A probabilistic model for text categorization
Classifying political content on Reddit*

Khaled Fouda khaled.fouda@mail.concordia.ca
Supervisor: Lea Popovic, PhD
Department of Mathematics and Statistics
Concordia University

December 2021

Abstract

Classifying social media content has been the interest of researchers during the last decade. This paper proposes a probabilistic representation of topic-related keywords on social media. We aim to estimate the conditional likelihood of a class given a short text like a tweet. We used Reddit as a case study with an interest in identifying political content. We reported the performance and compared it to machine learning methods. Our model achieves a precision of 97% and takes only a few seconds to fit over 500,000 data points.

I. Introduction

Classifying social media content has been the interest of researchers during the last decade. Being able to detect the subject of a short text is a challenging but valuable task. It allows for better product recommendations and community filtration of unwanted topics.

Reddit is a network of more than a million communities where people share interests and passions. It has 52 millions daily active users and is the 18^{th} most visited website in the world.

We have been interested in classifying political posts on Reddit. We will be using posts from known political and non-political communities to develop our model. We propose a probabilistic representation of selected topic-related keywords.

In chapter II, we review some papers with similar interests. In chapters III we discuss our model. In chapter IV we analyse the data and show the results. We also explain the choices we took while developing the model. In chapter V, we present the implementation and how to reuse the model on any data. Finally, we reflect on the model in chapter VI.

II. Related Work

Earlier work has employed machine learning in class detection. Colleoni, Rozza, and Arvidsson [3] have investigated political homophily on Twitter. They used the combination of machine learning and social network analysis to classify users as Democrats or as Republicans.

Twitter and Reddit are similar as they are both micro-blogging platforms that are used for news. However, since most Reddit users use anonymous accounts, Reddit lacks the user-network feature of Twitter. Pennacchiotti and Popescu [1] proposed a machine learning platform for large-scale classification on Twitter. They constructed user profiles using user posts and network of friends. They proposed a scoring model for the prototypical words (LING-WORD). For each class, a user is assigned a score based on the occurrences of prototypical words. We have used the main idea to build a probabilistic model for Reddit submissions.

*Honors Project Report

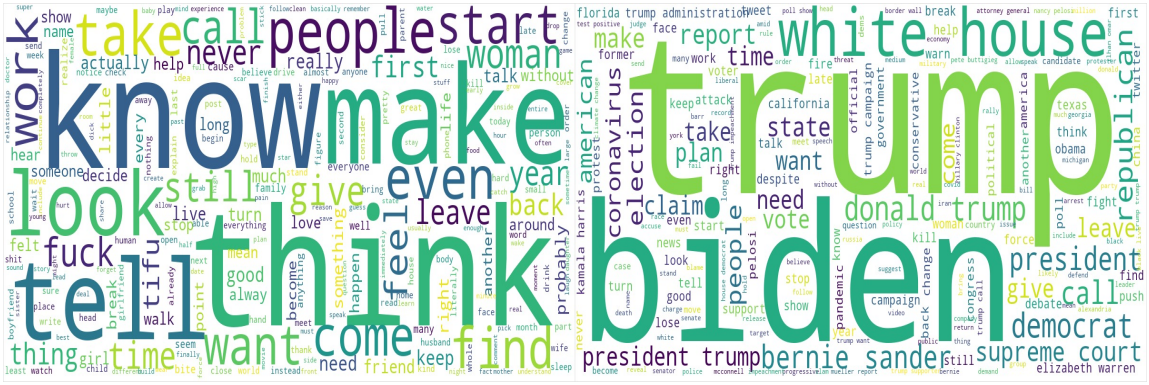

Figure 1: *word clouds of the training data grouped by class.*
The political class is on the right and the nonpolitical class is on the left.

III. Methodology

i. Data collection

Push shift [1] is a big-data storage and analytics project launched and maintained by Jason Baumgartner. The website keeps a copy of all Reddit submissions and comments[2]. The data is compressed in "ZST" format and are accessible through an API[3].

