288R Project Progress Report - Discovering Topics in Dialectic Behavioral Therapy through LLM

Project Group #: 04

Authors: Jesse Parent, Nathaniel Do

Emails: jeparent@ucsd.edu, ntd002@ucsd.edu

Background

Our project is attempting to identify core mental health topics within Reddit posts so that we can recommend helpful resources. Many people search Reddit for advice through their struggles and our work will assist their attempts to fight their anger, anxiety, depression, etc.

Dataset

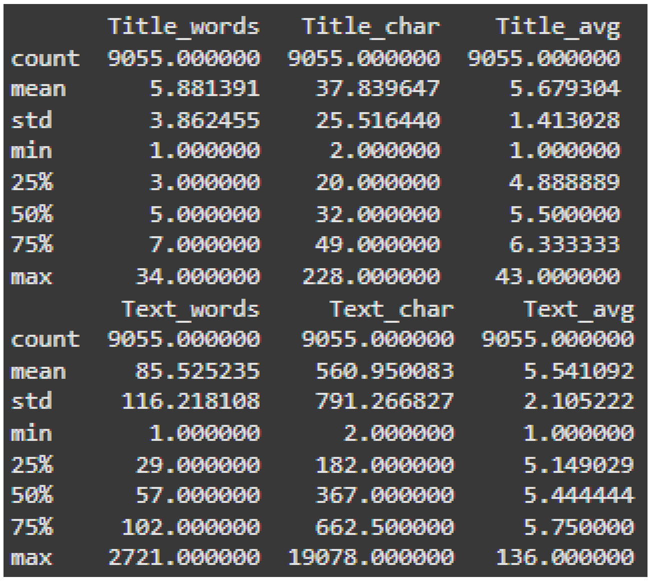
The dataset was created by us by scraping different subreddits using Async PRAW, a Python Reddit API Wrapper. It includes the top posts from different subreddits regarding different mental topics. Here is the link to our GitHub page with the CSV dataset, the Jupyter Notebook detailing how we scraped the data, and which subreddits are included in it (https://github.com/ntd002/DSC-288-Capstone/tree/a9ee75381e2ac9e3343d0305855ddb4c87d6aed3/Milestone%202%3A%201st%20Progress%20Report/Step%201%3A%20Scraping%20Reddit).

Data Pipeline

We have multiple data sources coming from across different subreddits. By making separate API calls, we have merged the posts into a pandas dataframe, saving the raw data to a CSV. The raw CSV was then preprocessed, removing null values and applying text cleaning methods summarized in the "Progress Report" section. We decided to keep separate columns of the unprocessed and preprocessed data so we can observe and input into models if we so desired. As we are dealing with text data and not numerical data, we did not need to normalize any numbers.

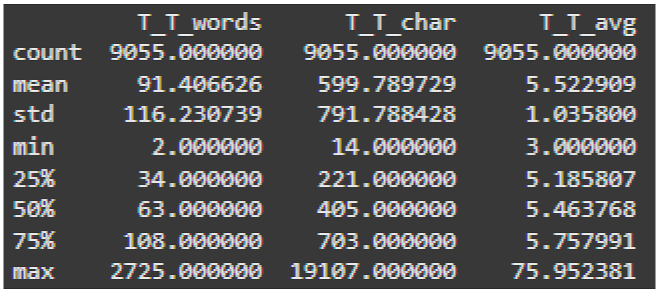
EDA Description

By performing univariate EDA on our processed titles and text of our posts, we see that while the titles average around 5.88 words per post, the text averages 85.53. The amount of characters follows this trend.



