Dataset: ChatGPT Reddit

https://www.kaggle.com/datasets/armitaraz/chatgpt-reddit

Objectives

Try to find out the most often words while referring to chatgpt.

Try to figure out Reddit user's attitudes about chatgpt.

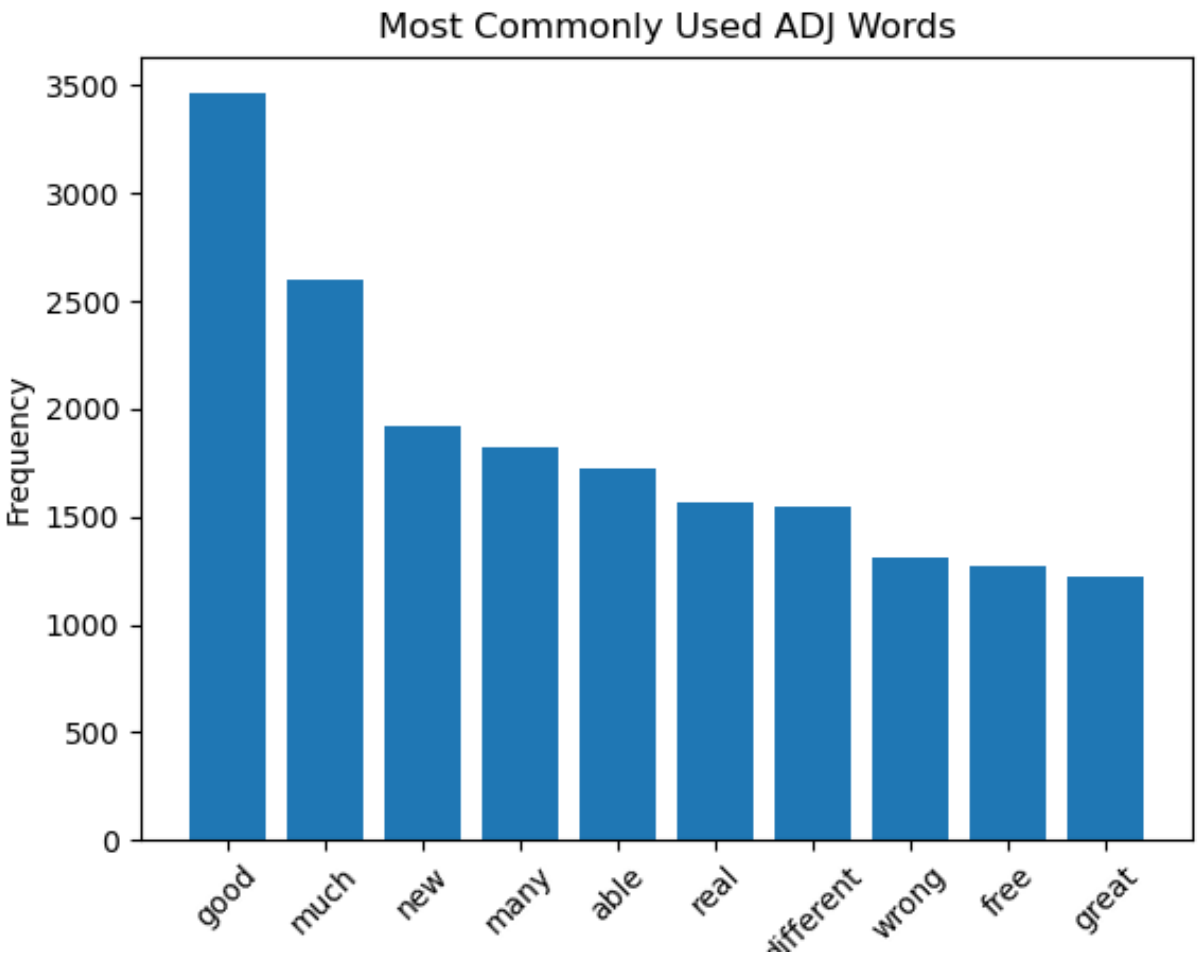
Study the differences in chatgpt views across commuinty