For our project, we scrapped the posts data for 2019 and 2020. The data are 120 GB while compressed and include all posts on Reddit for the two years. It was necessary to run an automated and efficient program to process the data.

We are interested in having three sets of data: political, non-political, and test data. For the political set, we chose the following subreddits: "Politics", "liberal", and "conservative". For the non-political set, we used "twoxchromosomes", "showerthoughts", "todayilearned", and "tifu". Finally, we used the Canadian community subreddit as a test set.

We ran an automated python program to go through the compressed raw data files and extract the data belonging to those subreddits. We saved the output data in JSON format. JSON objects facilitated our extraction. However, JSON files take a long time to read. We ran a second program to transform the resulting JSON objects into pandas data-frames and save them in feather format[4].

ii. Data processing and cleaning

We cleaned the data in two steps. The first step is a general automated cleaning process, and the second is a data-specific process where we tweaked the data after performing the exploratory analysis. We began by combining the title and the body of the posts in one string. We then removed words containing either nonalphabetical letters or three letters or fewer. After that, we transformed all letters into lowercase. Then we filtered the English stop words since they add no context. Finally, we converted all verbs, nouns, and adjectives into their lemmatized format. We preferred lemmatization over stemming because lemmatization does a morphological analysis of the words. Moreover, a study done by a Balakrishnan and Lloyd [2] showed that lemmatization produces better precision compared to stemming. Finally, we excluded posts with less than 20 characters in total.

[1] https://pushshift.io/
[2] https://files.pushshift.io/reddit/
[3] https://github.com/pushshift/api/
[4] https://github.com/wesm/feather/

Figure 2: *word clouds of the prototypical words of each class.*
The political class is on the right and the nonpolitical class is on the left.

After exploring the data, we added more cleaning steps. We deleted words containing the subreddits names (e.g. "tifu", "todayilearned"), deleted posts (containing keywords like "[deleted]" and "[removed]"), and URLs (start with "HTTP"). The data, especially the political data, contained popular names (e.g. "Trump", "Biden", "Bernie") and hence we removed names as well. We used the words corpus of the "Nltk" library to keep only English words. Unfortunately, we could not filter names with English meanings (e.g. "Sander" and "Macron").

For exploration, we produced word clouds, frequency tables of top words in each category, and outliers. We defined the outliers as words repeated over 30 times in a single post. Reddit's posts are short and usually under 200 words. Thus, words rarely get repeated over 20 times.

iii. WP model

We propose a probabilistic model based on the model introduced by Pennacchiotti and Popescu [1]. Let the pair (x, y) denote the training data points where x is text and y is binary, denoting the class. Let x_{new} denote unseen (unclassified) data points. Our purpose is to define the distribution of y over x_{new}. Let w denotes a single word, and W denote the list of words in all the training dataset. The length of this set is massive, and we desire to cut it down. Hence comes the need for prototypical words.

For each class, we aim to choose the most indicative k-terms. For class c, we need to select terms that appear significantly more in c while having a high probability of appearing in general. Let the collection of the chosen words be WP, a subset of W.

We want to give a score to each (w, y) pair. We will consider two probabilities: the probability that a word belongs to class and the probability that w will appear at least once.

$$P(w|y) = \frac{< w, X_y >}{||X_y||}$$

where $< w, X_y >$ is the number of occurrences of w in all x belonging to class y and $||X_y||$ is the number of data points with class y.
Next, the unconditional probability of having w is

$$P(w) = \frac{\sum_x I(w \in x)}{N}$$

where N is the dimension of X.
We will define the scoring function $PScore(w, y)$ as the harmonic mean of the two previous probabilities.
The harmonic mean of two quantities a and b is defined as $\frac{2ab}{a+b}$. The harmonic mean provides scores between 0 and 1 while giving equal importance to the two quantities. That makes it more efficient than the average mean

