Mental health support chatbot using GPT-2

Capstone Project 2

Yuliya Selevich

Introduction

Mental and physical health are equally important components of overall health. Mental health plays a huge role in general well-being and can affect physical health. Poor mental health can lead to poor physical health or harmful behaviors. An AI-based chatbot can be used as one of the primary care tools to improve general well-being. It can also benefit people who are hesitant to seek professional help due to anxiety caused by stigma and prejudice around mental illness. The goal of this project is to build a model that would take a question from a user as input and that would show an ability to give a sensible, relevant to the topic and human-like answer, as well as show strong grammar and spelling.

Data

The data was scraped from Reddit's r/mentalhealth and r/mentalillness subreddits using PSAW (Python Pushshift API Wrapper). Both posts and comments were scraped separately using the 'search_submissions' and 'search_comments' methods. Data from two subreddits were merged and stored in two Pandas DataFrames for submissions and comments. Submissions DataFrame contained the following columns:

- Id – Every submission has its unique identification number.
- Title – The title of a submission (often a question).
- Selftext – The author of a submission expands on the content of the title.
- Created – The date when a submission was created.

The DataFrame with comments also has 'id' and 'created' columns, as well as the following columns:

- Link_id – id that matches every submission with all the comments that were made on it.
- Body – The context of the comment that's been made.

Submissions DataFrame didn't have any duplicate ids and contained 415290 rows. Comments DataFrame had 77 duplicate comments that were consequently removed. The total number of rows in this DataFrame was 1556846.

Data Cleaning and Processing

The following consecutive steps were taken during the cleaning and processing of the data:

1) The 'created' column was removed as it was redundant.
2) All text data were converted into lowercase to decrease the noise and ambiguity in the data.
3) All the submissions and comments with missing values were removed.
4) There were a lot of comments that contained values indicating that the comment has been [removed] or [deleted]. Even though these values weren't identified as NaN by Pandas isna() function, they had to be removed as well.
5) All duplicate comments, which included a lot of automatic responses from a Reddit bot, were removed.
6) Comments that had the same 'link_id' were merged as they belonged to the same submission.
7) Later submissions and comments were merged into a single DataFrame on their corresponding ids.
8) Since I was interested in submissions that contained a question, I analyzed the submission titles to determine whether they start with one of the most common interrogative words. Submission titles that didn't start with one of these words were removed.
9) I also converted emojis to a word format to preserve more information and emotional coloring.
10) Some additional steps were taken to prepare the data for further analysis including tokenization, lemmatization, removal of stopwords, punctuation, digits and links, contractions expansion, and removal of extra space between tokens.

Exploratory Data Analysis

First I checked the most popular words in titles and comments using CountVectorizer() and counted the number of occurrences of every word in the text. Note that all stopwords were removed before this step. The distribution of the top 10 words in both titles and comments can be seen below.

