ML ASKREDDIT PROJECT

TEAM - NLP
SHIVYANSH GARG (IMT2019082)
SAURABH KUMAR SHARMA(IMT2019078)

PROBLEM STATEMENT

The moderators of AskReddit continuously strive to remove "troll" questions, however, it is not possible to manually inspect every question and do this -- thousands of questions are added every day.

Which is where OUR WORKS come in. The moderators of AskReddit have provided us with a sample of all the questions they received last year.

Our job: To create a model capable of automatically detecting troll questions so that they can be flagged and removed.

ABOUT DATASET

The data set was provided on Kaggle. The data set included three files:

- train.csv - the training set contains the labels named as 'target'
- test.csv - the test set, which does not contain the target variable 'target' .

Overall, the dataset has id colums along with question text and target variable.

Trainning data has 653061 entries

Test data has 653061 entries

SAMPLE DATA

	qid	question_text	target
0	a3dee568776c08512c89	What is the role of Lua in Civ4?	0.0
1	bdb84f519e7b46e7b7bb	What are important chapters in Kannada for 10 ...	0.0
2	29c88db470e2eb5c97ad	Do musicians get royalties from YouTube?	0.0
3	3387d99bf2c3227ae8f1	What is the difference between Scaling Social ...	0.0
4	e79fa5038f765d0f2e7e	Why do elevators go super slow right before th...	0.0
5	99912c31a1b6e043e776	Could the Jewish mafia control certain scienti...	0.0