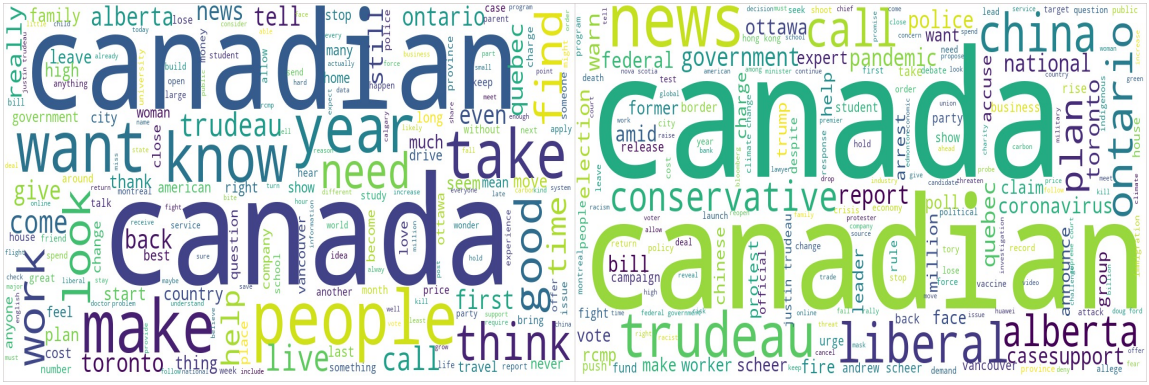

Figure 3: *word clouds of predicted classes in the test set.*
The political class is on the right and the nonpolitical class is on the left.

which would give a word appearing only once a high probability.

The dimension of the scoring matrix is $(D, 2)$ where D is the total number of words in the training set.
We will keep the top k words in each class. k is a hyper parameter which we will discuss in the next section.

$$WP = (\arg\max_{w} PScore(w, y))_{y=(0,1)}$$

The resulting set has a size of $(2K)$ which is significantly smaller than D. That would allow us to perform complex operations with less concern for the computing resources.
Next, we define the frequency matrix. The rows are posts and the columns are words.

$$NUME(wp, x) =< wp, x >= \sum_{a \in x} I(a = wp)$$

We seek the probability of having a response y given a prototypical word wp. Because predictors x are assigned to the responses, we will condition on them.

$$P(y|wp) = \sum_{x} P(y, x|wp) = \sum_{x} P(y|wp, x)P(x|wp)$$

$$P(y|wp) = \sum_{x} I(y_x == y)P(x|wp) \qquad (1)$$

The probability of having a predictor x given a prototypical word wp is the number of times wp occurred in x divided by the total number of occurrences of wp.

$$P(x|wp) = \frac{NUME(wp, x)}{||wp||}$$

where,

$$||WP|| = \text{col sum}(NUME)$$

Finally, we compute the probability of having a response given new data. This probability is proportional to sum of the probabilities of having the response y given the prototypical words appearing in the predictor.

$$P(y|x_{new}) \propto \sum_{wp \in x} p(y|wp)$$

$$y^* = \arg\max_{y \in (0,1)} p(y|x_{new})$$

iv. Performance assessment

We used the harmonic average of the precision and recall to assess our model. This average is known as the F-1 score and considers both the precision and recall to be important.
We will define the criterion non-responses as the proportion of predictors with no prototypical words. A high k value will guarantee a low non-responses value but it will lead to a slow model.
Moreover, we trained a Random Forest model

X	Y
police investigate possible hate crime proud take burn black live matter sign historically black church	1
modern belief gender attribute biological would allow sentient claim membership gender foundation defend civil	0
house dems demand barr cancel press conference mueller report	1

Figure 4: *A random sample of 3 rows of the training dataset. X is the cleaned text and Y are the labels (1 for political and 0 for non-political).*

Political	Non-Political	Canada
trump	like	canada
biden	make	canadian
house	know	trudeau
coronavirus	think	news
call	tell	would
democrat	would	make
president	want	like
white	take	want
election	time	people
vote	people	government
republican	feel	year
sander	start	ontario
state	year	call
donald	come	liberal
campaign	look	take

Table 1: *Top 15 keywords in each class*

to compare it with our model. We used a randomized search method to choose the hyperparameters of the model.