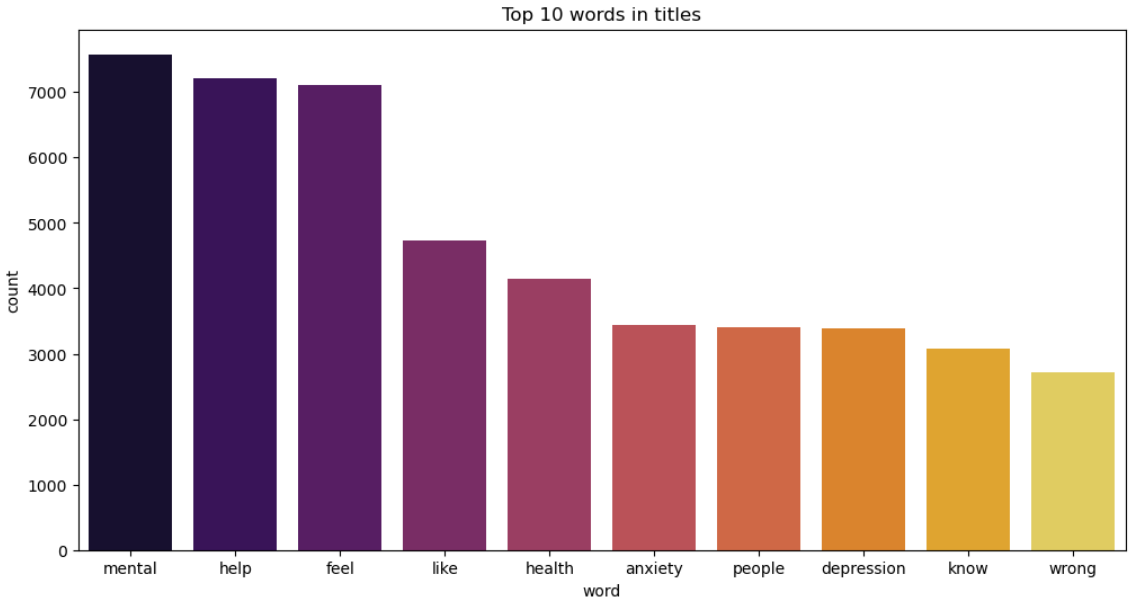

Figure 1 Top 10 words in titles

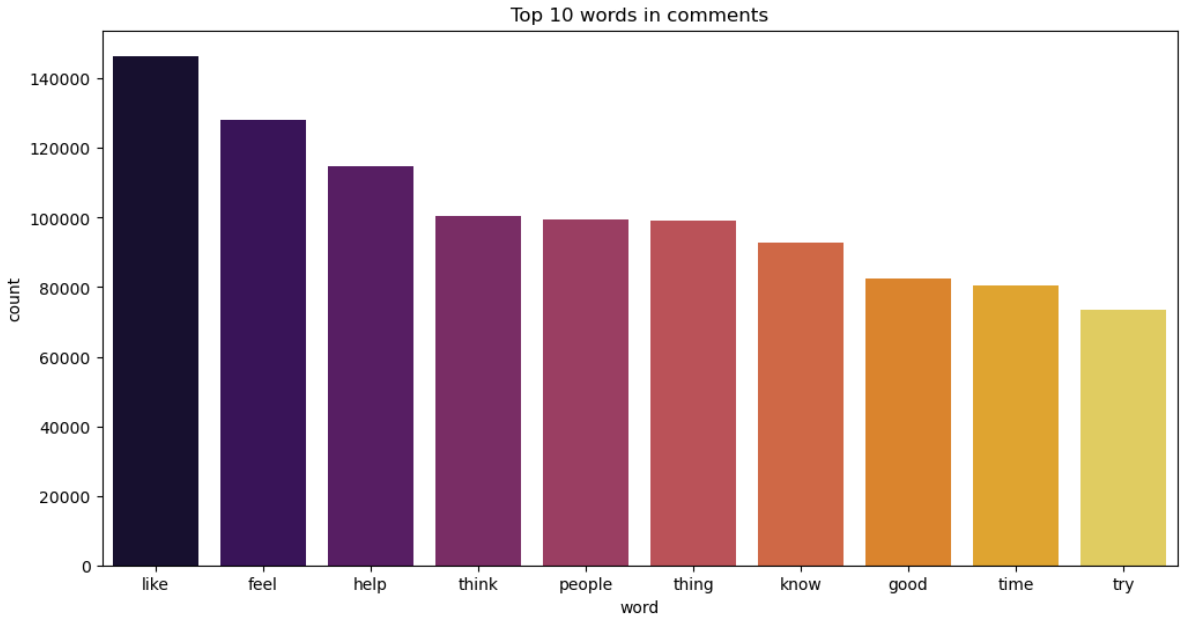

Figure 2 Top 10 words in comments

Just by visual analysis, there is a noticeable difference between titles and comments. Most of the top 10 words in titles (Fig.1) have a rather negative connotation, whereas the top 10 words in the comments (Fig. 2) have a neutral or positive connotation. Next, I determined the polarity of every title and comment using TextBlob. Polarity values lie in the range [-1, 1] where 1 means a positive statement, and -1 means a negative statement. As you can see in Figure 3 majority of sentences for both titles and comments have neutral polarity meaning that sentences have a neutral tone to them and don't seem to be emotionally expressive. The polarity of comments seems to have a normal distribution with the majority of sentences concentrated within the [-0.5, 0.5] range and lightly skewed towards a positive polarity indicating that the comments tend to be more supportive and encouraging. Although polarity is largely neutral in the titles, there are significantly more outliers and more posts seem to have a negative rather than positive polarity compared to those in the comments.