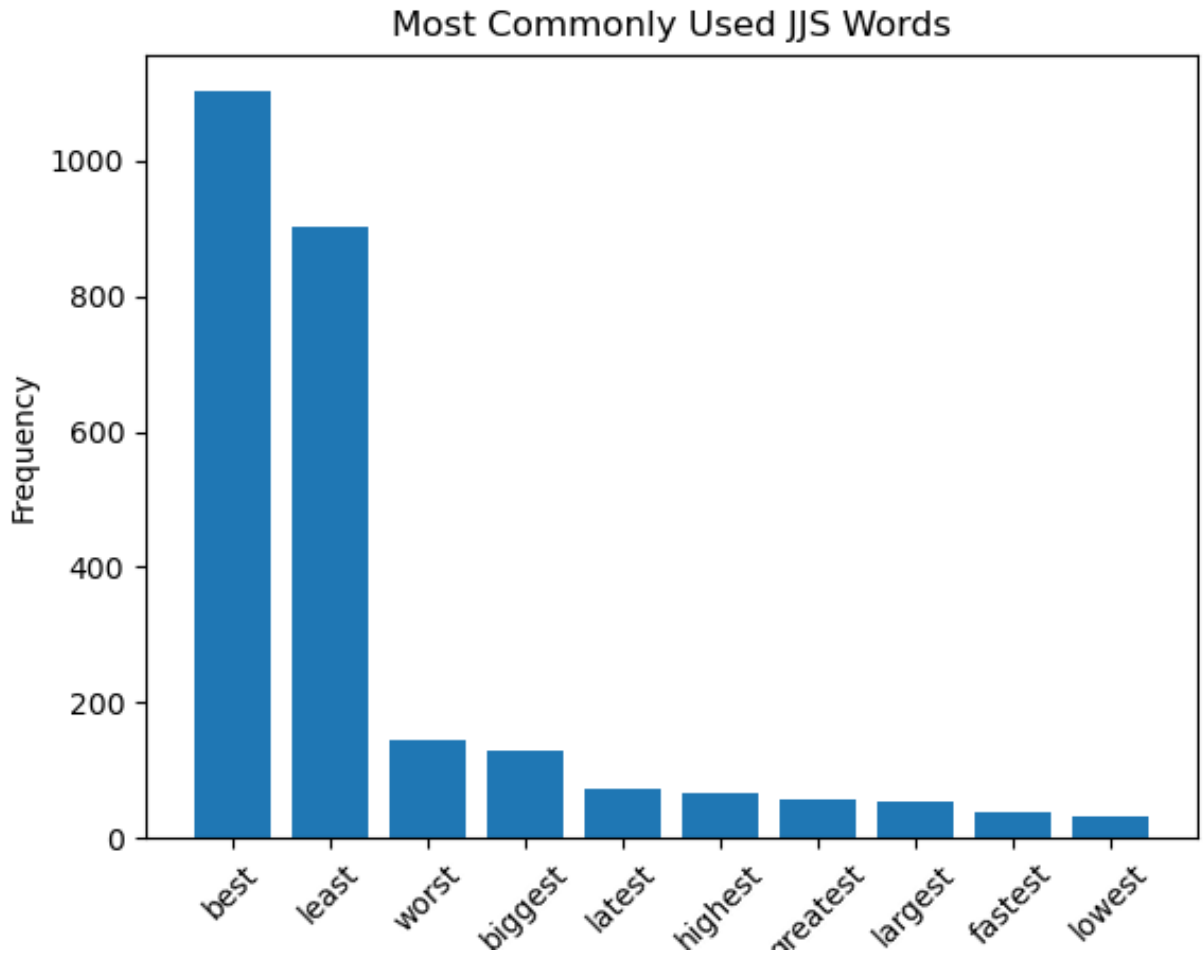
objective 1

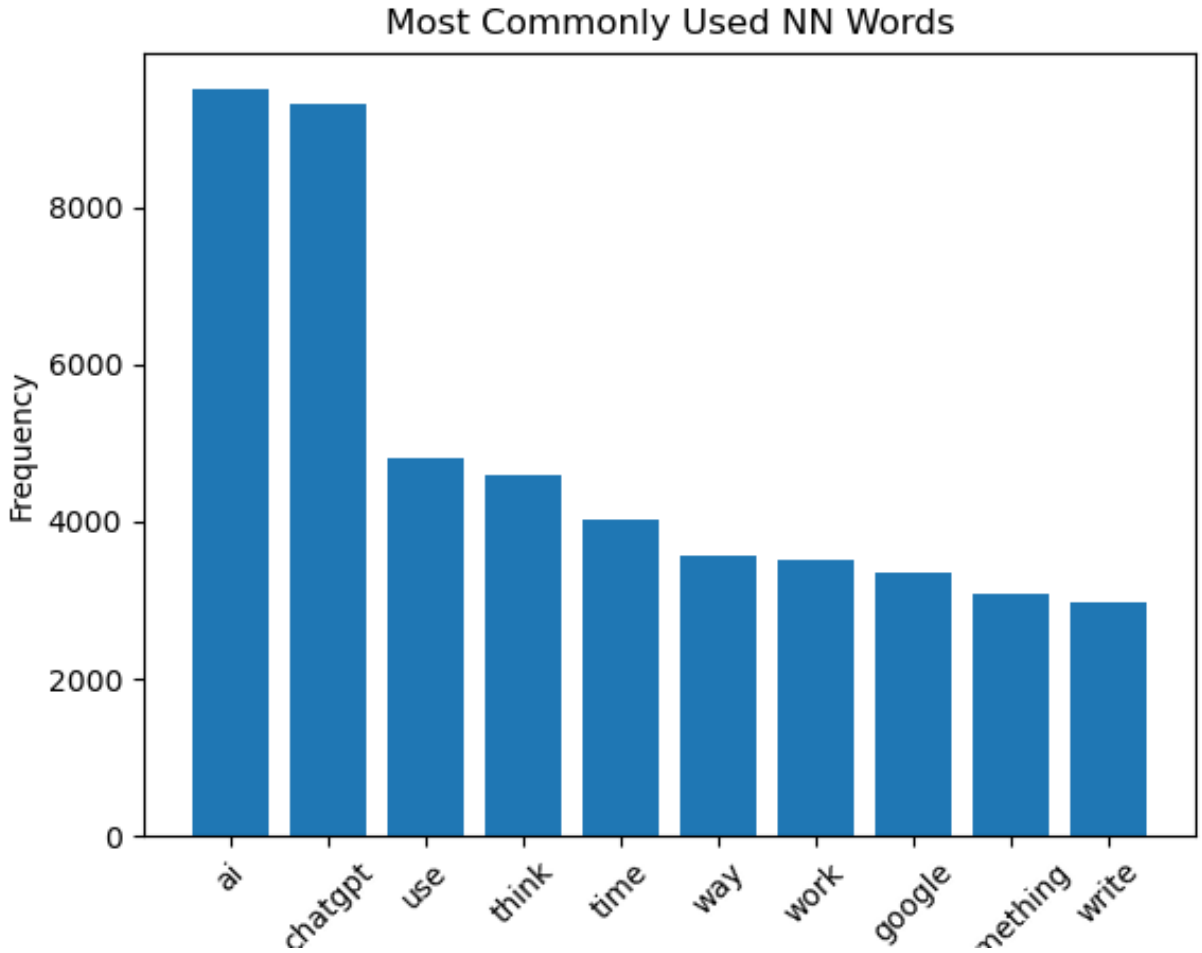
It uses the NLTK library to tokenize and tag the words in the comments, and the Matplotlib library to generate visualizations of the results.

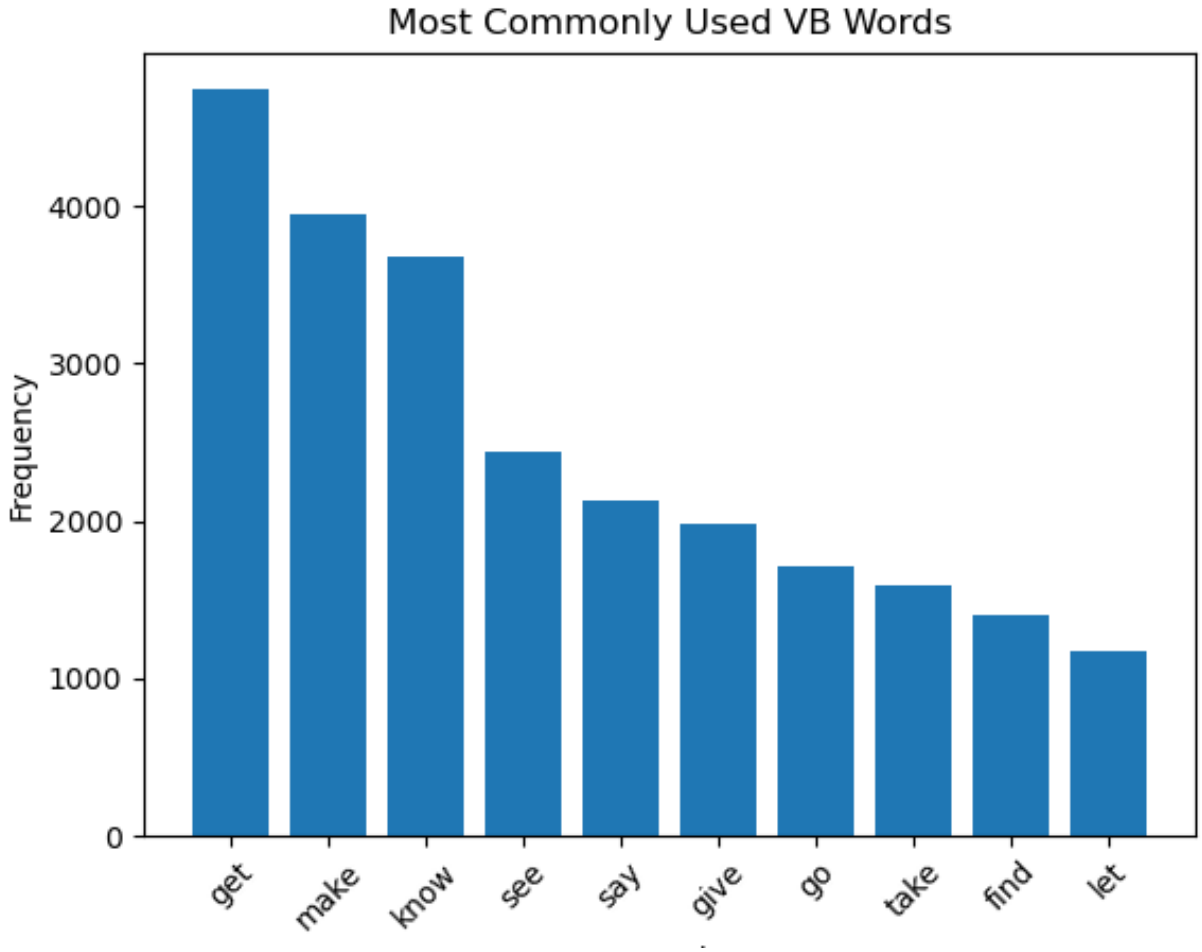
The main function reads the CSV file and extracts the comments. Then, it preprocesses the text by removing URLs, mentions, hashtags, and stop words. After that, it calls two functions: general_counter and part_of_speech.

