Sentiment Analysis of User Comments on Machine Learning Posts: A Data-driven Approach Using R*

Joseph Chung

Apr 20, 2023

Abstract

In this study, we present a comprehensive data analysis paper that utilize a sentiment model in R to analyze user comments from a dataset containing post_ids and comments related to the topic of Machine Learning. We aim to investigate sentiment polarity of user comments and its distribution across different posts, identifying trends and patterns that may provide valuable insights into the Machine Learning community's perception and engagement.

Table of contents

*Code and data are available at: https://github.com/UtopianYoungChung/Sentiment-Analysis-Using-R.git

List of Figures

List of Tables

1 Introduction

The rapid growth of online forums and communities discussing various topics, including Machine Learning(ML), has led to an abundance of user-generated content in the form of text data. Analyzing this data can reveal valuable insights into user perceptions, opinions, and sentiments toward specific subjects. Sentiment analysis, or opinion mining, is a natural language processing technique used to determine the sentiment expressed in a piece of text, such as positive, negative, or neutral. This paper focuses on employing sentiment analysis techniques to understand the sentiment polarity of user comments related to the topics of Machine Learning, Data Science, and Artificial Intelligence.

1.1 Background and motivation

Machine Learning, Data Science, and Artificial Intelligence have become popular fields of study and research due to their vast applications in various industries. As a result, online communities have emerged where people discuss these topics, share knowledge, and express their opinions. Analyzing the sentiment of these users' comments can provide insights into the perception of the Data Science community, identify common issues or concerns, and uncover emerging trends. This understanding could be valuable for researchers, educators, and industry professionals who aim to improve their work based on community feedback.

1.2 Research questions

This study aims to address the following research questions:

1. What is the overall sentiment polarity (positive, negative, or neutral) of user comments related to Machine Learning, Data Science, and Artificial Intelligence?
2. How is the sentiment polarity distributed across different posts in the dataset?
3. Are there any correlations between post attributes (e.g., post length, engagement) and sentiment polarity?
4. Can any trends or patterns in sentiment polarity be identified over time or in relation to specific subtopics within the topics of Machine Learning, Data Science, or Artificial Intelligence?

1.3 Scope and limitations

The scope of this study is limited to the analysis of a dataset containing post _ids and comments of the users related to the topics. The results and interpretations are based solely on the data provided and may not be generalizable to other online communities or topics. The sentiment analysis model employed in this study is subject to the limitations inherent in natural languages processing techniques, such as context ambiguity and handling of sarcasm or irony. Additionally, the study does not account for demographic information or biases present in the dataset, which could potentially influence the sentiment analysis results.

2 Literature Review

2.1 Sentiment analysis in online communities

Over the past decade, sentiment analysis has become increasingly popular as a tool to mine user opinions and emotions from online platforms, such as social media, blogs, and forums (Pang and Lee 2008). Numerous studies have focused on the application of sentiment analysis in various domains, such as politics (Tumasjan et al. 2010), finance (**citeBollen?**), and consumer reviews (Dave, Kushal and Lawrence, Steve and Pennock, David M. 2003). These studies demonstrate the potential of sentiment analysis as a valuable technique for understanding user-generated content and informing decision-making processes.

2.2 Sentiment analysis techniques in R

R is a widely used programming language for statistical analysis and data visualization, with several libraries and packages available for natural language processing and sentiment analysis (R Core Team 2020). Some of the most popular sentiment analysis packages in R include tidytext (Silge and Robinson 2016), syuzhet (Jockers 2020), and sentiment (**citeRinker?**). These packages provide various algorithms and techniques for sentiment analysis, such as lexicon-based approaches, machine-learning models, and deep-learning methods. Researchers have employed these techniques to analyze sentiment in diverse contexts, ranging from Twitter data (Wijeratne 2017) to news articles (earney 2019)

2.3 Applications of sentiment analysis in Machine Learning research

Sentiment analysis has been employed in Machine Learning research to understand user perceptions, opinions, and trends related to the field. For example, Aliza (Sarlan, Aliza and Nadam, Chayanit and Basri, Shuib 2014) analyzed the sentiment of tweets related to Machine Learning conferences, revealing insights into the community's responses to specific presentations and events. In another study, Wang (Wnag 2019) investigated the sentiment of user comments on popular Machine Learning platforms, such as GitHub and Stack Overflow, to identify trends and patterns in the community's engagement with specific algorithms and techniques. These studies highlight the potential of sentiment analysis to uncover valuable insights into the Machine Learning community and inform future research directions.