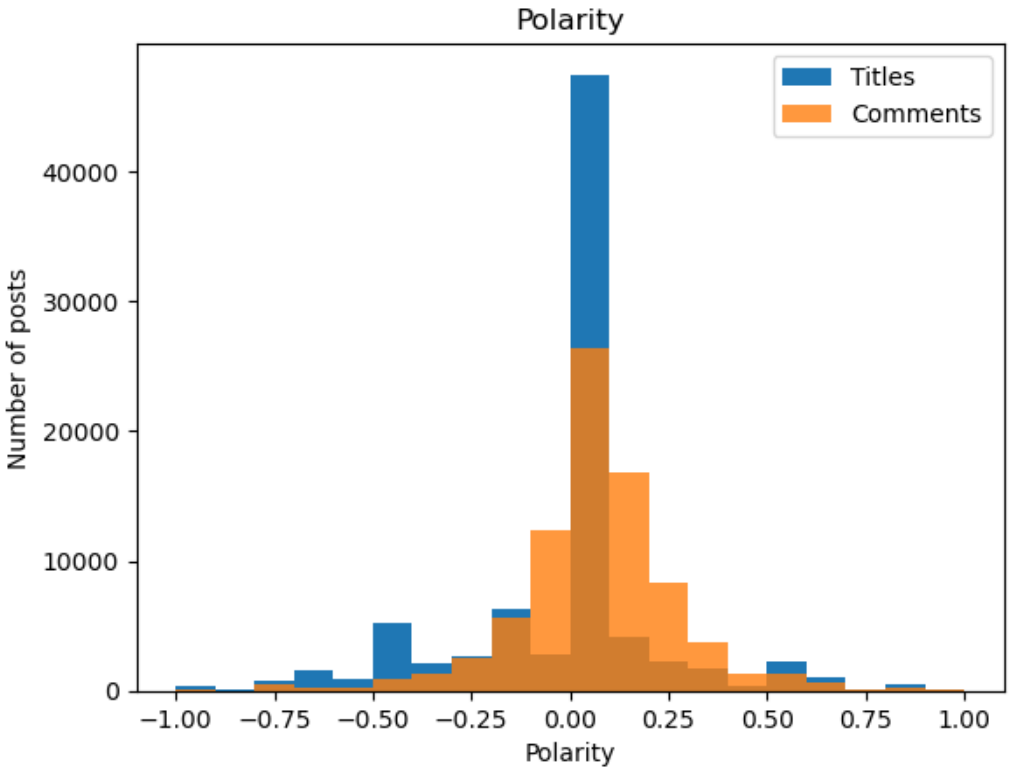

Figure 3 Polarity of titles and comments

Next, I decided to check the top 10 words in the titles and comments with the highest and the lowest polarity. As we can see if Figure 4 and Figure 5, there is an obvious difference in the emotional coloring of words where the top ten words in titles with the highest polarity include

mostly positively connoted words including ‘perfect’, ‘great’, ‘good’ whereas words in titles with the lowest polarity are largely negative including ‘insane’, ‘horrible’, ‘terrible’, ‘awful’, ‘miserable’ and ‘evil’.