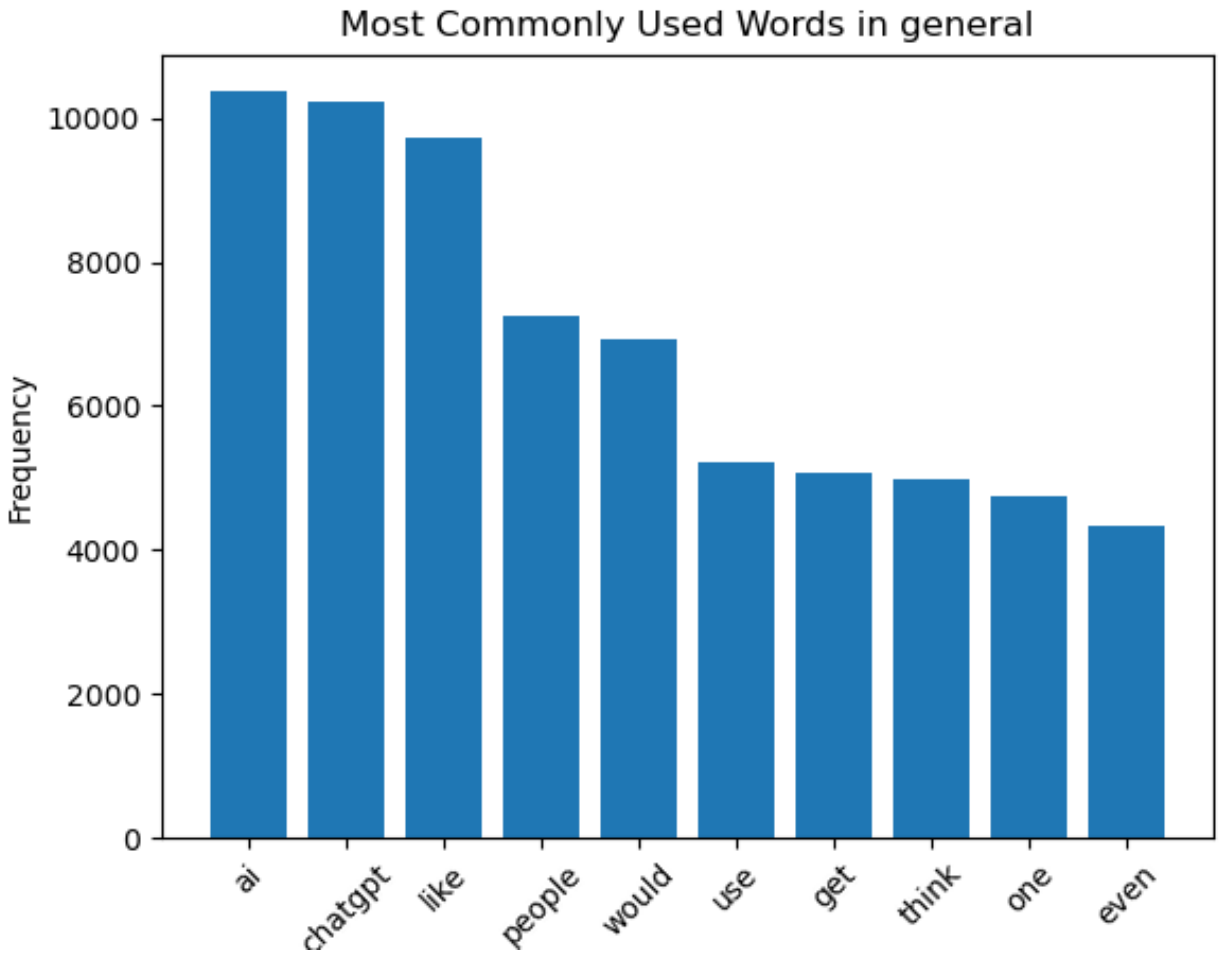
The general_counter function counts the frequency of each word in the comments and generates a bar chart of the top 10 most common words. The part_of_speech function uses NLTK to tag the words in the comments by their part of speech (POS), and then generates bar charts of the top 10 most common adjectives, adverbs, verbs, and nouns.




Most Commonly Used JJS Words
Frequency
0
200
400
600
800
1000
best
least
worst
biggest
latest
highest
greatest
largest
fastest
lowest


Most Commonly Used NN Words
Frequency
8000
6000
4000
2000
0
ai
chatgpt
use
think
time
way
work
google
write


Most Commonly Used VB Words
Frequency
0
1000
2000
3000
4000
get
make
know
see
say
give
go
take
find
let

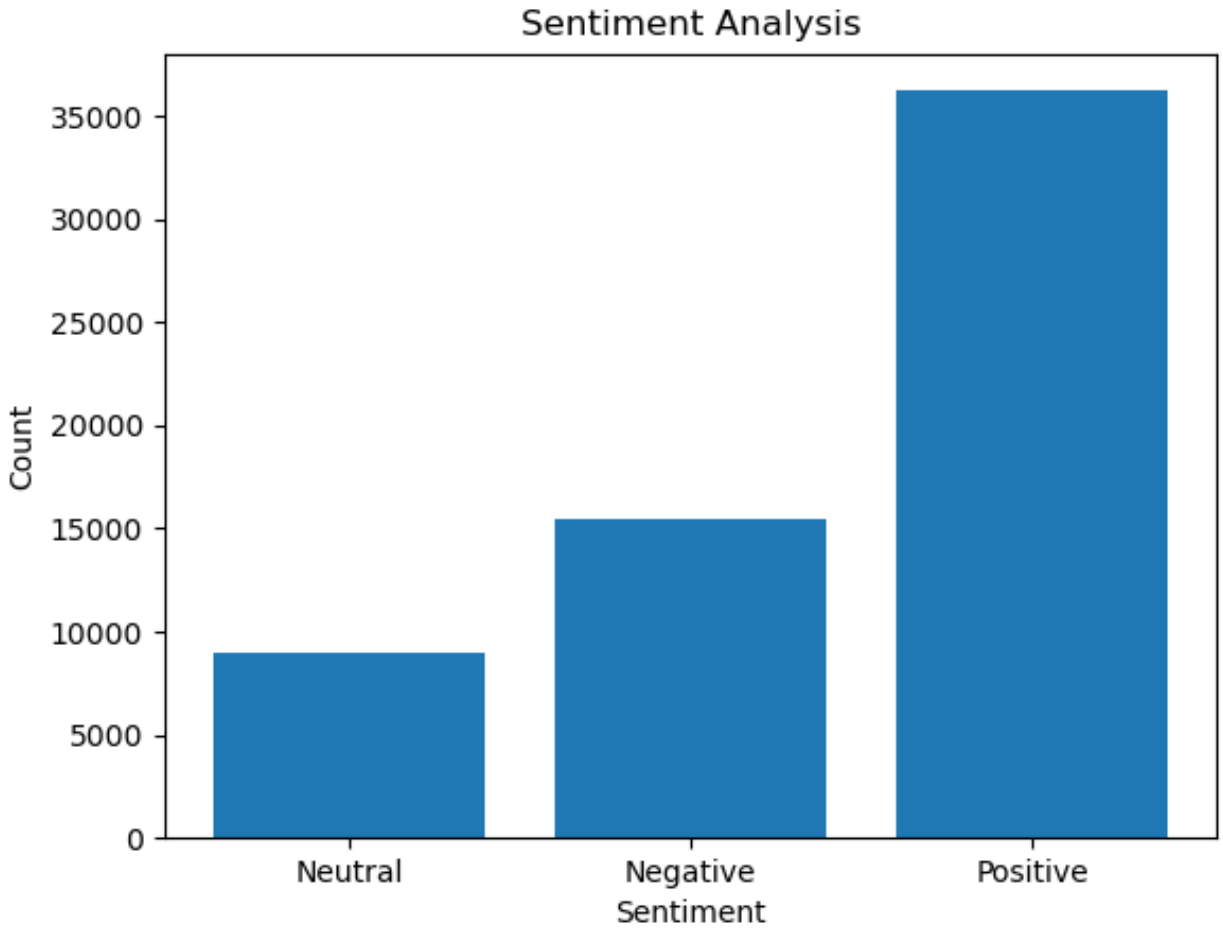


objective 2

How the code works

1. It reads in the CSV file using Pandas, and drops any rows that contain missing values.
2. It extracts the "comment_body" column from the DataFrame, converts it to a NumPy array, and converts all the comments to lowercase.
3. It loads the movie reviews dataset from the NLTK corpus, which contains 1000 positive and 1000 negative reviews.
4. It uses a function called "extract_features" to convert each review into a dictionary of word features. The function takes a list of words and returns a dictionary where each key is a word and its value is True.
5. It uses the "extract_features" function to extract the features of the positive and negative reviews in the movie reviews dataset, and assigns the label "Positive" or "Negative" to each review.
6. It splits the positive and negative reviews into two sets: a training set and a test set. It uses 80% of the data for training and 20% for testing.
7. It trains a Naive Bayes classifier on the training set using NLTK's "NaiveBayesClassifier.train" function.
8. It tokenizes each comment in the Reddit dataset, applies the Naive Bayes classifier to it, and assigns a sentiment label (Positive, Negative, or Neutral) based on the classifier's output. If the classifier outputs a probability less than or equal to 0.55, the comment is labeled as "Neutral".
9. It creates a bar chart showing the distribution of sentiment labels across all the comments, and saves

the chart as "SentimentAnalysis.png".