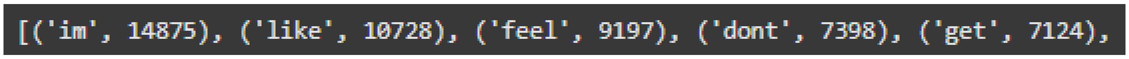
```
       Title_words   Title_char    Title_avg
count  9055.000000  9055.000000  9055.000000
mean      5.881391    37.839647     5.679304
std       3.862455    25.516440     1.413028
min       1.000000     2.000000     1.000000
25%       3.000000    20.000000     4.888889
50%       5.000000    32.000000     5.500000
75%       7.000000    49.000000     6.333333
max      34.000000   228.000000    43.000000
        Text_words     Text_char     Text_avg
count  9055.000000   9055.000000  9055.000000
mean     85.525235    560.950083     5.541092
std     116.218108    791.266827     2.105222
min       1.000000      2.000000     1.000000
25%      29.000000    182.000000     5.149029
50%      57.000000    367.000000     5.444444
75%     102.000000    662.500000     5.750000
max    2721.000000  19078.000000   136.000000
```

The average word length however is higher in titles at 5.68 as opposed to the text at 5.54. Both title and text were combined as well.



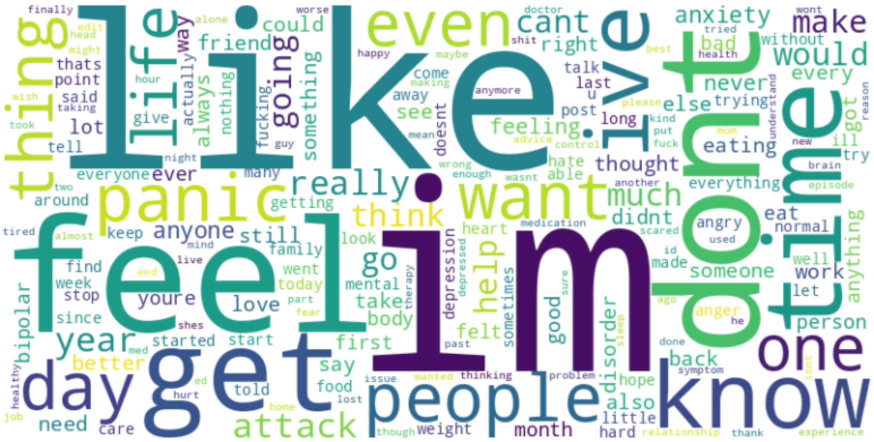
```
         T_T_words      T_T_char      T_T_avg
count  9055.000000   9055.000000  9055.000000
mean     91.406626    599.789729     5.522909
std     116.230739    791.788428     1.035800
min       2.000000     14.000000     3.000000
25%      34.000000    221.000000     5.185807
50%      63.000000    405.000000     5.463768
75%     108.000000    703.000000     5.757991
max    2725.000000  19107.000000    75.952381
```

Using this combined information, we also found the most common words.



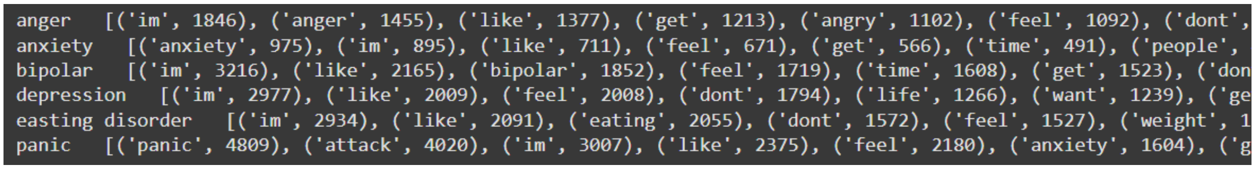
```
[('im', 14875), ('like', 10728), ('feel', 9197), ('dont', 7398), ('get', 7124),
```

While “I” and “like” are stop words which should have been cleaned during preprocessing, “im” and “like” are not found in the stop word library we used from NLTK. We were also able to produce a word cloud for visualization.