IV. Analysis and Results

Our study case is identifying political posts on Reddit. In table 1, we see the most common keywords in each class in the training set. We notice that top political terms are either the names of politicians or political jargon. Meanwhile, the non-political keywords are all words of thoughts and feelings. We have likewise included the top keywords in the test dataset (Canada), and we observe a mix of political and non-political keywords.

In figure 1, we drew word clouds of both classes. We note that American politics is heavily affecting political data. We considered excluding names and non-English words from the data to generalize our model. However, we will show later that it will not affect our prototypical terms.

Our training dataset contains 505,676 rows. We reserved 20% of the data for validation. The proportion of political labels is 68.3%. In figure 4, we show a sample of processed data along with their corresponding labels. For our study case, $y = 1$ corresponds to political posts and $y = 0$ to non-political posts.

The total number of words in the training set is 6,430,767. However, the number of unique words is 74,019. We first aim to choose a subset of prototypical words. We used cross-validation to determine the optimal number of prototypical words. We analyzed both the f-score and the percentage of non-responses of the validation set. The results illustrate that values over 1000 do not improve the model by much. Hence, we set k to 1000. The resulting number of prototypical words is 1490 because of words belonging to both classes. Only 1% of the validation data lack any prototypical word. The corresponding f1 score is 0.9. Later, we show that our model has a very high precision score. In table 2, we display the results of applying cross-validation over different values of k.

In Figure 2, we have word clouds of the prototypical words for each class. As suggested earlier, American politics are not as influential in the prototypical terms as in the training set. In table 4, we display a random sample of prototypical words along with their probabilities (see equation 1).

k	non-responses	f1 score	f1 score adjusted[a]
200	6.53%	0.847	0.875
400	3.06%	0.875	0.886
600	1.87%	0.887	0.893
800	1.36%	0.894	0.898
1000	1.04%	0.898	0.902
1200	0.84%	0.901	0.903
1400	0.7%	0.903	0.904

Table 2: *cross validation results*

[a]Excludes non-responses

wp	$P(y=0\|wp)$	$P(y=1\|wp)$
vote	0.04	0.96
scene	0.85	0.15
third	0.55	0.45
another	0.75	0.25
shock	0.74	0.26
poll	0.01	0.99
fear	0.45	0.55
match	0.81	0.19
invite	0.66	0.34
actually	0.88	0.12
service	0.39	0.61
definitely	0.93	0.07
promise	0.31	0.69
slam	0.13	0.87
reelection	0	1
advice	0.84	0.16
lincoln	0.25	0.75
window	0.91	0.09
campaign	0.03	0.97
survive	0.78	0.22

Table 3: *A random sample of prototopyical words with their probabilities of belonging to each class. (y=1 for political and y=0 for non-political)*

We applied the model to the training and validation set. We adopted the scoring methods illustrated in the previous section to evaluate the model. We have also trained a Random Forest model on the data. The results in table 4 show that our model has greater precision and lower recall. We believe the reason is that it is easier to detect political keywords than general keywords. We hope that by generalizing the model to consider more classes, we can ensure a higher f1 score. The training time (The time to calculate the probabilities) is 0.09 minutes, compared to 65 minutes for the machine learning model. However, the prediction time is higher since we need to match our list of prototypical words to each post.

Finally, we tried the model on the test dataset. Our test dataset is unlabelled, and hence we cannot directly compute the scores. In figure 5, we present a random sample and their predicted labels. We validated the results as well. We found 2 out of the 15 random samples to be misclassified. Finally, in figure 3, we show word clouds of the text of the predicted classes of the test set.