We can learn from this chart that the vast majority of people are optimistic about chatgpt.

objective 3

我们选择了LDA聚类算法。

我们先是将全部评论用clean_text函数清洗掉url，无意义信息和emoji表情。并且去掉长度小于等于2的文本。

然后将评论列表转换成词列表。并且去掉其中的停止词。

把清洗过后的词列表，用gensim库构建词典，用“filter_extremes()”将词典中只在极少量或者极大部分文档出现的词去掉。并且去掉前15出现频率最大的词去掉。得到最终的字典。最后使用gensim库中的dictionary和doc2bow函数将文本列表 转换成了一个词袋（bag-of-words）语料库（corpus）。

然后把语料库和字典传进lda模型训练。设置topic数量从1到15，并且计算出每次的coherence score和perplexity score。每次训练出来的模型保存在model文件夹下。之后需要使用时直接load。

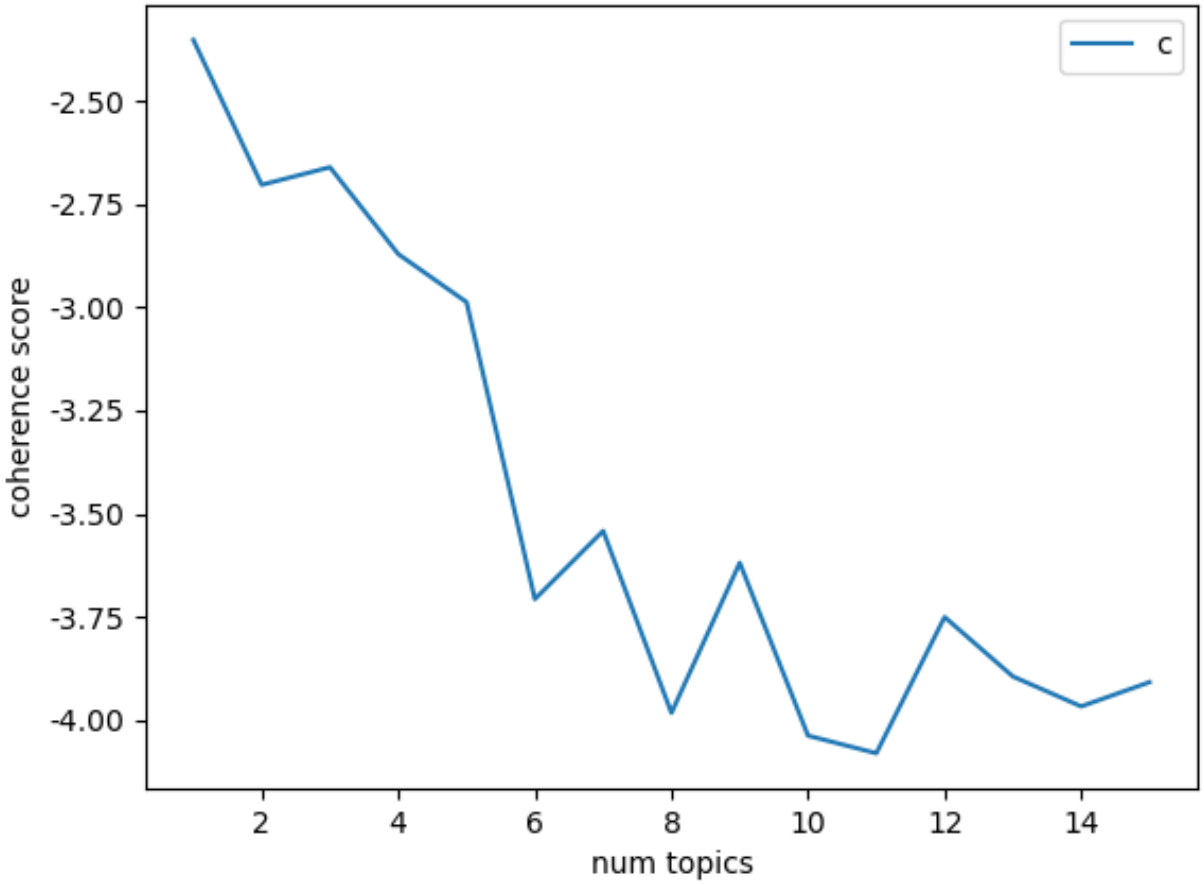
We chose the LDA clustering algorithm.

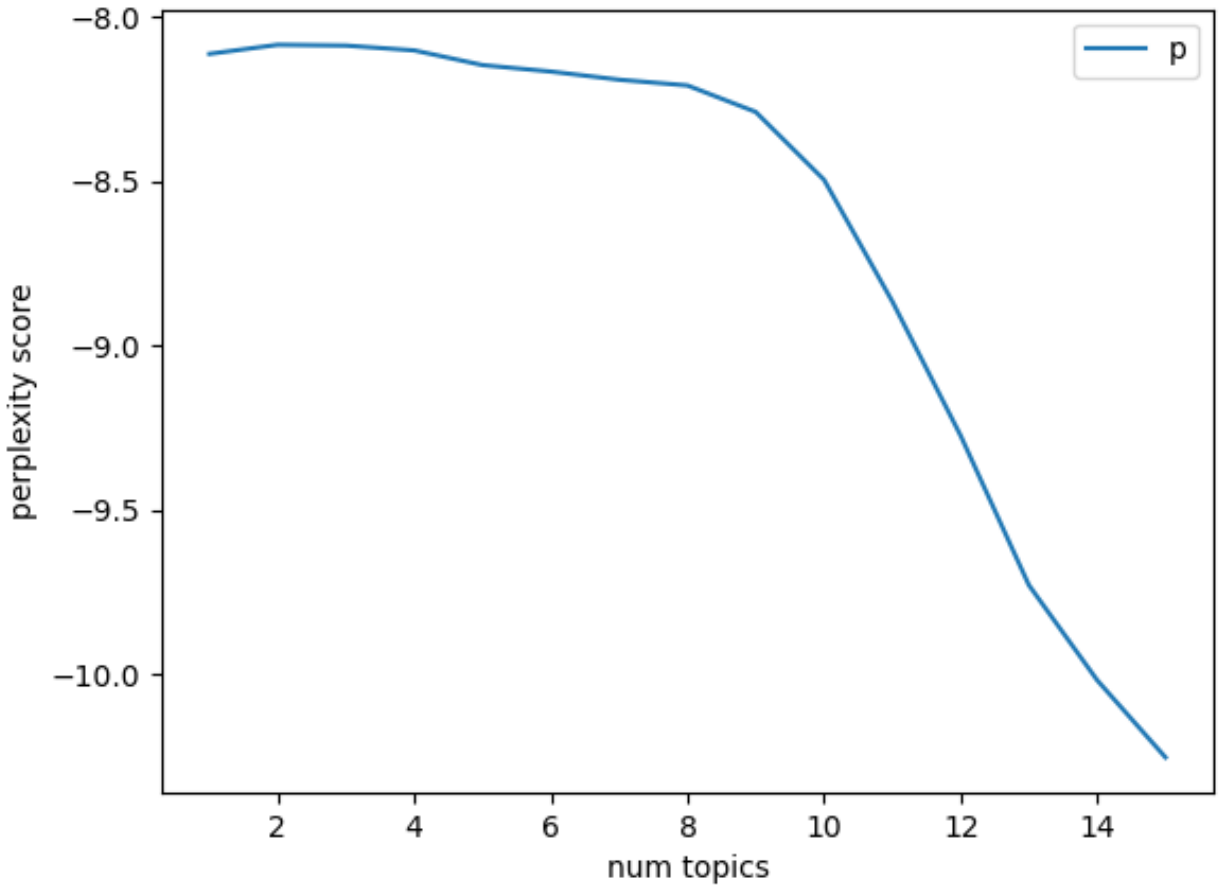
We first cleaned all the comments with clean_text function to remove the url, meaningless information and emoji emoji. And remove the text whose length is less than or equal to 2.