The literature demonstrates the growing interest in sentiment analysis as a valuable tool for understanding user-generated content in various domains. R has emerged as a popular platform for sentiment analysis due to its extensive libraries and packages. However, there remains a need for more comprehensive studies that investigate the sentiment of user comments related to Machine Learning, particularly in the context of online communities where users share their knowledge and opinions on the topic. This study aims to address this gap by employing sentiment analysis techniques in R to analyze a dataset of post IDs and comments related to Machine Learning on the Reddit platform.

3 Data Collection and Preprocessing

This section describes the process of data collection, including web scraping and API requests, and the subsequent preprocessing steps applied to the dataset to prepare it for sentiment analysis.

3.1 Data Collection

The dataset used in this study was collected from Reddit, a popular online platform where users can share and discuss a wide range of topics. The data was obtained by performing web scraping and API requests on the topics of Machine Learning, Data Science, and Artificial Intelligence posts between 2013 to 2023.

The Reddit API (https://www.reddit.com/dev/api/) was utilized to access and retrieve relevant posts and associated comments from the specified subreddits (e.g., r/MachineLearning, r/datascience, r/artificial).

To perform the web scraping and API requests, the Python package PRAW (Python Reddit API Wrapper) (**citePRAW?**) was used, which provides a convenient interface to interact with the Reddit API. The data collection process involved:

1. Authenticating with the Reddit API using a registered application's credentials.
2. Querying the API for posts related to the specified topics (Machine Learning, Data Science, and Artificial Intelligence) within the targeted subreddits.
3. Combining the collected data into a structured dataset for further analysis.

Table 1: 10 observations from dataset of Reddit API

post_id	subreddit	category	score	created_year
gmy6p0	MachineLearning	News	447	2020
8h2wzn	MachineLearning	Project	366	2018
otbote	datascience	Discussion	372	2021
mn8r7f	MachineLearning	Research	438	2021
sxaiq8	MachineLearning	Discussion	347	2022
hnt2kk	datascience	Discussion	248	2020
v6sv06	datascience	Discussion	743	2022
8lm5f0	MachineLearning	Research	345	2018
n6fgjw	datascience	Career	447	2021
k77sxz	MachineLearning	Discussion	501	2020

Table 1 shows ten observations of the dataset, with 5 selected attributes of post_id, subreddit, category, score, and created_year. It contains 223,781 observations and 7 variables in total.

3.2 Data cleaning and preprocessing

After obtaining the raw dataset, several preprocessing steps were performed to clean and prepare the data for sentiment analysis. These steps include:

3.2.1 Basic cleaning

There are different methods we can use to condition the data, but this paper will stick to the basics and use gsub() and apply() functions to do the cleaning (Becker and WLLKS 1998).

First, we got rid of those pesky contractions by creating a little function that handles most scenarios using gsub(), and then applied that function across all comments. Second, all those special characters that muddy the text were removed with gsub() function and a simple regular expression. Lastly, to be consistent, we converted everything to lowercase with tolower() (**citetolower?**) function.

All the work was done on a separate R file called 02-data_preprocessing.R. It can be found in the 'script' folder.

3.2.2 Text Mining

Text mining and text analytics can be used interchangeably. The primary objective is to uncover significant data that may be concealed or unfamiliar beneath the surface. One of the techniques utilized in text mining is Natural Language Processing (NLP), which aims to unravel the intricacies in written language through various methods such as tokenization, clustering, entity extraction, and analyzing the relationships between words. Furthermore, NLP employs algorithms to detect patterns and quantify subjective data.

3.2.3 Tokenization

To unnest the tokens, we use the tidytext library (**citetidytext?**). From the tidy text framework, we broke the text into individual tokens and transform it into a tidy data structure. To do this, we used tidytext's unnest_tokens() function. unnest_tokens() requires at least two arguments: the output column name that will be created as the text is unnested into it ("word," in this case) and the input column that holds the current text (i.e. comments). We then took the Reddit dataset and pipe it into unnest_tokens() and then removed stop words. There are different lists to choose from, but here we used the lexicon called stop_words from the tidytext package.

After the tokenization, we used dplyr's anti_join() (Wickham et al. 2021) to remove stop words. Then we used distinct() to get rid of any duplicate records as well. Lastly, we examined the class and dimensions of our new tidy data structure:

```
[1] "data.frame"
```

```
[1] 2914188       8
```

reddit_comments_filtered is a data frame with 2914188 total words (not unique words) and 8 columns. The following is the snapshot:

Table 2: Tokenized Format

word	post_id	subreddit	score	years
deepmind	w6kj9y	MachineLearning	1732	22-23
deepmind	wiqjxv	MachineLearning	1339	22-23
deepmind	y89xqw	MachineLearning	946	22-23
deepmind	vfl57t	MachineLearning	515	22-23
deepmind	wcug1f	MachineLearning	469	22-23
deepmind	xpe6bi	datascience	418	22-23
deepmind	w44lkv	datascience	407	22-23
deepmind	xw6r5k	datascience	408	22-23
deepmind	zetvmd	MachineLearning	368	22-23
deepmind	xwfvlw	MachineLearning	364	22-23

Table 2 shows us the tokenized, unsummarized, tidy data structure.

