R Final Assignment
Will Power-Jenkins
INF2167
Monday, December 13th, 2021

Classification: Satirical Distinction in News Article Headlines

1. Background

This project attempts to classify and distinguish whether an article headline was written by the satirical journalism & media company *The Onion* or was instead one posted on Reddit's /r/nottheonion, a sub-Reddit (or sub-forum) that collects articles from legitimate news sources that can be misconstrued as satirical out of context. This distinction holds important implications for the detection of false-news, disinformation, and misleading headlines. For this project to be applicable to those classification models studied in class, these headlines first needed to be run through sentiment analysis packages to determine, numerically, the overall positive or negative connotations of the headlines as well as the associated emotions with each. Results are not promising, which means either it is difficult for machines to differentiate between article headlines of this type, or else more robust sentiment analysis packages are required. This latter hypothesis seems much more likely, as GitHub user lukefeilberg was able to obtain 87% accuracy in their machine learning model using Python and the Keras library, and Rahman et al. achieved similar results in Python on contextualizing news headlines.

Attempts were made with the *FeatureHashing* and *XGBoost* packages in R but were unsuccessful. Instead, the *syuzhet* package was chosen, utilizing four different sentiment analysis methods as well as a normalization factor to obtain a wide variety of sentiment scores from which to train models on. For future analysis, Naldi suggests 3 further packages which may yield better results: *Rsentiment*, *SentimentR*, and *Sentiment Analysis*. Results and future approaches are discussed in more detail in the last two sections of this paper.

2. Data Source & Cleaning

Data was collected using the *RedditExtractoR* library created by Ivan Rivera from CRAN. Using this library allowed the downloading of article headlines directly from applicable subreddits. The two forums chosen were /r/theoion and /r/nottheonion, the former including the most popular Onion headlines of all time (those assigned the "top" category) and the latter including articles that might be misconstrued as satirical headlines based on their outlandishness utilizing the same criteria.

```
drop <- c("text", "date_utc", "comments", "url")
onion_list <- find_thread_urls(keywords = NA, sort_by = "top", subreddit =
"theonion", period = "all")
onion_list <- onion_list[,!(names(onion_list)%in%drop)]
rownames(onion_list) <- 1:nrow(onion_list)
```

Existing datasets were available and were generally trustworthy based on the type of data being collected, but collecting it manually ensures the project can be self-contained and guaranteed integrity. Headlines from The Onion were assigned a label of 1, while headlines from /r/nottheonion were assigned a variable of 0. The lists were then combined into a data frame and the order randomized. The file was stored to a .csv for later access, verification, and record-keeping.

3. Goals & Problem

The classification models constructed had the goal of attempting to be able to distinguish between satirical news article headlines and news headlines that only *appeared* to be satirical but were in fact listed from legitimate news sources. This problem is difficult because even humans may have a hard time distinguishing satirical humour from factual headlines, especially when situations are either themselves out of the ordinary or are represented linguistically as such. For instance, the article headline “Man Tries to Rob a Bank After Paying $500 to a Wizard to Make Him Invisible”, sourced from /r/nottheonion, has the potential to be widely misconstrued as fiction because of the ostensibly ridiculous nature of the scenario. However, syntax and choice of language here are especially important as well. An event, scenario, or otherwise reportable item of public interest can be conveyed through language to the viewer or reader in any number of ways, a reflection of the infinite complexity of language (and, likewise, of the as of yet insurmountable complexity of getting machines to truly understand it, especially subsets of language such as irony, satire, or sarcasm) (Reyes & Rosso, 2013).

If natural language understanding could be achieved, we might be able to build better fake-news and clickbait detection systems and help curb the spread of falsified information online, a source of many current political issues. In whole, any system that recognized misleading headlines would also need to be able to (either by itself or in tandem with another model) analyze the text of the article as well and determine whether the headline was 1) verifiable and factual, and 2) whether it represented accurately what was attempting to be conveyed in the article body.

4. Literature Review

Reyes and Rosso explore the concept of irony (which can be viewed as akin to, or the parent branch of, satire) and the complexity involved in detecting it using ML models. In irony, language is generally figurative, which “unlike literal language, masks its real meaning by exploiting different linguistic devices to veil its underlying and real meaning” (2013, p. 596). They propose avoiding a cornucopia approach and instead suggest identifying “specific aspects of irony that are susceptible to computational analysis, and from these individual treatments, attempt to synthesize a gradually broader solution" (2013, p. 596).