Then we convert the comment list into a word list. And remove the stop words from it.

After cleaning the word list, we use the gensim library to build a dictionary, and use "filter_extremes()" to remove the words that only appear in a very small or large portion of the document. And remove the first 15 words that appear most frequently. Get the final dictionary. Finally, we use the dictionary and doc2bow functions in the gensim library to convert the text list into a bag-of-words corpus.

Then the corpus and dictionaries are passed into the lda model for training. The number of topics is set from 1 to 15, and the coherence score and perplexity score are calculated for each time. Each time the trained model is saved in the model folder, it can be loaded directly when it is needed later.



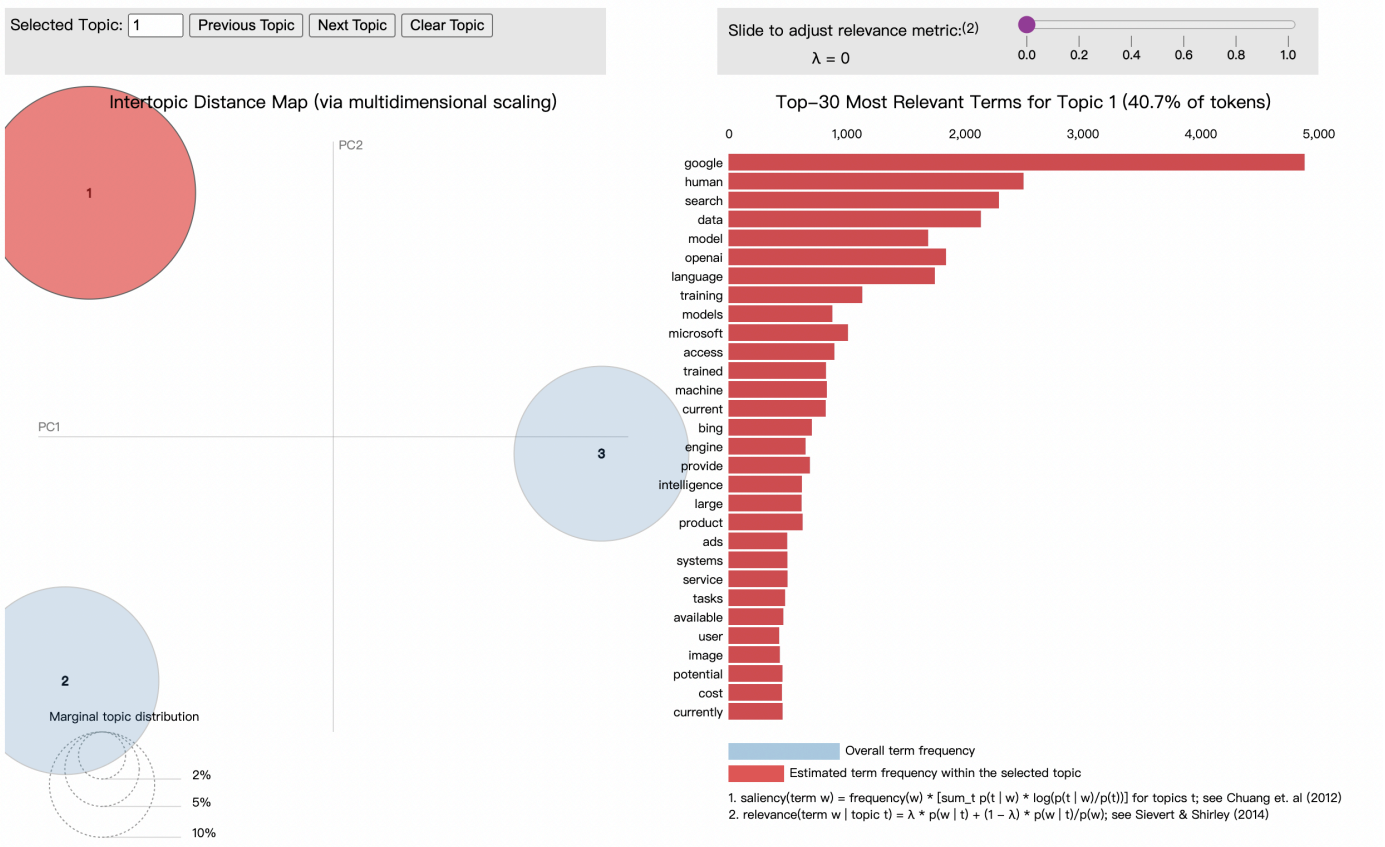


我们可以看出来，当topic等于3时最佳。

下列是用pyLDAvis生成的分析图

We can see that it is best when the topic is equal to 3.