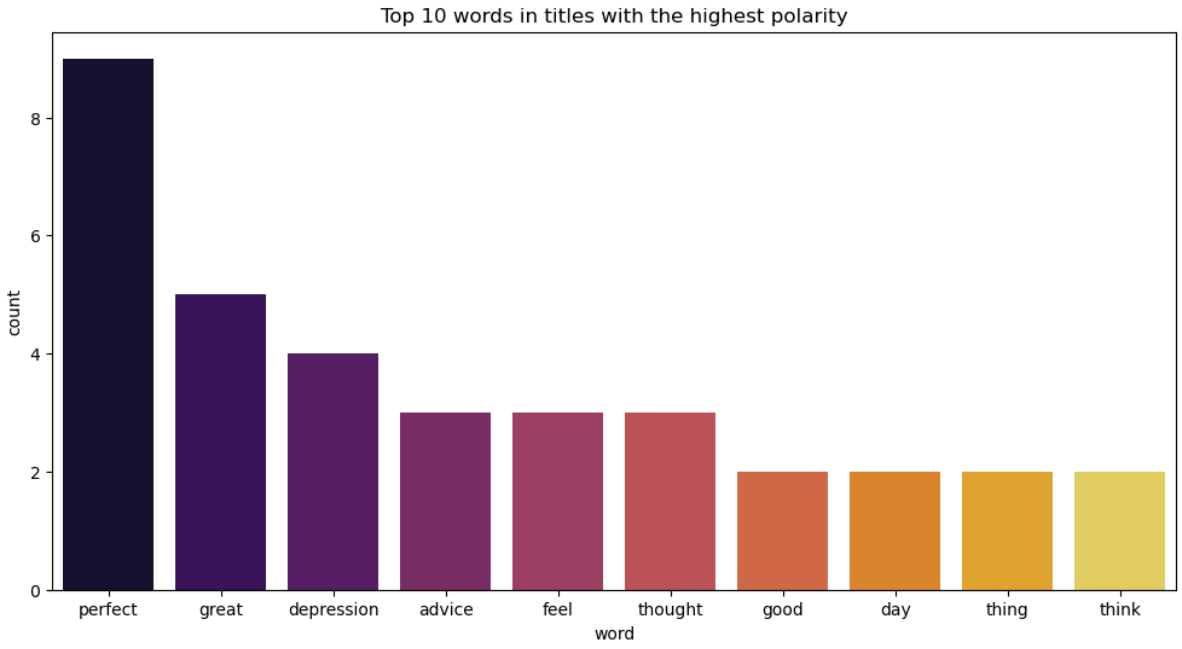

Figure 4 Top 10 words in titles with the highest polarity

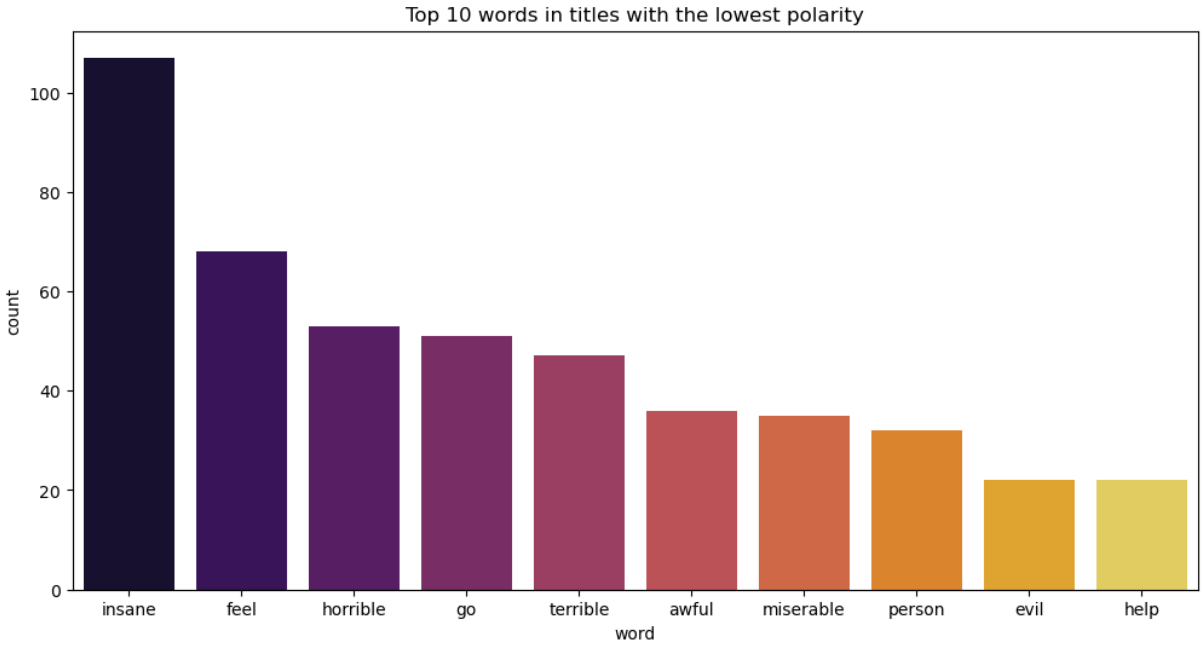

Figure 5 Top 10 words in titles with the lowest polarity

We can see a similar pattern in the comments (Fig. 6, Fig. 7). The top words with the highest polarity include the words ‘thank’, ‘happy’, and ‘great’. The top words with the lowest polarity include the words ‘awful’, ‘terrible’, and ‘horrible’.