4 Descriptive Statistics

During this section, we will be using creative graphs from the ggplot2 (Wickham 2016), circlize (**citecirclize?**), and yarrr (**citeyarrr?**) packages.

4.1 Shipshape: Word Count Per Post

The term "shipshape" is commonly used to indicate that everything is well-organized, neat, and tidy. In this context, an interesting perspective is presented on the orderly and organized data that demonstrates the lexical diversity, or the range of vocabulary, found within post comments over time. The pirate plot, a sophisticated technique for graphing a continuous dependent variable (such as word count) against a categorical independent variable (such as 'year'), is utilized to create a comprehensive and informative visual representation that incorporates both raw data points and statistical analysis.

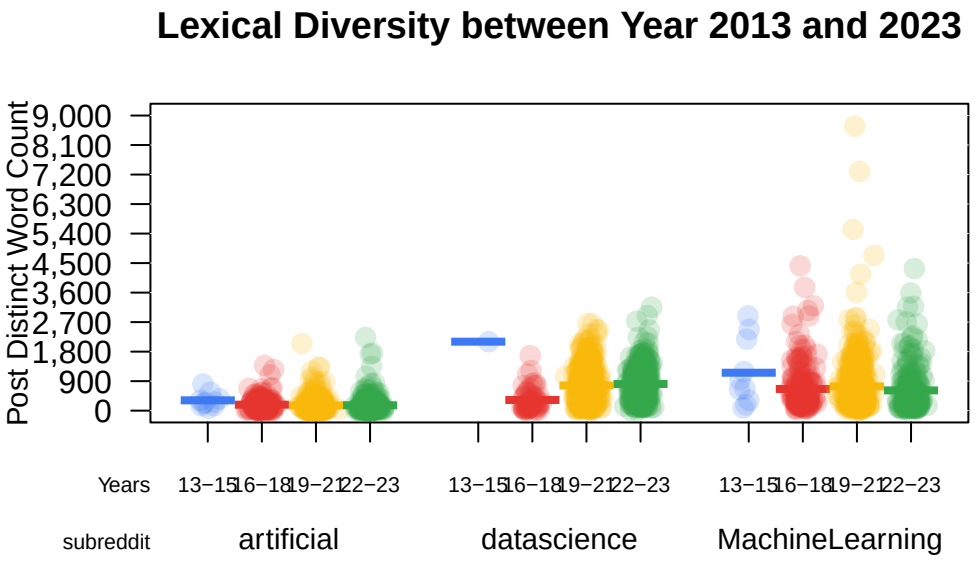

Figure 1: Lexical Diversity over the past 10 years, between '13 and '23

In this pirate plot displayed in figure Figure 1, each colored circle symbolizes a post. The dense clusters within each category indicate a considerable volume of posts in the dataset. Upon observation, it becomes evident that there were only a few posts during the period of '13-15 across all topics, whereas from '16-18 onwards, there was a surge in post activity. Additionally, it is worth noting that there is a slight upward trend in the number of distinct words per post in the Machine Learning topic. The solid horizontal line represents the mean word count for the corresponding years.

5 Assumption Checking and Methods

5.1 All Year Round: Post Count Per Year

Figure 2 depicts the total number of posts in the areas of 'Machine Learning', 'Artificial Intelligence', and 'Data Science' from 2019 to 2022. During this time, there were many advancements in artificial intelligence, including the development of ChatGPT and other important breakthroughs. Now that we are in 2023, the number of posts has already grown a lot compared to previous years. This suggests that people are more active in these topics than they were in the last 10 years.