The following is the analysis graph generated with pyLDAvis

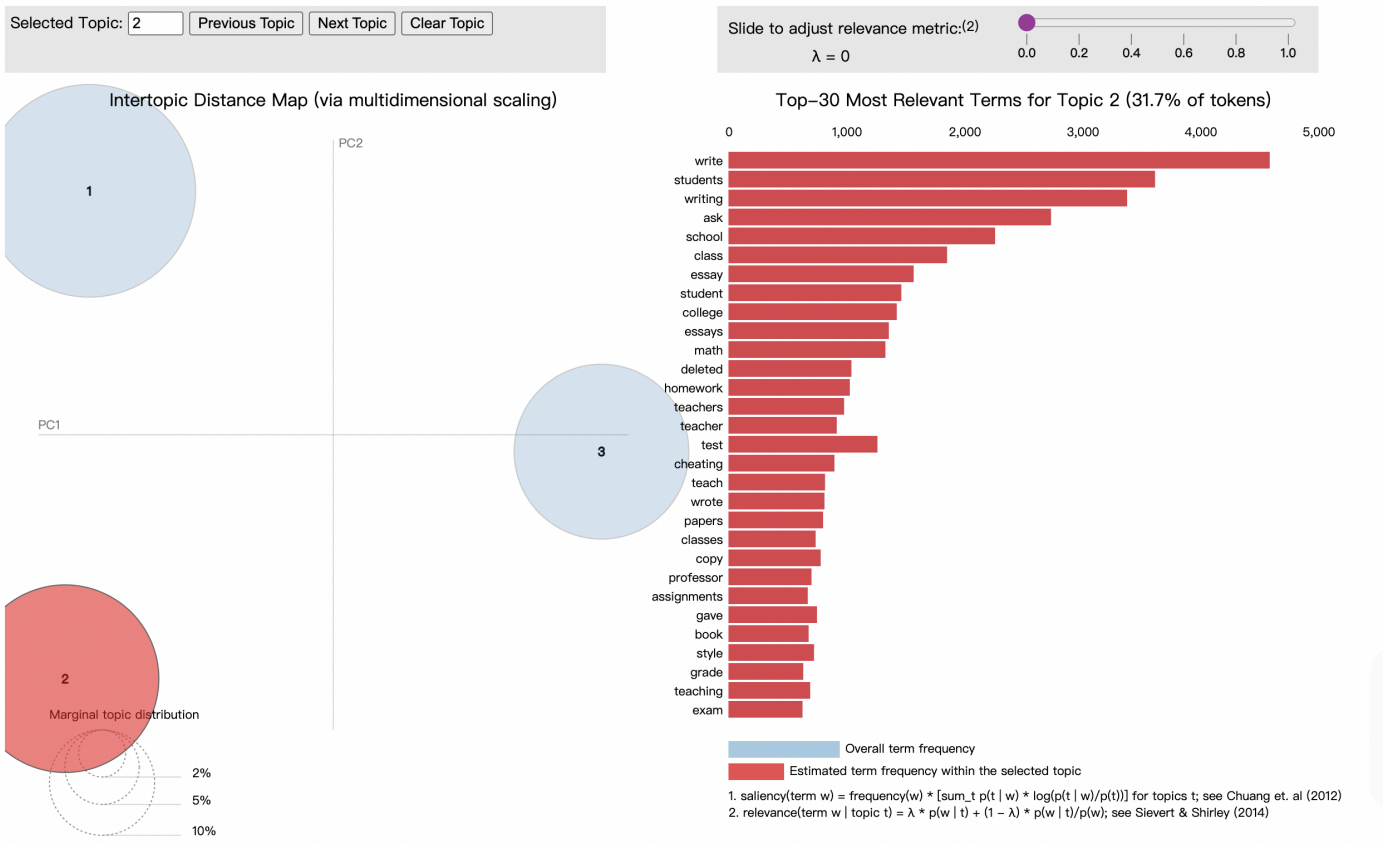


分析可知。

Topic1主要是聚焦于chatgpt实现所需要的技术，以及人工智能技术相关的话题。

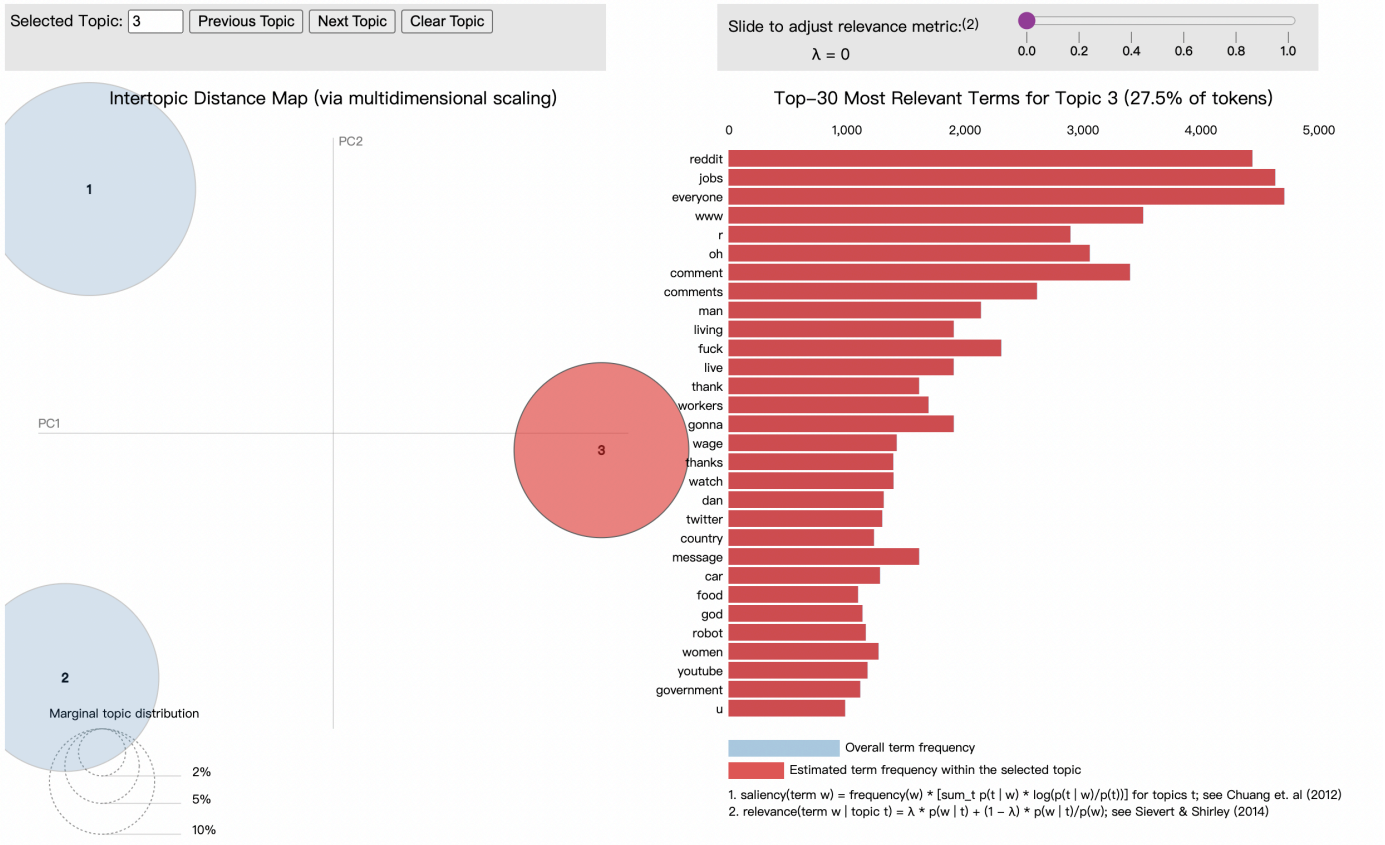
The analysis can be seen.

Topic1 is mainly focused on the technology needed for chatgpt implementation, and AI technology related topics.



Topic2主要是在教育领域，学生和教师使用chatgpt的情况

Topic2 is mainly in the field of education, students and teachers use chatgpt

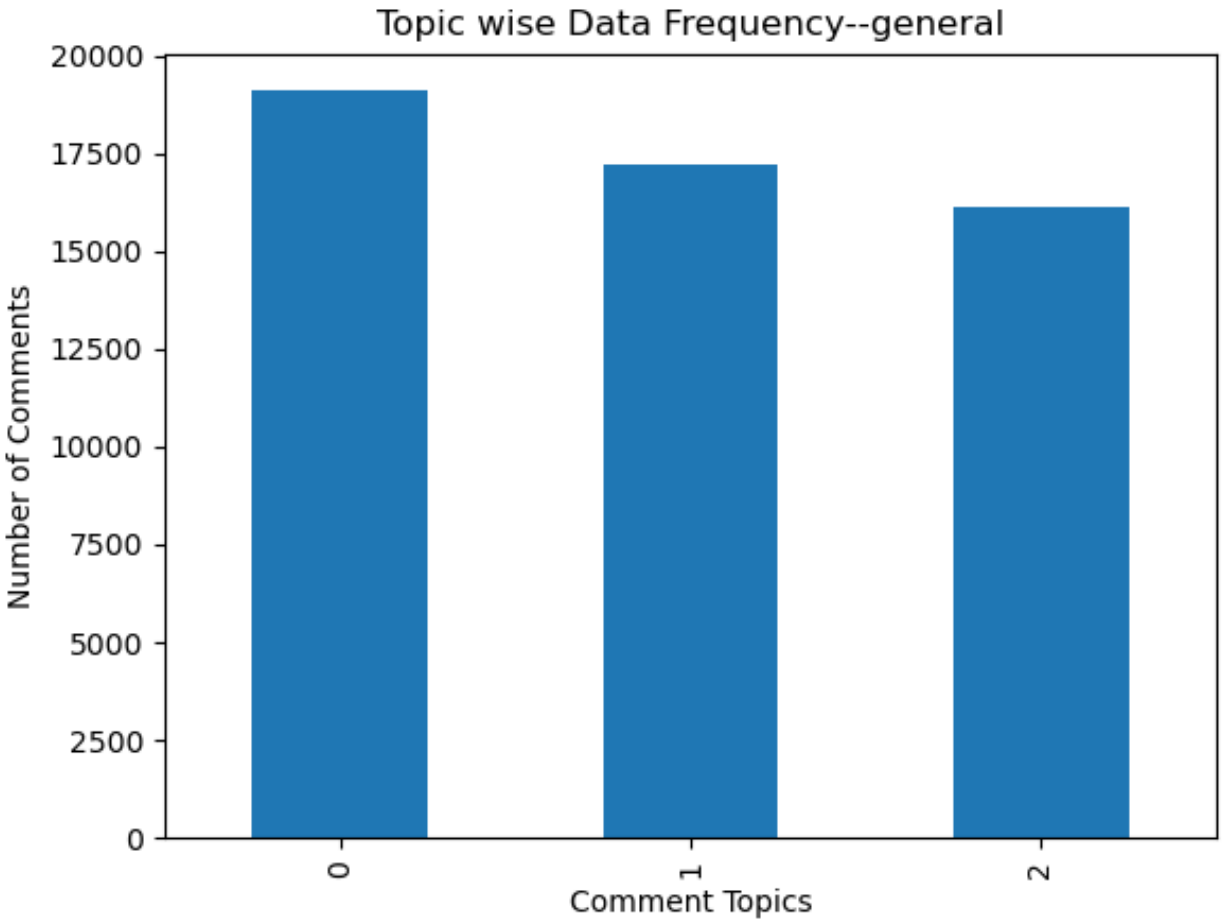


Topic3主要在与人工智能对未来职场的影响。

Topic3 focuses on the impact of artificial intelligence on the future workplace.