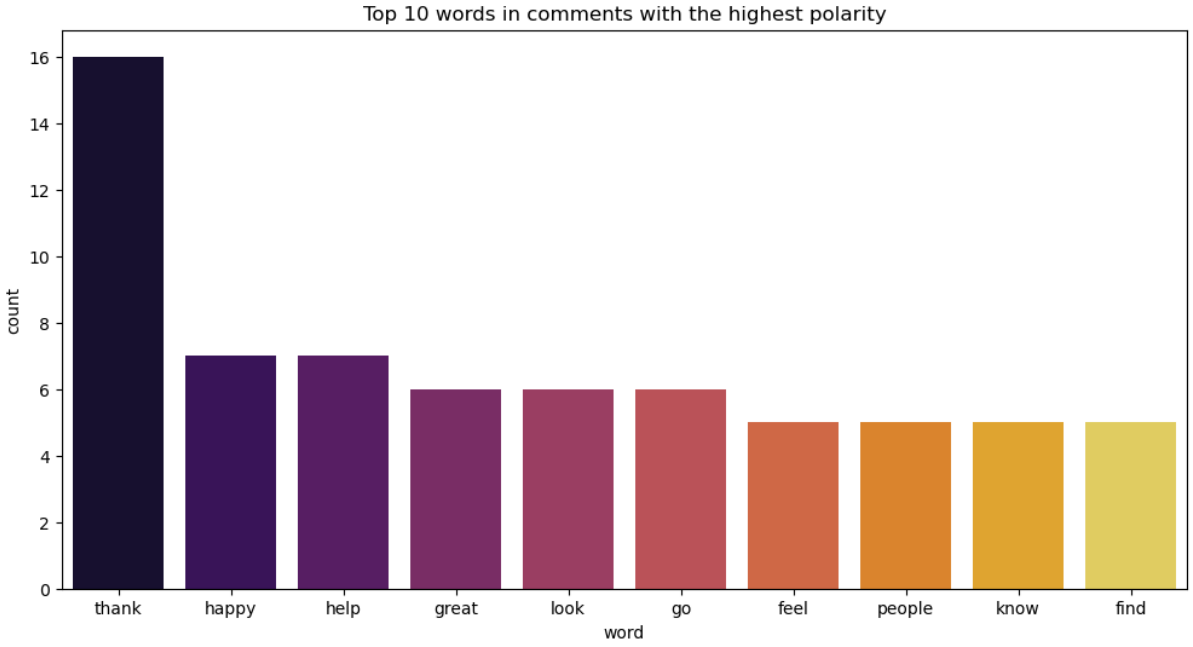

Figure 6 Top 10 words in comments with the highest polarity

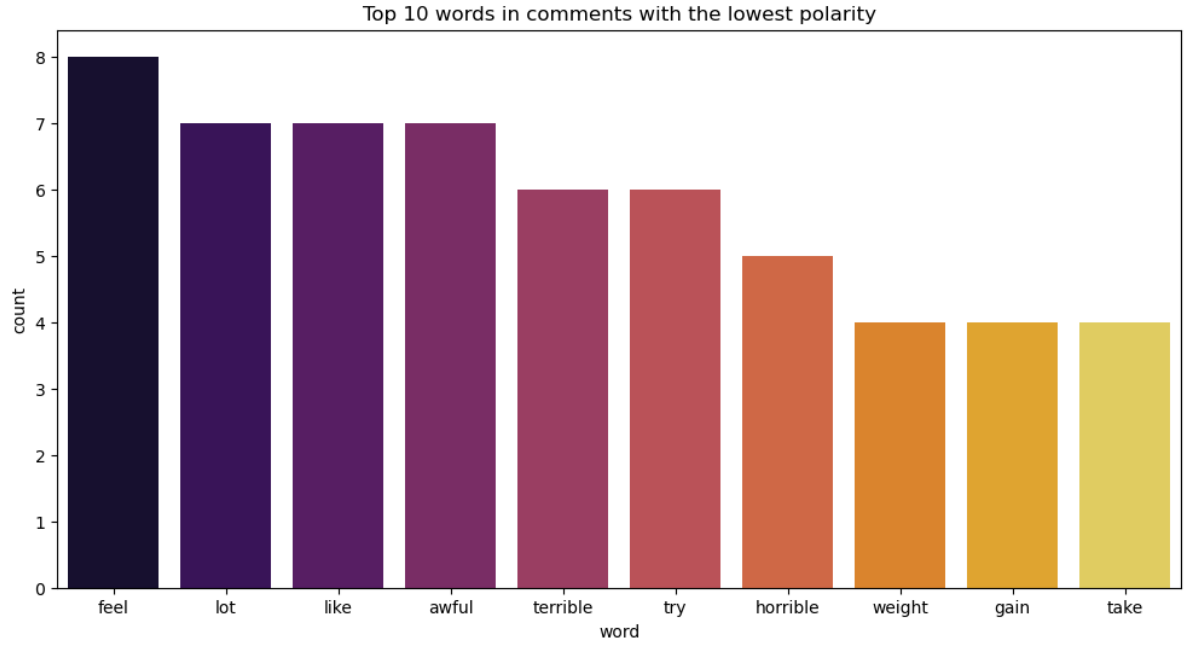

Figure 7 Top 10 words in comments with the lowest polarity

Next, I calculated the subjectivity of the titles and comments using TextBlob. Subjectivity quantifies the amount of personal opinion and factual information contained in the text and can range from 0 (objective or factual information) to 1 (subjective information). As shown in Figure 8, the subjectivity of titles seems to be randomly distributed with the vast majority of scores being equal to 0. The subjectivity of comments seems to be, on average, higher than the titles. This may be related to the fact that most of the titles are questions with most questions containing information that the author considers to be factual, whereas comments contain opinions and rather subjective statements.