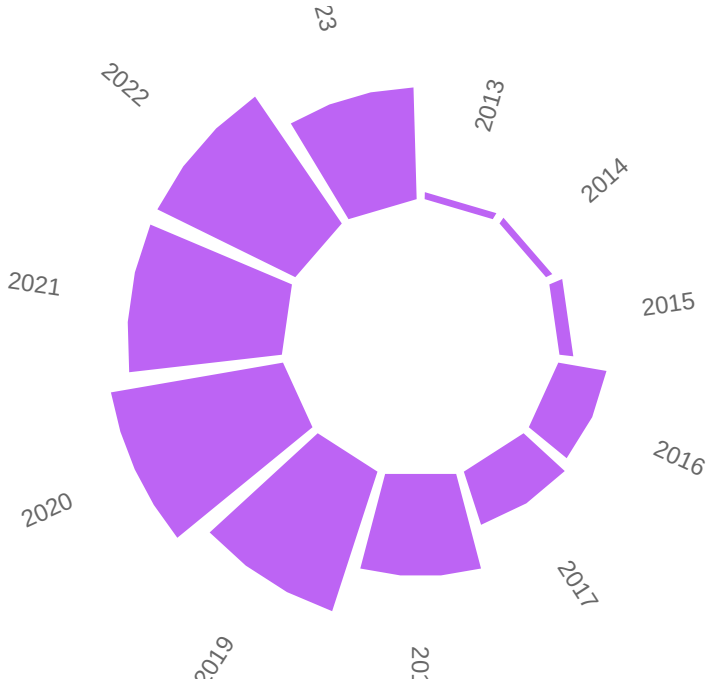

Figure 2: The total amount of posts on Reddit platform between Year '13 and '23

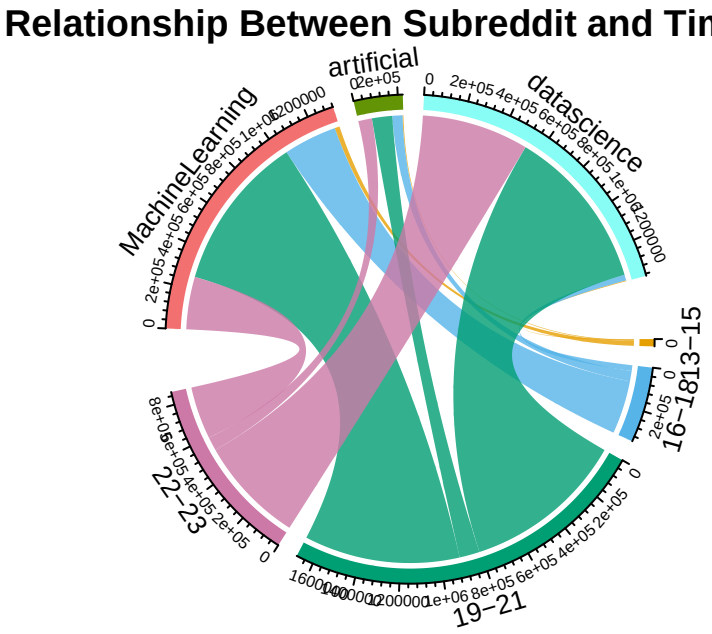

Figure 3: Relathiopship between Subreddit posts and timeline in semi-annual frames

5.2 Chords: Subreddit posts by Years

As shown above in Figure 3, the subject Machine Learning is expereienced significant growth between 2019 and 2021, and this trend has persisted into 2023.

5.3 Sentiment analysis model selection

5.3.1 Explore sentiment lexicons

The tidytext (**citetidy?**) package includes a dataset called sentiments (**citesentiments?**) which provides several distinct lexicons. These lexicons are dictionaries of words with an assigned sentiment category or value. Tidytext provides three general purposes lexicons:

- AFINN: assigns words with a score that runs between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment
- Bing: assigns words into positive and negative categories
- NRC: assigns words into one or more of the following ten categories: positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust

In order to examine the lexicons, we create a data frames.

5.3.1.1 Create sentiment datasets

Start off by creating Post sentiment datasets for each of the lexicons by performing an inner_join() on the get_sentiments() function. We pass the name of the lexicon for each call. For this paper, we use Bing for binary and NRC for categorical sentiments. Since words can appear in multiple categories in NRC, such as Negative/Fear or Positive/Joy, we will also create a subset without the positive and negative categories to use later on. Refer to 03-simulation.R

5.3.1.2 Mood changes over time

Since we are looking at sentiment from a polarirty perspective, we might want to see whether or not it changes over time. We use geom_smooth() (Wickham 2016) with loess method for a smooth curve and another geom_smooth() with method=lm for a linear smooth curve.

Figure 4 shows that the overall polarity trend over time is negative in both cases.

5.3.1.3 Sentiment in Machine Learning

Let's take a deeper look into the mood of a specific topic over time: Machine Learning

How would NRC model interpret the mood of posts under Machine Learning?

Although highly subjective, the results in Figure 5 appear to confirm that there is overwhelmingly positive and trust sentiment with Machine Learning. Let's go further in depth by looking at the sentiment categories with distinctive topics ('Career' and 'discussion') and see if they appear to be correlated.

Sub-NRC sentiments, Figure 6, show that anticipation is increasing under career category over time. Does this tell us that people are worrying about their jobs?