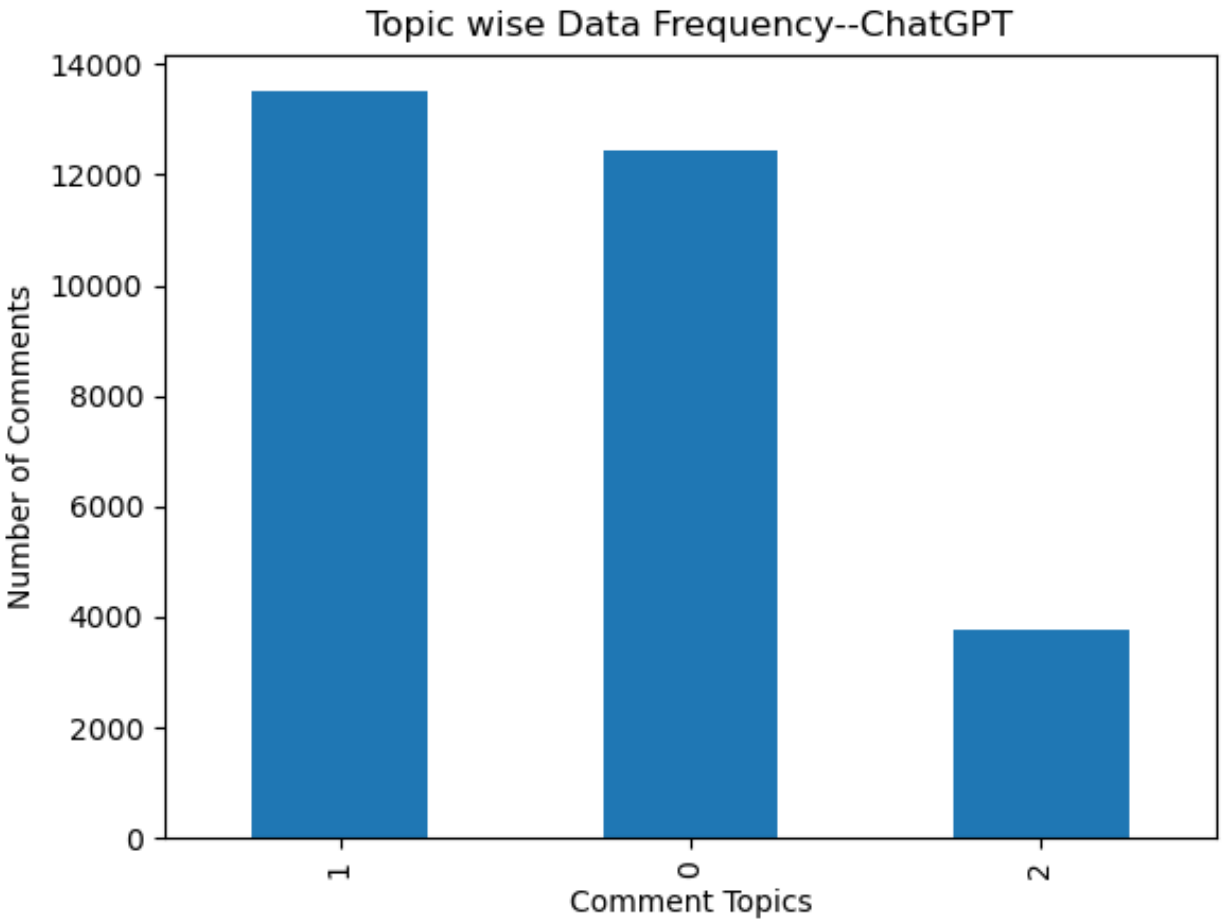
我们可以得到总的Topic分布(0->Topic1, 1->Topic2, 2->Topic3)

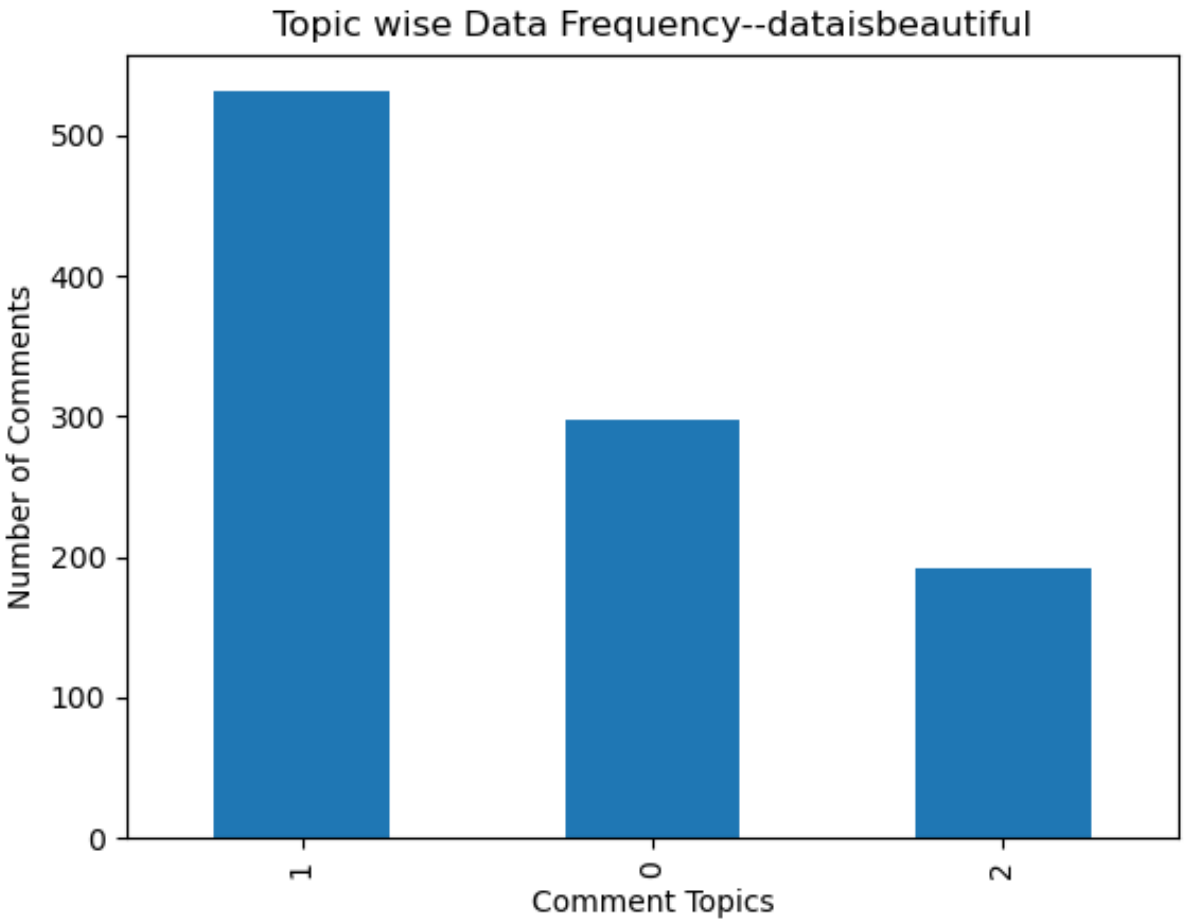
We can get the total Topic distribution (0->Topic1, 1->Topic2, 2->Topic3)