EXPLORATORY DATA ANALYSIS

PIE CHART

Shows no. of labels counts

WORDCLOUD

Shows which words are frequent

GRAPH

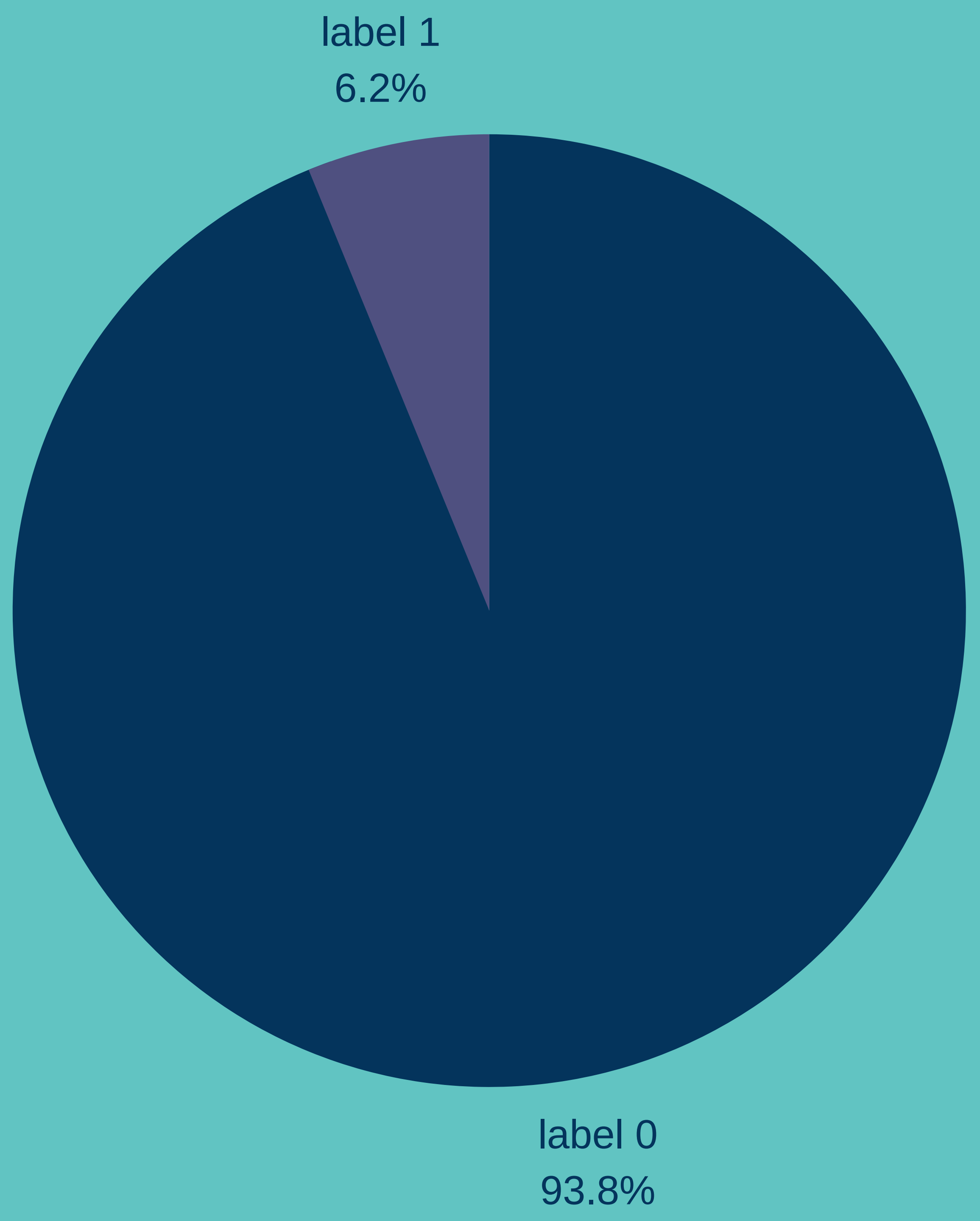
Shows most used words

HISTOGRAM

Shows length vs count of labels

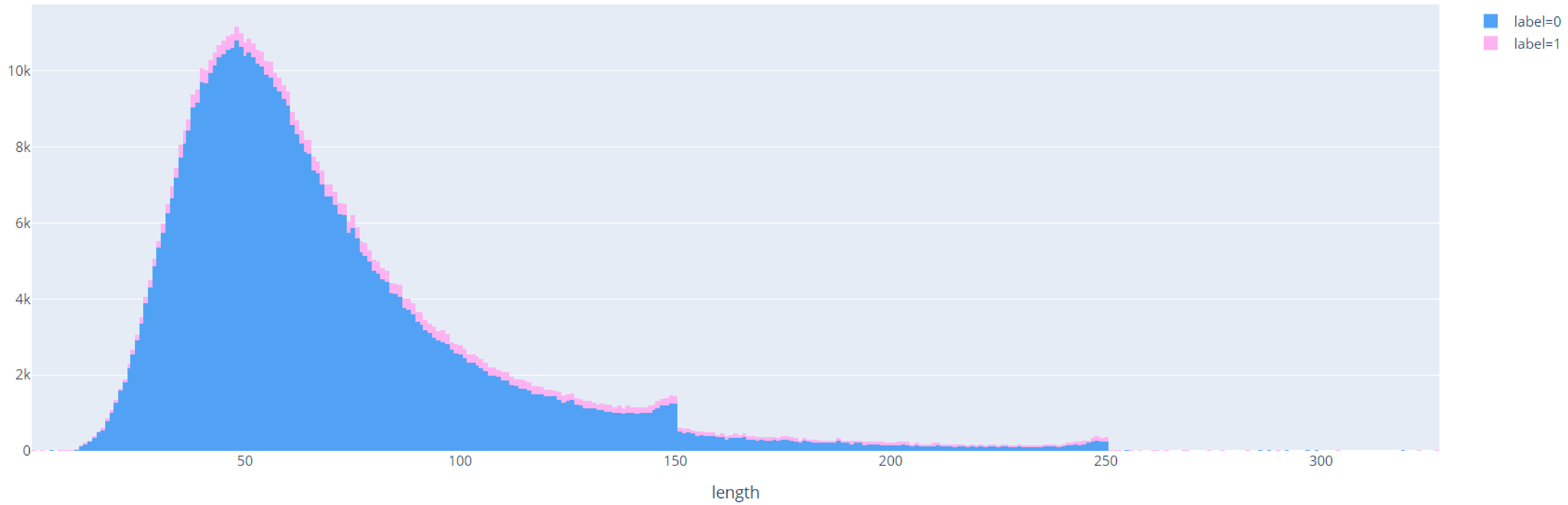
PIE CHART

This chart shows us data entries with label 0 are way higher than data entries with label 1