Sub-NRC sentiments, Figure 7, show "anticipation" is also increasing under "discussion' category. Here, interesting fact we can notice is the polarity between trust and anticipation is decreasing as we have witnessed above.

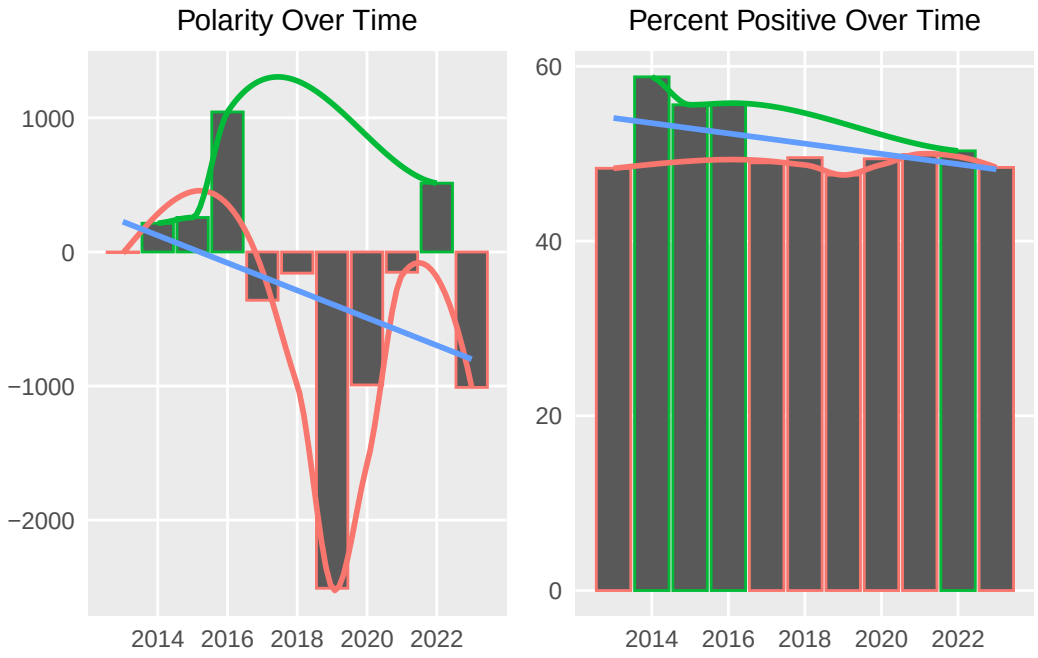

Figure 4: Sentiment ploarity over time and Percet postivie over time are descreasing

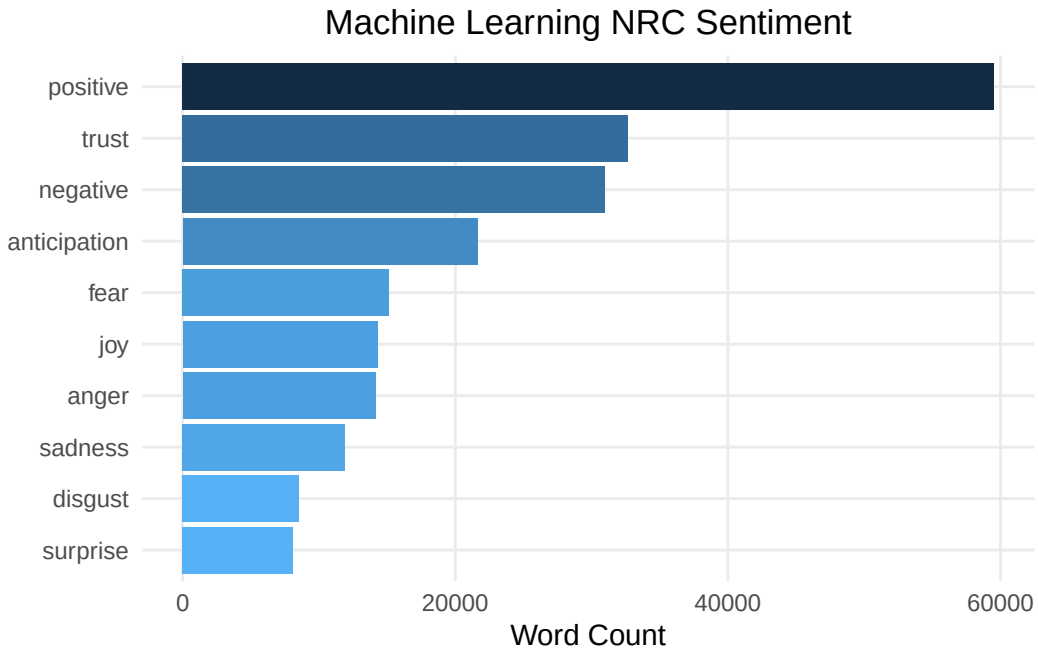

Figure 5: The interpretation of NRC model in mood of posts under Machine Learning

