Data Science and Economics

Text mining and sentiment analysis

Sarcasm detection on Reddit

Andrea Joseph Froiio

922907

Index

Chapter 1

Introduction

The aim of this paper is, given **only** the parent comment and the Reddit category (subreddit), to predict the probability of that parent comment to receive a sarcastic reply.
Cross Validation is used to evaluate the accuracy of the models tested.
These tasks are performed on a dataset from Reddit, a social network in which sarcastic comments are usually self-annotated.
The existing literature is mainly focused on training models to find sarcasm in a text itself, not whether a reply to that will be sarcastic, which is far more difficult to predict.
The models used are:

- *Logistic Regression*, it is a model used to estimate the probability of an event with binary outcome.
- *Naive Bayes*, it is a simple probabilistic classifier satisfying:

  $$f^*(x) = \operatorname*{argmin}_{\hat{y} \in \mathcal{Y}} \mathbb{E}[\ell(Y, \hat{y}) | X = x]$$

  Where $\mathcal{Y}$ is the distribution of the labels and ℓ is the loss function over the real label Y and the predicted one $\hat{y}$, conditioned on the datapoint x.
- *Support Vector Machine*, it is a classification technique that splits the data in the best possible way, which means that it finds the farthest hyperplane from the "support vectors" by maximizing the margin between them, making it the one with the least probability of misclassification.

Chapter 2

Research question and methodology

What is the probability that a given comment in a certain subreddit will receive a sarcastic reply?

Some models will be tested and combined with different text mining approaches.

2.1 Sarcasm

Sarcasm is saying the opposite of what you mean for the purpose of humour or criticism; usually the underlying meaning is obvious but not always.
When spoken, sarcastic sentences rely on verbal cues such as a mocking tone and body gestures such as eye and hand movement that are lacking when working with textual data.

2.2 Stemming

Stemming is the process of reducing words to their root forms even if the stem itself is not a valid word in the chosen language.
The stem is the part of the word to which inflectional affixes are added, such as -ed, -ize, -s, -de, -mis and so on.
Another approach similiar to stemming is *lemmatization*, in which words are reduced to their root forms using a dictionary of actually existing words.

2.3 TF-IDF Vectorization

Term Frequency - Inverse Document Frequency is a vector space model in which terms, which may be words or n-grams (group of n words), are given higher weights if they appear frequently in the document but are not common in the corpus (the collection of documents).
A term weight is computed:

$$w_{d,t} = TF_{d,t} \times IDF_t$$

Where:

- $TF_{d,t} = \frac{n_{d,t}}{\sum_i n_{d,i}}$
  - $n_{d,t}$ is the number of times the t-th term appears in the d-th document.
- $IDF_t = log_{10}\left(\frac{N}{N_t}\right)$
  - N denotes the total number of documents.
  - N_t denotes the number of documents containing the t-th term.

Chapter 3

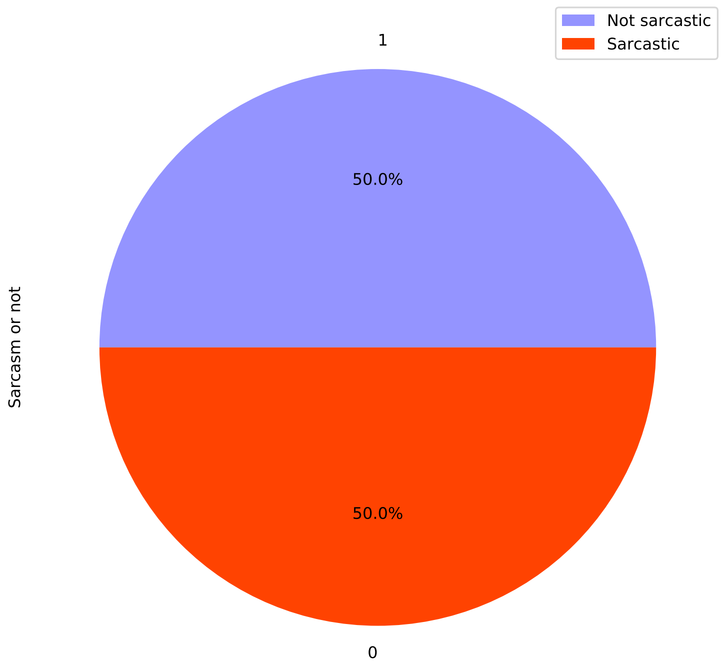
Experimental results

3.1 Dataset

The dataset used for the experiment is a subset of the *Self-Annotated Reddit Corpus* (**SARC**) which contains 1010826 comments from Reddit, a social network divided into subreddits (categories, topics) where users can reply to comments in a tree-like structure and upvote-downvote them.
It is a good dataset because there is a "convention" on Reddit where a sarcastic comment is followed by a **/s** in its body, which explains the "Self-Annotated" title.
For each comment, they are provided:

- *label*, indicates whether the comment is sarcastic (1) or not (0).
- *comment*, the raw comment (without the "/s") to which the label is referred.
- *author*, the username of the redditor who posted the comment.
- *subreddit*, the "area of interest" in which the comment was posted, can be seen as the topic.
- *score*, the amount of upvotes minus downvotes, but some say it is also affected by the time passed since the comment has been posted.
- *ups*, the amount of upvotes received by the comment.
- *downs*, the amount of downvotes received by the comment.
- *date*, the month and year of posting.
- *UTC*, the exact time and date of posting, following Coordinated Universal Time.
- *parent*, the parent comment under which the comment which is sarcastic or not was posted.

