-and-keras-using-python/

https ://www.parsehub.com/blog/scrape-reddit-data/

https ://towardsdatascience.com/scraping-reddit-data-1c0af3040768

https ://towardsdatascience.com/an-easy-tutorial-about-sentiment-analysis-with-deep-learning-and-keras-2bf52b9cba91

https ://arxiv.org/abs/1803.11175