V. Implementation

Below, we illustrate how to import and apply the probabilistic model. We divided the implementations into two parts: data preprocessing and model fitting. We can reuse the

	WP model		Random Forest	
	Training	Validation	Training	Validation
Non-responses	0.01	0.01		
Accuracy	0.87	0.87	0.92	0.92
F1 score	0.90	0.90	0.94	0.94
Precision	0.97	0.97	0.95	0.95
Recall	0.83	0.83	0.94	0.93
Training time (min)	0.09		65	
Prediction time (min)	2.24	0.7	1.01	0.14

Table 4: *Training results of our model (k=1000) and the optimal random forest.*

id	text	class	Accurate?
4376	federal payroll system could almost double memo	non-political	No
29990	leger bloc green	political	Yes
19418	winter storm could break snowfall record single toronto	non-political	Yes
10526	citizen would canada require temporary residency visa week vacation curious ...	non-political	Yes
5858	house help immigrant family close wealth	political	Yes
5983	political compass veer right	political	Yes
28686	special force seek outside intelligence advice intelligence expert ...	political	Yes
13115	prisoner right vote federal	political	Yes
3968	andrew andrew scheer find difficult right	non-political	No
31426	tasha plan serve union business parent	political	Yes
18713	father lucky enough travel wonderful stay fremont alberta ...	non-political	Yes
11771	canadian reddit tourism question someone outside canada plan ...	non-political	Yes
25108	albertan government purge public judicial appointment oversight committee ...	political	Yes
29867	leader singh accuse liberal fail support canadian live disability pandemic	political	Yes
14416	climate activist greta thunberg speak edmonton rally	political	Yes

Figure 5: *A random sample of 15 rows of the test (Canadian) dataset. The text is cleaned and lemmatized and each prediction was hand-validated.*

former to download Reddit submissions for the same or any other subreddits or years. Moreover, the probabilisitc model can be applied to any data. For full details, please read the "*code_structure.md*" file on the project's GitHub page[5]. Below, we provide an example of importing and applying our model to any data.

VI. Epilogue

We note that the model captures the subject-related keywords with high precision and short training time. However, there are still some deficiencies and opportunities for improvement. In the longer term, we plan to generalize the model for multiple classes, which requires new probability equations and working the likelihood with n variables.

For this project, we considered the submissions of 5 subreddits. However, over 50% of the content in top subreddits are links, with 52% of them linking to websites[6]. We believe we could increase the performance by crawling the contents of the web pages. Adding comments to the model would increase the accuracy. With the aid of the multi-label model, we could include more subreddits and assign each a proper label.

Finally, we note that the prediction time of the model is lengthy. Text processing is a challenging task for many programming languages. For the next steps, we will consider faster methods for prediction.

References

[1] Marco Pennacchiotti, Ana-Maria Popescu *Democrats, Republicans and Starbucks Afficionados: User Classification in Twitter* [*https://doi.org/10.1145/2020408.2020477*].

[2] Vimala Balakrishnan, and Ethel Lloyd-Yemoh *Stemming and lemmatization: A comparison of retrieval performances* [*http://eprints.um.edu.my/id/eprint/13423*]

[3] lanor Colleoni, Alessandro Rozza, Adam Arvidsson *Echo Chamber or Public Sphere? Predicting Political Orientation and Measuring Political Homophily in Twitter Using Big Data, Journal of Communication, Volume 64, Issue 2, April 2014, Pages 317–332* [*https://doi.org/10.1111/jcom.12084*]

[5] https://github.com/khaledfouda/Plitical-changes-on-R-canada–Reddit-

[6] https://foundationinc.co/lab/reddit-statistics/

```
import pandas as pd
from wp_model import Proto # ensure that the file is in the same folder
from 6_clean_all import clean # the data cleaning function.
data = pd.read_csv('...') # ensure that the data is in data frame format
cleaned_data = clean(data, 'x', 'y', 'output_file.feather')
# The second and third inputs are the names of the predictor and response columns
# The last parameter is the name of the file where the clean data will be saved to.
model = Proto('label', k=1000, log_to_file=True, harmonic_pscore=True)
# This initializes a model object. The first parameter is a label. It can be any
    word, it's used to identify model output files and logs.
model.fit(cleaned_data)
# calculates the probabilities
model.train_valid_predict()
# predicts the labels for the training and validation data and produces the tables
    and graphs in this report.
preds = model.test_predict(test.X)
# returns predictions for the input text series.
model.fit_cv(data) # runs the cross-validation for the k value.
```

Figure 6: *Example of importing and applying the WP probabilistic model.*