For our task, only the *label*, the *subreddit* and the *parent* comment are needed, since we have to predict the probability that a new comment will receive a sarcastic reply (in a given subreddit), not that a comment is sarcastic itself.
Labels are balanced between the two outcomes:



3.2 Evaluation strategy

The evaluation strategy to evaluate the performances of the models used is a 5-fold Cross Validation which works like this:

- The set is split into 5 smaller sets.
- A model is trained using 4 of the folds as training data.
- The resulting model is tested on the remaining part of the data.
- The performance measure reported by a k-fold Cross Validation is the average of the values computed in the loop, 5 in our case.

3.3 Experimental methodology

First of all it is checked whether some subreddits are more inclined to sarcasm than others, to confirm that that they will indeed have an impact on the classifier performances, here the ”most sarcastic” ones between subreddits with at least 500 entries:

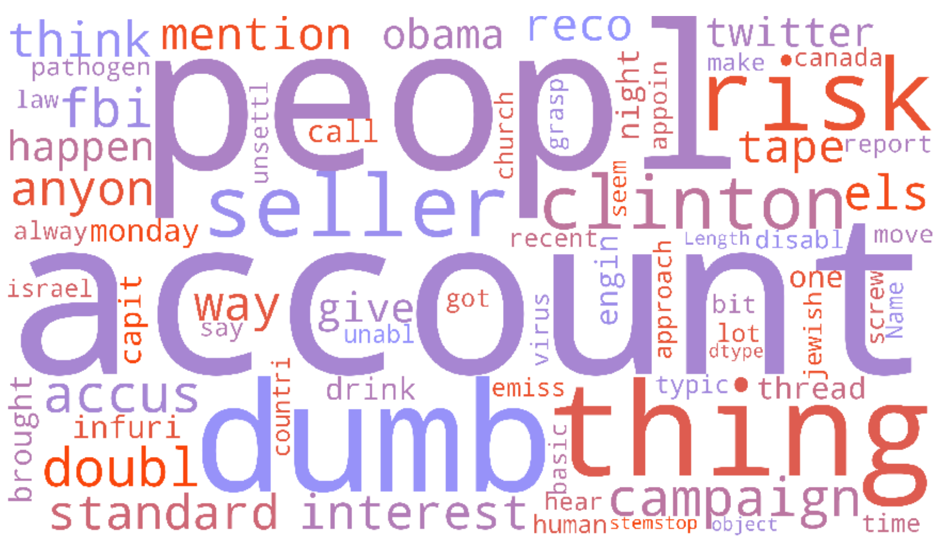
subreddit	**size**	**mean**	**sum**
creepyPMs	5466	0.784303	4287
progun	541	0.709797	384
rage	886	0.697517	618
MensRights	3356	0.680870	2285
Bad_Cop_No_Donut	902	0.678492	612
ShitRedditSays	1284	0.661994	850
niceguys	910	0.653846	595
conspiratard	663	0.653092	433
worldnews	26377	0.642529	16948
Libertarian	2562	0.640125	1640

3.3.1 Raw text

Comments are passed to a TF-IDF Vectorizer keeping 1- and 2-grams, while only 1-grams for subreddits.

Model	**5-fold CV score**
Logistic Regression	0.5912877202743936
Naive Bayes	0.5942931824855989