In a more related area, Rahman *et al*. explore ML models that classify news article headlines from several news sources based on their polarity and propose models that can detect greater neutrality in wording so as not to automatically bias people when they glean initial information from the title of the news article. In a similar vein, Peterson and Spirling analyze the speeches of Members of Parliament in the British House of Commons to determine which political party they belong to, and the level of polarization that is evident in their language. When the classifier has high accuracy, meaning the model can distinguish members between the two main parties (labour and Conservative) it is said that that the period in which the speeches were analyzed represents an era of high political polarity. Likewise, when the model is less accurate, the period in question represents one of low polarity.

Identifying political bias in headlines is not the goal of this project, but it represents a similar problem in that how readers interpret headlines greatly influences how they perceive subsequent information and how during periods of high political polarization language can be a key indicator of the information landscape.

5. Methods

The constructed model would be one that could download data from the relevant sources, clean and analyze the text, and then classify the headlines based on whether or not they were satire or only ostensibly appeared to be satire. As discussed above, using the RedditExtractoR package, headlines were downloaded from two subreddits: /r/theonion and /r/nottheonion. A quick initial data scrub was performed (dropping unnecessary columns such as 'date' and 'comments') as well as a label assignment of 1 or 0 depending on whether the article title came from /r/theonion or /r/nottheonion, respectively. The lists were then combined as a dataframe and randomized using the *sample()* function. After being stored to a .csv for later verification and retrieval, the data was then cleaned using the *str_replace_all* function included with the *stringr* package, which consisted of removing punctuation, unrecognized characters, and symbols such as ampersands.

```
final_list$title <- tolower(final_list$title)
final_list <- final_list[,!(names(final_list)%in%"X")] # drops the old index
final_list$title <- str_replace_all(final_list$title, "[]", "")
final_list$title <- str_replace_all(final_list$title, "[']", "")
final_list$title <- str_replace_all(final_list$title, "[-:;“”]", "")
final_list$title <- str_replace_all(final_list$title, "[&]", "and")
```

This included separating out all lines into a corpus (using the *tm* package) and then removing numbers, stopwords, whitespaces, and reducing words to their root form using the *tm_map* function.

```
bulk_text <- final_list[,1]
TextDoc <- Corpus(VectorSource(bulk_text))
TextDoc <- tm_map(TextDoc, removeNumbers)
TextDoc <- tm_map(TextDoc, removeWords, stopwords("english"))
TextDoc <- tm_map(TextDoc, removePunctuation)
TextDoc <- tm_map(TextDoc, stripWhitespace)
TextDoc <- tm_map(TextDoc, stemDocument)
```

Below is some basic analysis of word frequency. This list excludes stopwords such as ‘a’, ‘the’, ‘as’, and any other very common English words that would skew frequency analysis.

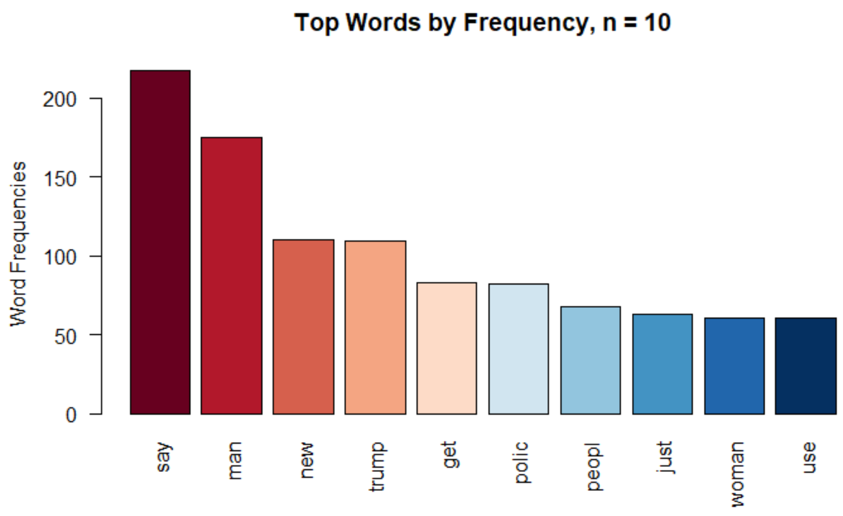

Fig. 1: Article headline word frequency

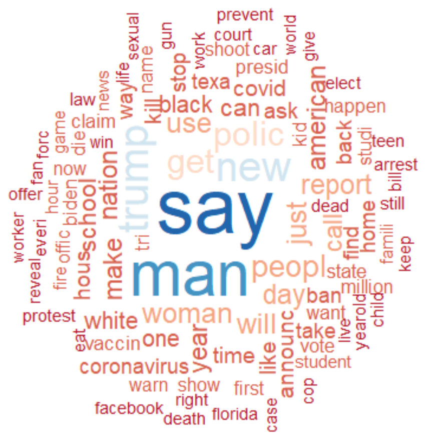

Fig. 2: Wordcloud of most common words in article headlines (n = 100)