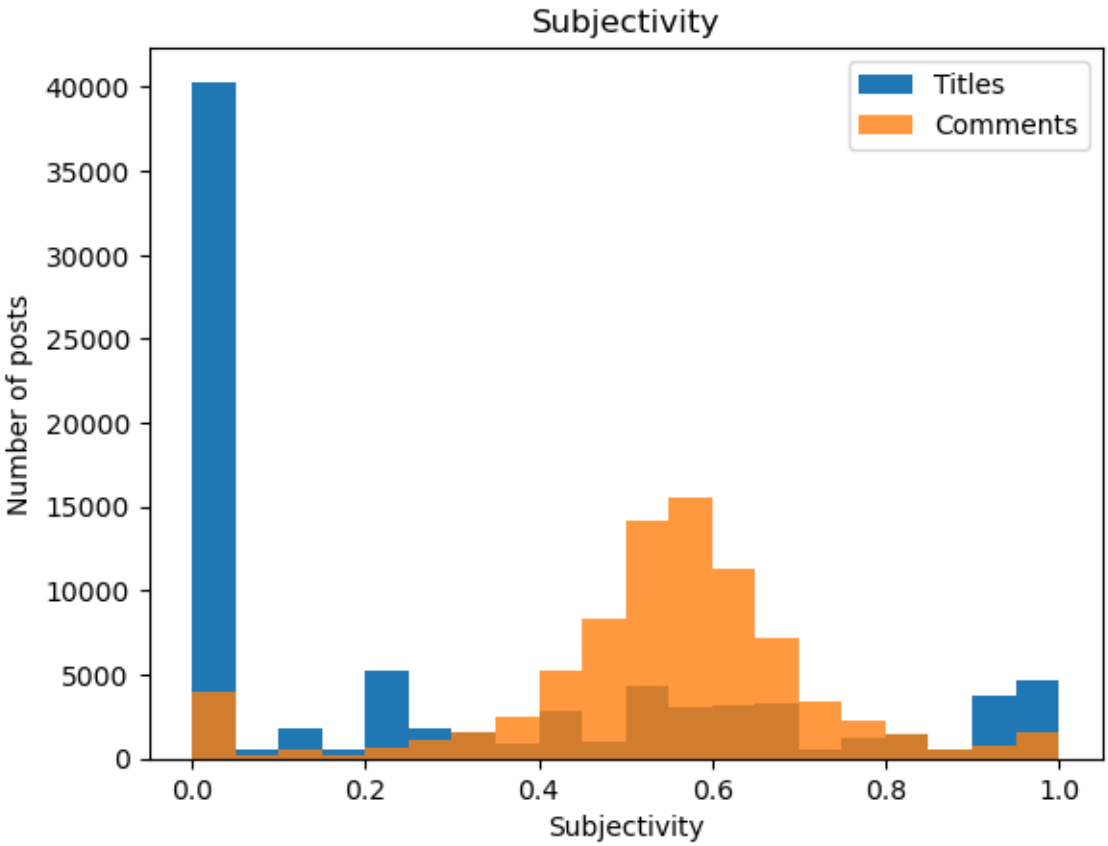

Figure 8 Subjectivity of titles and comments

Next, I calculated the readability level using the Flesch Reading Ease score. As you can see in Figure 9, readability scores are high for titles, and very low for comments. This may be related to the fact that most of the titles contain one short easily interpretable question, whereas comments are much longer and may have been comprised of multiple comments that belonged to the same submission.