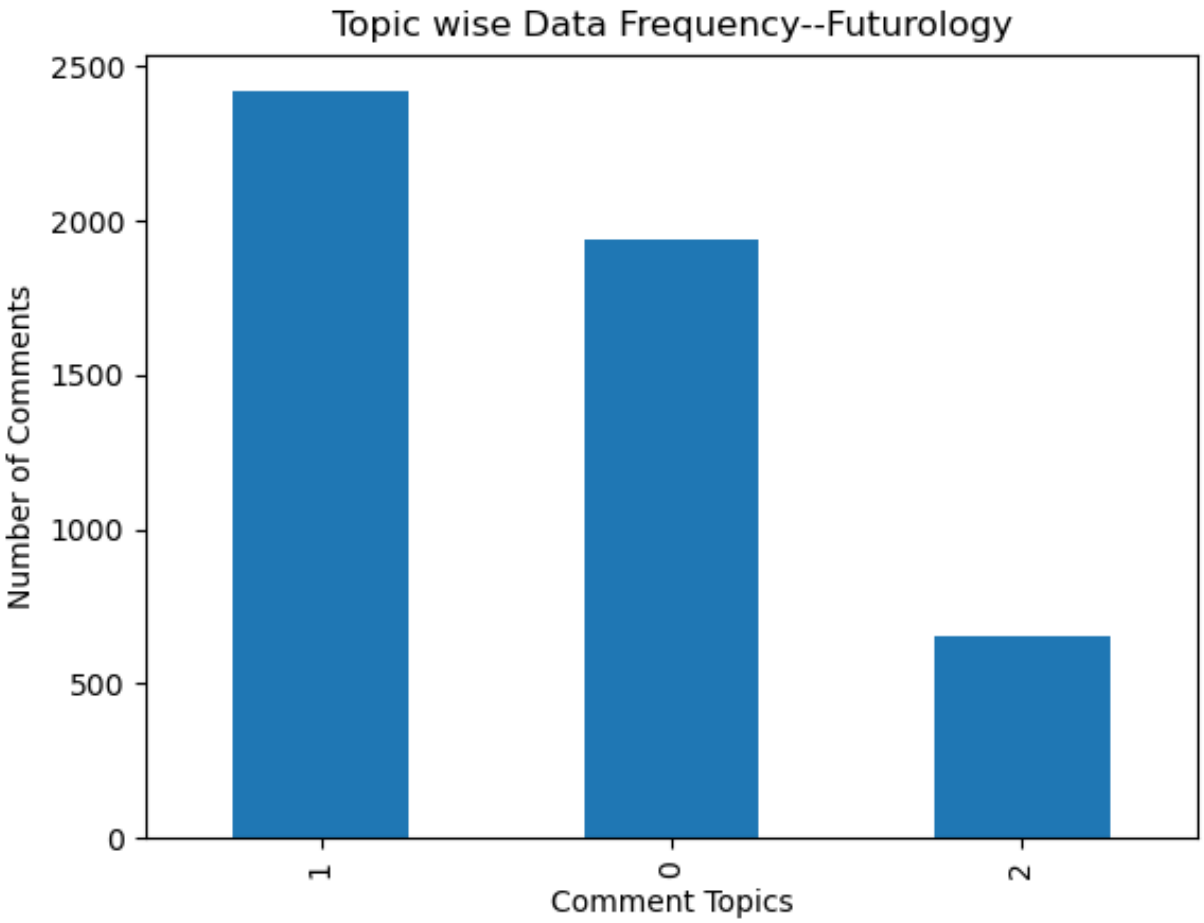


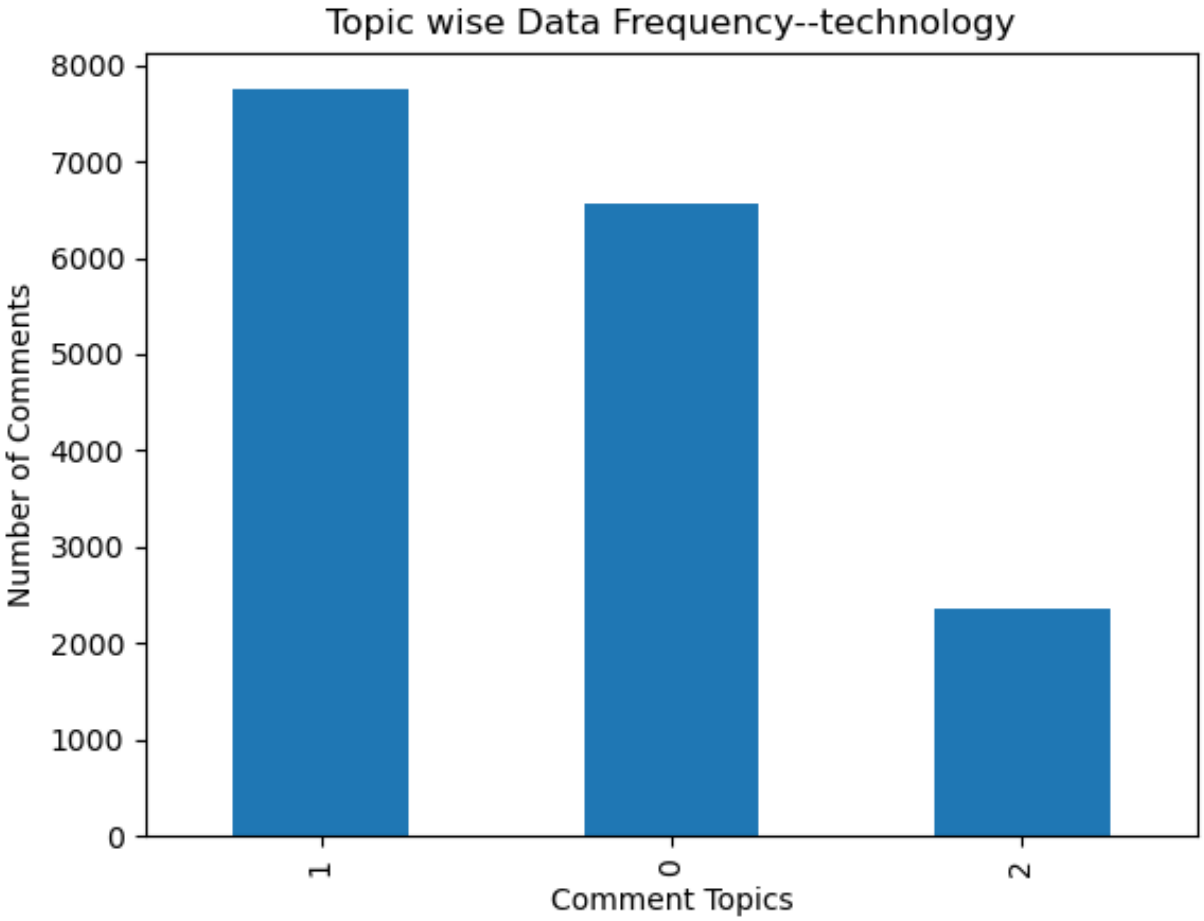
将不同reddit社区的comment通过我们已经训练好的模型，分别得到其分布。

The comments of different reddit communities are put through our already trained model to get their distribution separately.


Topic wise Data Frequency--ChatGPT
14000
12000
10000
8000
6000
4000
2000
0
Number of Comments
1
0
2
Comment Topics


Topic wise Data Frequency--dataisbeautiful
Number of Comments
500
400
300
200
100
0
1
0
2
Comment Topics


Topic wise Data Frequency--Futurology
Number of Comments
2500
2000
1500
1000
500
0
1
0
2
Comment Topics



总体趋势是三种Topic相差并不大，三种话题热度相同。

但在主流的四大社区中，Topic2，即其在教育领域的话题出现的更多。

Topic3出现的概率都是最低。

由总分布可知，Topic1出现的概率最大，Topic3的数量也不少，因此可以推断其他零散的社区讨论Topic1和2的比较多。

The overall trend is that the three Topics do not differ much from each other, and the three topics are equally hot.

However, among the four mainstream communities, Topic2, i.e. its topics in the field of education, appeared more often.

The probability of Topic3 appearing are the lowest.

From the total distribution, it can be seen that the probability of Topic1 appearing is the highest, and the number of Topic3 is not small, so it can be inferred that other fragmented communities discuss Topic1 and 2 more.