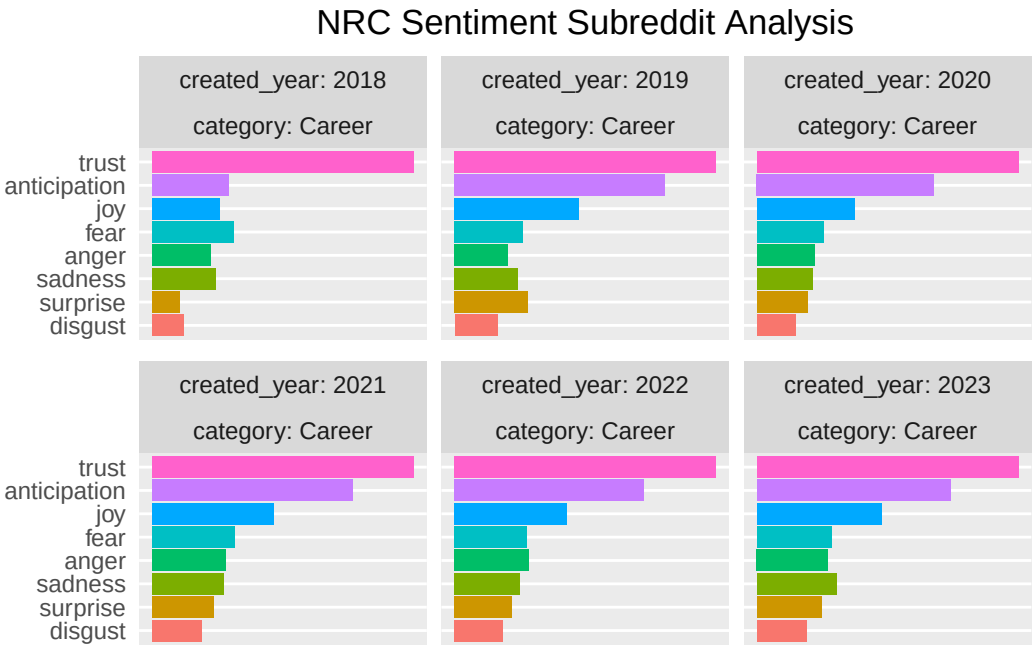

Figure 6: NRC sentiment showing the level of anticipation is increasing over time, under 'Career'

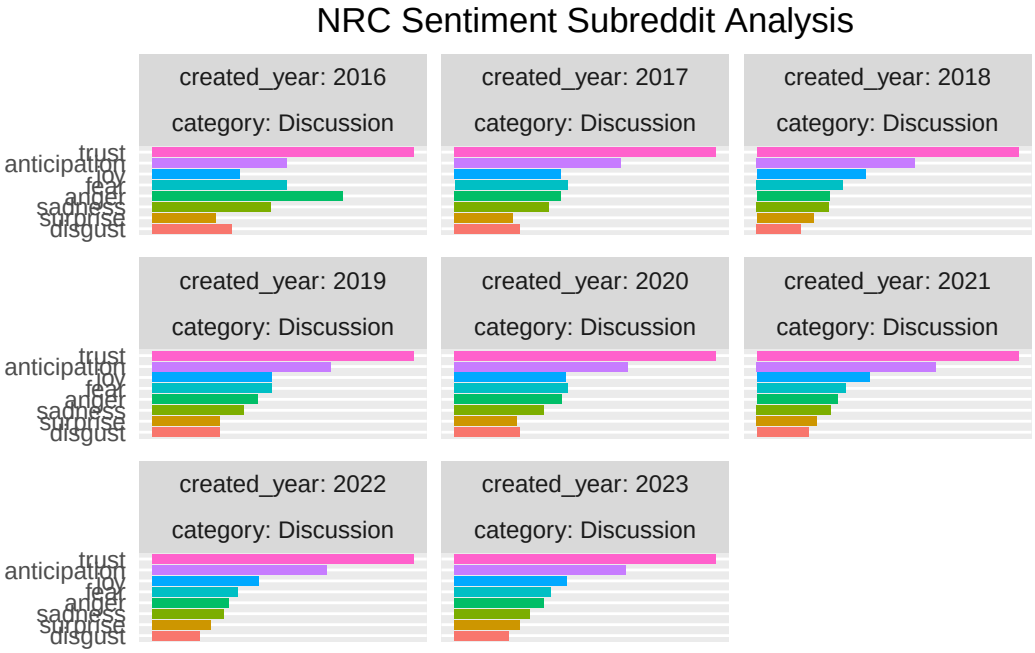

Figure 7: NRC sentiment showing the level of anticipation increasing over time, under 'Disscussion'

5.4 Model training and validation

5.4.1 World embeddings

Word embeddings are a way to represent text data as vectors of numbers based on a huge corpus of text, capturing semantic meaning from words' context. First, let's filter out words that are used only rarely in this data set and create a nested dataframe, with one row per post.