As we can see, many of the most common words are politically charged ('Trump', 'Covid', 'police'). Considering the implications of reading satirical news articles as reality (and likewise reading non-satirical news articles as satire) it becomes clear that ML models have a lot of responsibility if they are to achieve accurate distinction abilities with informational language.

Using the *syuzhet* package, three methods were used for basic sentiment analysis: *syuzhet*, *afinn*, and *bing*. These methods all incorporate different lexicons and scales. The most general, *syuzhet*, contains 10,748 words (3,587 positive and 7,161 negative) and utilizes a scale of -1 to 1, with the score of each word being added cumulatively (e.g., a sample headline ended up with a finalized score of -1.40) (Naldi, 2019).

```
syuzhet_vector <- get_sentiment(text, method = "syuzhet")
head(syuzhet_vector, 10)
summary(syuzhet_vector)
sentiment_list <- cbind(final_list, syuzhet_vector)
```

The other two, *afinn* and *bing*, have lexicons comprising of 2,477 words (878 positive and 1,598 negative) and 6,789 words (2,006 positive and 4,783 negative), and utilize scales of -5 to 5 and values of either 1 or -1, respectively (Naldi, 2019). The *afinn* method in particular has potential to prove useful in this context, as Naldi notes it contains “Internet slang and obscene words” in its lexicon (2019, p. 3). These scores were then normalized across vectors using a scale of -1 to 1 and added to the dataframe.

```
normalized_vectors <- rbind(
  sign(syuzhet_vector),
  sign(bing_vector),
  sign(afinn_vector)
)
```

Next, the *nrc* sentiment analysis method was run to see how 8 particular emotions (anger, anticipation, disgust, fear, joy, sadness, surprise, or trust) scored amongst the corpus of words aggregated from the headlines.

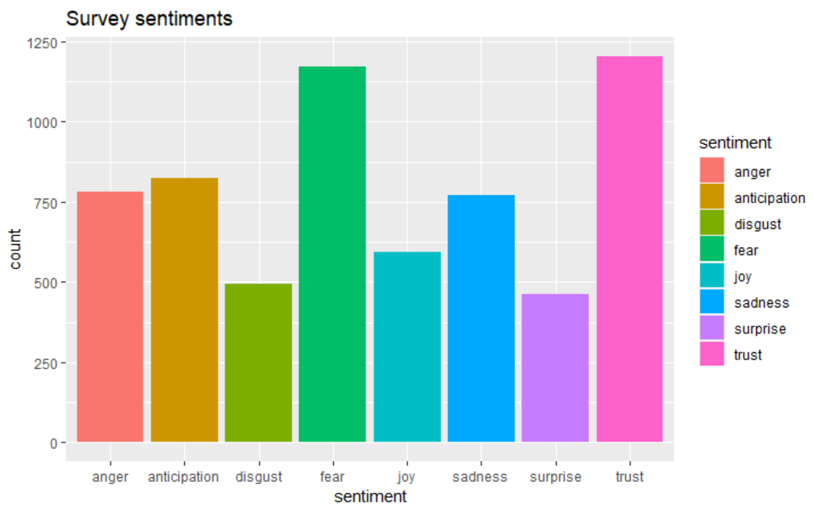

Fig. 3: Words associated with particular sentiment in headlines

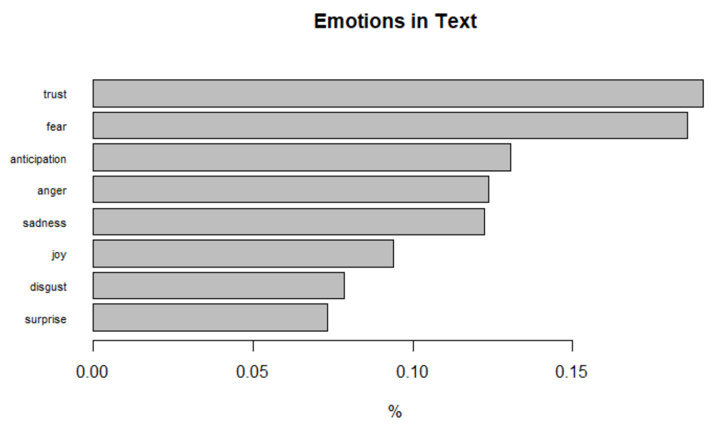

Fig. 4: Relative percentage of associated emotions in headlines

Perhaps this is not the most revealing analysis, but it is interesting to note how the *nrc* method has interpreted the overall emotional charge of the headlines.