Feature Engineering

Diving deeper into the EDA, we can draw the most common words per topic.



```
anger   [('im', 1846), ('anger', 1455), ('like', 1377), ('get', 1213), ('angry', 1102), ('feel', 1092), ('dont',
anxiety   [('anxiety', 975), ('im', 895), ('like', 711), ('feel', 671), ('get', 566), ('time', 491), ('people',
bipolar   [('im', 3216), ('like', 2165), ('bipolar', 1852), ('feel', 1719), ('time', 1608), ('get', 1523), ('don
depression   [('im', 2977), ('like', 2009), ('feel', 2008), ('dont', 1794), ('life', 1266), ('want', 1239), ('ge
easting disorder   [('im', 2934), ('like', 2091), ('eating', 2055), ('dont', 1572), ('feel', 1527), ('weight', 1
panic   [('panic', 4809), ('attack', 4020), ('im', 3007), ('like', 2375), ('feel', 2180), ('anxiety', 1604), ('g
```

We observe common words we've seen before like "im," "like," and "dont," but we also can see some unique words. Each of the topics has their own name as a common name ("anger" for anger, "panic" for panic). We can assume that if a post has these words, it's most likely the associated topic. But some unique words are shared because many topics have shared symptoms. Panic has "anxiety" as its 6th most common word. This will make it interesting when trying to predict the topics between panic and anxiety. We may do some feature engineering to remove the common words that link the topics ("im," "like," "feel," etc.), so the models can focus on the unique words like "weight" or "heart." Potentially we can check the accuracy after removing the top 5 most shared words and then increase that to top 10 and so on until we reach a desired accuracy.

Models Used

As per our abstract, we were planning on using:

- Sentiment analysis using pre-trained models like BERT or GPT.

- Topic modeling via Latent Dirichlet Allocation (LDA) or non-negative matrix factorization (NMF).
- Classification using SVM, logistic regression, or fine-tuning transformers.

The classification models have been used by us during our past quarter while the former two have been mentioned or new to us. When the data was initially preprocessed, we ran the data through a multinomial Naive Bayes classifier to check for any errors in our preprocessing steps.

Progress Report

We constructed our dataset by utilizing Reddit's API to scrape the top posts within a number of subreddits. Each post was assigned a label depending on the topic of the subreddit (anger, depression, anxiety, etc.). With this raw dataset, we performed some preprocessing on the text data. We normalized the data by converting to lower case and removed numbers, special characters, and emojis. We delved into advanced preprocessing by tokenizing, removing stop words, and lemmatization. We tested the data with a multinomial Naive Bayes classifier which produced accuracies ranging from 62-68% (not ideal but satisfactory for our first foray).

Team Member Contribution

Due to the business of professional and personal lives, we have been trading off "leading" the project every so often. During the first two weeks, Jesse led in connecting with Dr. Wiebel and his team from the HXI lab at UCSD to learn more about this topic and how to approach it. In the recent weeks, Nathaniel applied that knowledge to scrape Reddit as well as preprocess the text data. Together we make major decisions such as modeling choices as well as writing text documents.

Risks

The three main challenges we set forth in our abstract were bias generated in an online forum, privacy of scraping data from an online forum, and investigating in a relatively unknown domain. Even though there is bias with the input as we are drawing data from Reddit posts, the bias should be mitigated since we will be conducting analysis on more Reddit posts. If we were instead testing on Facebook or tumblr posts, then the bias would have an impact. But staying within the same domain/platform should keep the bias lower. As far as privacy is concerned, Reddit has made it clear that "anyone who has access to the internet can see public posts, comments, usernames, profiles, karma scores, upvote/downvote ratios, and related metadata (collectively, "public content")." There should not be a legal issue regarding the privacy of these posts. Finally, the unknown domain is rather exciting as we will have to chart our own path forward. We are still in contact with Dr. Weibel so we can bounce ideas and methods off each other.

References

Stop Words in NLTK: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
Sentiment Analysis (BERT): https://www.geeksforgeeks.org/sentiment-classification-using-bert/
Sentiment Analysis (GPT): https://medium.com/@financial_python/use-chatgpt-api-for-sentiment-analysis-in-python-5a152ddb3238
Topic Modeling (LDA): https://www.geeksforgeeks.org/topic-modeling-using-latent-dirichlet-allocation-lda/
Topic Modeling (LDA and NMF): https://scikit-learn.org/stable/auto_examples/applications/plot_topics_extraction_with_nmf_lda.html
Classification Modeling: https://www.geeksforgeeks.org/comprehensive-guide-to-classification-models-in-scikit-learn/
Multinomial Naive Bayes: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
Reddit's Content Policy: https://support.reddithelp.com/hc/en-us/articles/26410290525844-Public-Content-Policy