Next, we create a slide_windows() function, using the slide() function from the slider package (**citeslider?**). This new function identifies skipgram windows in order to calculate the skipgram probabilities, how often we find each word near each other word.

Now we can find all the skipgram windows, let's calculate how often words occurs on their own, and how often words occur together with other words. We do this using the point-wise mutual information (PMI).

In statistics, probability theory and information theory, pointwise mutual information (PMI), or point mutual information, is a measure of association. It compares the probability of two events occurring together to what this probability would be if the events were independent.

$$pmi(x;y) \equiv \log_2 \frac{p(x,y)}{p(x)p(y)} = \log_2 \frac{p(x|y)}{p(x)} = \log_2 \frac{p(y|x)}{p(y)}$$

It's the logarithm of the probability of finding two words together, normalized for the probability of finding each of the words alone. We use PMI to measure which words occur together more often than expected based on how often they occurred on their own. This step is the computationally expensive part of finding word embeddings. However, by using furrr package (**citefurrr?**), we can take advantage of parallel processing because identifying skipgram windows in one document is independent from all the other documents.

When PMI is high, the two words are associated with each other, i.e., likely to occur together. When PMI is low, the two words are not associated with each other, unlikely to occur together.

We then determine the word vectors from the PMI values using singular value decomposition (SVD). SVD is a method for dimensionality reduction via matrix factorization (**citeGolub?**) that works by taking our data and decomposing it onto special orthogonal axes.

In our application, we will use SVD to factor the PMI matrix into a set of smaller matrices containing the word embeddings with a size we get to choose. We will be using the widely_svd() function in widyr (**citewidyr?**), creating 100-dimensional word embeddings.

We now have our reddit word embeddings.

5.5 Exploring reddit word embeddings

Now that we have determined word embeddings for the data set of Reddit posts, let's explore them and discuss how they are used in modeling.

Each word can be represented as a numeric vector in the new set of 100-dimensional feature space. A single word is mapped to only one vector, therefore, all sense of a word are conflated in word embeddings. Because of this, word embeddings are limited for understanding lexical semantics.

In order to find out which words are close to each other, we created a simple function that will find the nearest words to any given example in using our newly created word embeddings (the code can be found under 03-simulation.R). This function takes the tidy word embeddings as input, along with a word/token as a string. Let's explore what words are closest to "slave" in the data set of reddit posts, as determined by our word embeddings.

```
# A tibble: 5,820 x 2
   item1           value
   <chr>           <dbl>
 1 machinelearning 1
 2 reddit          0.795
 3 remindmebot     0.777
 4 datascience     0.772
 5 comments        0.759
 6 compose         0.737
 7 message         0.726
 8 remind          0.714
 9 excludeme       0.712
10 totesmessenger  0.696
# i 5,810 more rows
```

Since we have found word embeddings via singular value decomposition, we can use these vectors to understand what principal components explain the most variation in the Reddit posts.

In linear algebra, the singular value decomposition (SVD) is a factroziation of a real or complex matrix. It generalizes the eigendecomposition of a square normal matrix with an orthonormal eigenbasis to any $m \times n$ matrix. It is related to the polar decomposition. Specificially, the singular value decomposition of an $m \times n$ complex matrix is factorization of the form:

$$M = U\Sigma V$$

, where U is an $m \times m$ complex unitary maxtrix, $ $ is an $ m $\times$n $ rectangular diagonal matrix with nonnegative real numbers on the diagonal, V is an $ n $\times$n $ complex unitary matrix, and V* is the conjugate transpose of V. Such decomposition always exists for any complex matrix. If M is real, then U and V can be guanranteed to be real orthogonal matrices; in such contexts, the SVD is often denoted:

$$U\Sigma V^T$$

The orthogonal axes that svd used to represent our data were chosen so that the first axis accounts for the most variance, the second axis accounts for the next most variance, and so on. We can now explore which and how much each original tokens contributed to each of the resulting principal components produced using SVD.

Reserved.

At this point we can use stm() from stm(**citestm?**) to implement an LDA model.