6. Optimization & Results

Once these scores had all been encoded as integers, a wide variety of classification algorithms was used to attempt to create predictive models. These included the general linear model, linear discriminant analysis, quadratic discriminant analysis, as well as binary decision trees using both the general *rpart* package and the *randomForests* package. The GLM, LDA, QDA, binary decision tree, and random forest methods were then all re-checked using cross-fold validation. The results are presented in the table below:

Method		MSE
General Linear Model		0.5972
General Linear Model (k-fold CV)		
	k = 10	0.5771
	k= 100	0.5671
	k = 1000	0.5660
	k = nrow[1]	0.5656
Linear Discrete Analysis		0.5992
Linear Discrete Analysis (k-fold CV, k = nrow)		0.5009
Quadratic Discrete Analysis		0.5971
Quadratic Discrete Analysis (k-fold CV, k = nrow)		0.5014
Binary Decision tree (rpart)[2]		0.4945
Binary Decision tree (rpart) (k-fold CV, k = nrow)		0.4973
Random Forest		0.4830
Random Forest (k-fold CV, k = 100)		0.4896

[1]nrow = 1974

[2]Pruning was not performed as there were too few branches to dictate its necessity

As we can see, the results are paltry and less than desirable, and cross-validation, which ensures higher accuracy and a more trustworthy MSE, exacerbates the display of ineffectiveness of this algorithm by up to 10% in some cases. It is true that some of these methods, like random forests, are not as applicable to a dataframe with such a low variety in values and so don't represent the most effective way of predicting headline association in this case, but even the most effective models are sub-par at best. Therefore a project such as this is more likely to benefit from alternative sentiment analysis packages (or even another approach outside of sentiment analysis entirely) such as the ones Naldi suggests. While the *syuzhet* package is easy to work with, its applications lie elsewhere in areas such as general text analysis and not in the detection of satire, so least as it was used in this manner. The nature of satirical headlines demands a different analytic approach.

7. Discussion

While this project was carried out with humorous undertones, it is nonetheless an important area of exploration that can potentially help those attempting to track and monitor falsified or misleading headlines in the fight against mis/disinformation. By attempting to best determine whether an article's title is accurate and forthcoming or whether it is deceptive and disingenuous, researchers can help prevent readers from falling into mis-informational traps and to teach them to think critically about what it is they are reading online. While this is a far-cry from completely expunging campaigns that spread falsified information using internet technologies, it can act as a very important first step. Of course, one such limitation of a machine learning model with so narrow a scope is that it does not examine what is contained within the article itself, but rather looks only at the titles to determine its classification decisions. Further analysis and machine learning would be needed in conjunction with this model to critically evaluate what language is contained within the text itself such that it can best determine 1) whether the headline is an appropriate summary and representation of the text and 2) whether the text itself is misleading, falsified, or otherwise contributory to a furthered separation from fact and fiction.

References

lukefeilberg. (2020). onion. GitHub repository, https://github.com/lukefeilberg/onion

Naldi, M. (2019). A review of sentiment computation methods with R packages. *arXiv preprint arXiv:1901.08319*.

Peterson, A., & Spirling, A. (2018). Classification accuracy as a substantive quantity of interest: Measuring polarization in Westminster systems. *Political Analysis, 26*(1), 120-128. doi:10.1017/pan.2017.39

Rahman, Hossain, S. S., Islam, S., Chowdhury, M. I., Rafiq, F. B., & Badruzzaman, K. B. M. (2019). Context-based news headlines analysis using machine learning approach [*sic*]. Computational Collective Intelligence, 167–178. https://doi.org/10.1007/978-3-030-28374-2_15

Reyes, & Rosso, P. (2013). On the difficulty of automatically detecting irony: Beyond a simple case of negation. *Knowledge and Information Systems*, 40(3), 595–614. https://doi.org/10.1007/s10115-013-0652-8