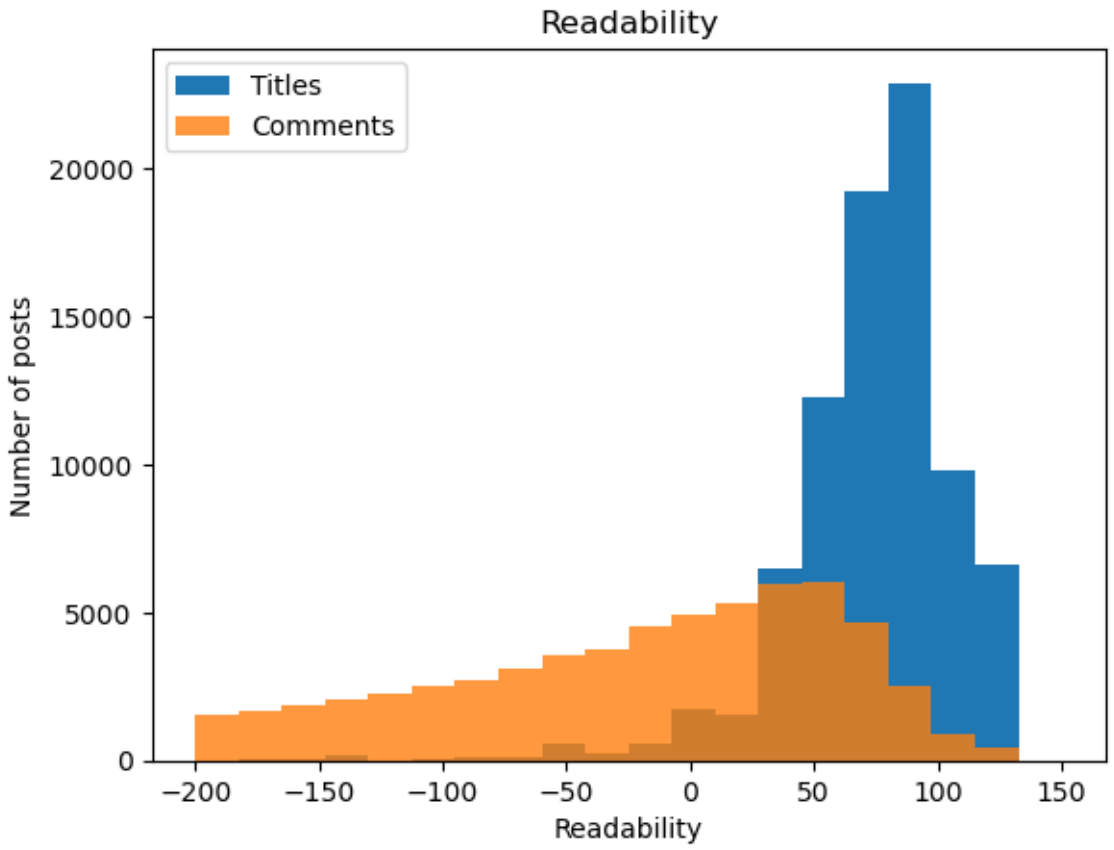

Figure 9 Readability scores for titles and comments

To investigate the topics further, I used gensim's LDA (Latent Dirichlet Allocation) model. The model can estimate and infer topic distribution in the text. To determine the optimal number of topics I build several LDA models with various numbers of topics and calculated their coherence scores. Coherence scores are used to measure how interpretable the topics are. The models for both titles and comments had very low coherence scores. Coherence scores for titles are much lower than those for comments (0.12 and 0.3 accordingly). Using gensim's filter_extremes() function I filtered out tokens that occur too often or too rarely to reduce the noise in the text. However, this process as well as hyperparameter tuning didn't improve coherence scores. This may be related to the fact that LDA models don't perform well on short text. Overall the model and simple eyeball analysis couldn't reveal specific topics that would have been much different from the others. One way or another, all sentences and tokens are united under one topic which is mental health.

Modeling

I used the GPT-2 pre-trained model and gpt-2-simple package for this project. GPT-2 is an unsupervised deep learning transformer-based language model created by OpenAI back in February 2019 for the single purpose of predicting the next word(s) in a sentence. GPT-2-simple wraps OpenAI's fine-tuning code in a functional interface and adds many utilities for model management and generation control. The so-called "medium" 355M model with the size of 1.5GB on disk has been selected. The inputs for a model are a sequence of tokens, and the outputs are the probability of the next token in the sequence, with these probabilities serving as weights for the AI to pick the next token in the sequence. Fine-tuning has been performed using finetune() method using 2100 steps and printed output samples every 500 steps. The fine-tuning took 39 hours and 28 minutes in total. The model seems to have converged relatively quickly around step 1200 where the loss function stopped a rapid decline.

Results

To generate the results a generate() method was used. Important parameters that were considered and tested out:

- Temperature – float value controlling randomness in Boltzmann distribution. Lower temperature results in fewer random completions. As the temperature approaches zero, the model will become deterministic and repetitive. Higher temperature results in more random output. Optimal values for this parameter are in the range of 0.7 to 1.