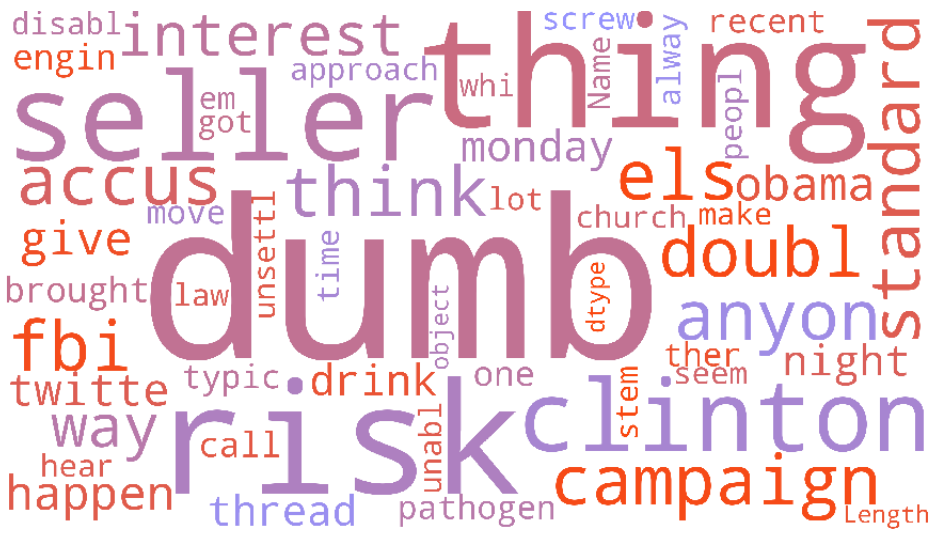
3.3.2 Stemming and stopwords



Before being passed to the same TF-IDF Vectorizer, comments are stemmed and stopwords are **removed**.

Model	5-fold CV score
Logistic Regression	0.5841183345239652
Naive Bayes	0.5873374846370086

3.3.3 Stemming



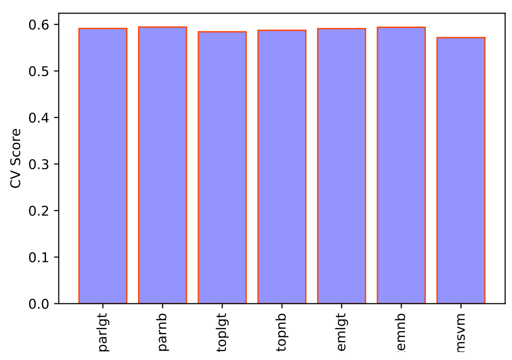
Before being passed to the same TF-IDF Vectorizer, comments are stemmed.
Stopwords are **kept**.

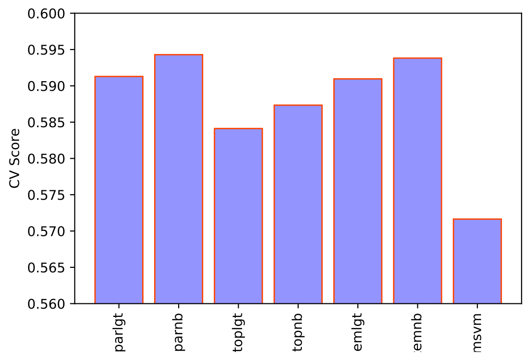
Model	**5-fold CV score**
Logistic Regression	0.5909582864646342
Naive Bayes	0.5938183228852901
Support Vector Machines	0.5716424007967985

Chapter 4

Concluding remarks

Despite not being precise, the best model, though by a bit, is the Naive Bayes classifier on raw data after it has been vectorized, which could be due to the fact that all information is precious, not only stopwords, but also the suffixes:





Where, respectively:

- *parlgt*, raw parent comment vectorized and passed to a Logit regression.
- *parnb*, raw parent comment vectorized and passed to a Naive Bayes classifier.
- *stemstoplgt*, parent comment stemmed and stopwords removed, vectorized and passed to a Logit regression.
- *stemstopnb*, parent comment stemmed and stopwords removed, vectorized and passed to a Naive Bayes classifier.
- *stemlgt*, parent comment stemmed, vectorized and passed to a Logit regression.
- *stemnb*, parent comment stemmed, vectorized and passed to a Naive Bayes classifier.
- *stemsvm*, parent comment stemmed, vectorized and passed to a Support Vector Machines classifier.

Due to the fact that we are not detecting sarcasm in a comment itself, the lack of many important features for sarcasm recognition can be ignored as we are focusing only on the parent comment; but what can lead to a sarcastic reply?
Usually when the topic in question is controversial and people will mock and make fun of the upper comment.
The most "controversial" topics should have been covered by the *subreddit* feature and more in depth by the words used in the comment.
The imprecision may be due to the fact that the same comment can receive a different reply based

on who is answering, if he is agreeing or not, if he is someone who usually makes fun of things or not; all this is information that we do not have.
A way to improve the classifier could be to have more information about the post under which comments are posted to understand the context of the discussion and analyze its title (body too, if it is an article) to give a polarity to it, whether it is bad or good, to which people tend to react differently.

The model is tested on a new comment regarding Donald Trump, who is not in a good light with the Reddit community and often the cause of many sarcastic comments:

subreddit	**parent**	**Probability of receiving a sarcastic reply**
politics	Trump is very good	0.61990203

"A sarcasm detector? That's a real useful invention."
Comic Book Guy, The Simpsons - Source

Chapter 5

References

1. Khodak Mikhail, Saunshi Nikunj, Vodrahalli Kiran (2017). *A large self-annotated corpus for sarcasm.*

2. Eke Christopher Ifeanyi, Norman Azah Anir, Shuib Liyana, Nweke Henry Friday (2019). *Sarcasm identification in textual data: systematic review, research challenges and open directions.*

3. Joshi Aditya, Bhattacharyya Pushpak, Carman Mark J. (2017). *Automatic sarcasm detection: a survey.*