5.6 Evaluation metrics

place holder

6 Results

place holder

6.1 Sentiment polarity distribution

place holder

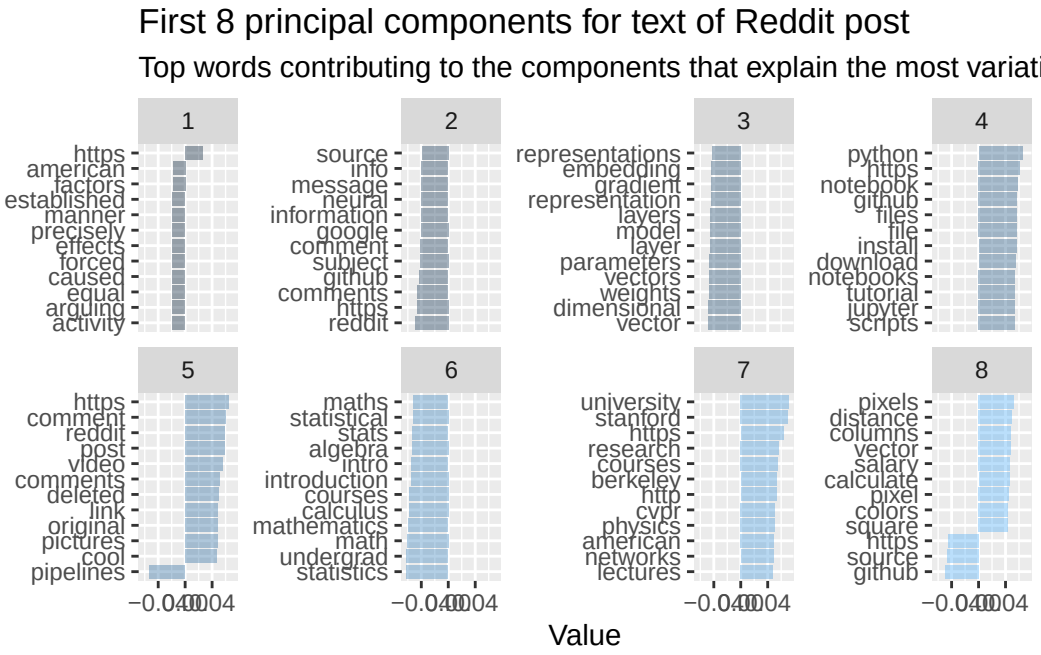

Figure 8: **?(caption)**

6.2 Correlation between post attributes and sentiment

place holder

6.3 Identification of trends and patterns

place holder

6.4 Comparison with previous studies

place holder

7 Discussion

place holder

7.1 Interpretation of results

place holder

7.2 Implications for Machine Learning community

place holder

7.3 Limitations of the study

place holder

7.4 Future research directions

place holder

8 Conclusion

place holder

References

Becker, Chambers, R. A., and A. R. WLLKS. 1998. *The New s Language*. https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/grep.

Dave, Kushal and Lawrence, Steve and Pennock, David M. 2003. "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews." In *Proceedings of the 12th International Conference on World Wide Web*, 519–28. WWW '03. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/775152.775226.

earney, Michael W. 2019. "Rtweet: Collecting and Analyzing Twitter Data." https://doi.org/http://dx.doi.org/10.21105/joss.01829.

Jockers, Matthew. 2020. "Syuzhet: Introduction to the Syuzhet Package." In. https://cran.r-project.org/web/packages/syuzhet/vignettes/syuzhet-vignette.html.

Pang, Bo, and Lillian Lee. 2008. "Opinion Mining and Sentiment Analysis" 2 (1–2): 1–135. https://doi.org/10.1561/1500000011.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Sarlan, Aliza and Nadam, Chayanit and Basri, Shuib. 2014. "Twitter Sentiment Analysis." In *Proceedings of the 6th International Conference on Information Technology and Multimedia*, 212–16. https://doi.org/10.1109/ICIMU.2014.7066632.

Silge, Julia, and David Robinson. 2016. "Tidytext: Text Mining and Analysis Using Tidy Data Principles in r." *Journal of Open Source Software* 1 (3): 37. https://doi.org/10.21105/joss.00037.

Tumasjan, Andranik, Timm Oliver Sprenger, Philipp G. Sandner, and Isabell M. Welpe. 2010. "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment." *Proceedings of the International AAAI Conference on Web and Social Media*.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2021. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.

Wijeratne, Balasuriya, s. 2017. "EmojiNet: An Open Service and API for Emoji Sense Discovery." https://doi.org/https://doi.org/10.48550/arXiv.1707.04652.

Wnag, Chen, Y. 2019. "A Comparative Study of Machine Learning Algorithms and Their Applications." https://doi.org/https://doi.org/10.1109/ICRITO.2018.8748793.