- Top_k – limits the generated guesses to the top k guesses. Integer value controlling diversity. 1 means only 1 word is considered for each step (token), resulting in deterministic completions, while 40 means 40 words are considered at each step. 0 (default) is a special setting meaning no restrictions. 40 generally is a good value.

Even though the two parameters above were supposed to control randomness and diversity, the model generated seemingly random output regardless of the values of the 'temperature' and 'top_k' parameters. The output could be nonsensical or repetitive while using high and top_k temperature values, or on the contrary, the output could be sensible with lower values which can seem counterintuitive.

Examples of a good (sensible) model's outputs:

1) 'The first steps are to accept the problem and then solve it. If you're having trouble, you can try talking about it with a mental health professional'.
2) 'I have trouble sleeping because of my depression. I get so tired that I don't even get up to leave the house.'
3) 'The main thing you want to do is take care of yourself and your health.'
4) 'I'm sorry you have to go through this. There's a lot of advice and information out there about how to deal with depression, what to do and what not to do. If you're not feeling good, you should go to a doctor for a checkup. They may be able to help you. Best of luck!'

Examples of a bad (nonsensical) model's outputs:

1) 'It just so happens my mom is also his grand-nephew.'
2) '…die peacefully due to an accident.'
3) 'I'm 23 and a self-taught neurosurgeon.'
4) 'My mental illnesses helped me a lot in a way that was very uncomfortable and painful.'
5) 'Just because you can't do something doesn't mean you can't do it: and that's okay.'

Conclusion

I used GPT-2 model to examine its efficacy in giving mental health advice. Most of the time the output generated by the model didn't make much sense. There are a lot of cases where the same words or even sentences repeat over and over again. Or a lot of times the output is grammatically correct but convoluted and difficult to interpret. Therefore improving a model and finding a way to filter out bad output before model deployment is important (detecting duplicates in the text, using a 'bleu' score). However, it is worth mentioning that the model was able to understand the overall theme, and most of the model's outputs related one way or another to mental health issues.

Taking into consideration the randomness in the model's output, the following steps may be taken to improve a model's performance:

- Finetune a GPT-3 model which showed improved performance compared to a GPT-2 model especially when given tasks in a specialized area or topic.

- Finetune the model with more data and steps.
- Perform more text cleaning keeping in mind its effect on the model’s generalization.