HISTOGRAM

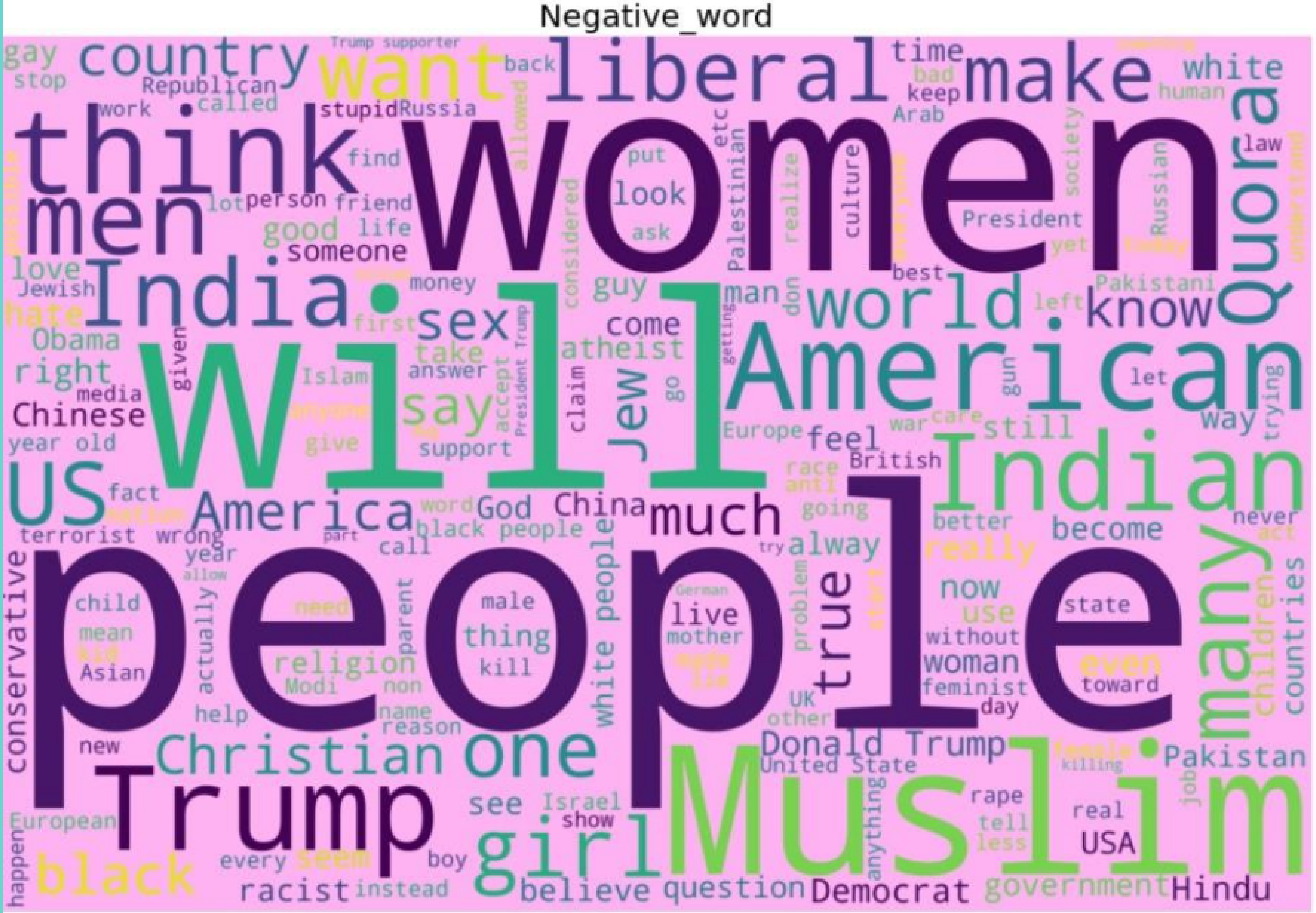
Shows us lenghts vs counts in our data, of both of our labels


label=0
label=1
10k
8k
6k
4k
2k
0
50
100
150
200
250
300
length

WORD CLOUD

Word cloud showing all the prominent words among all the data entries with different labels

label=1



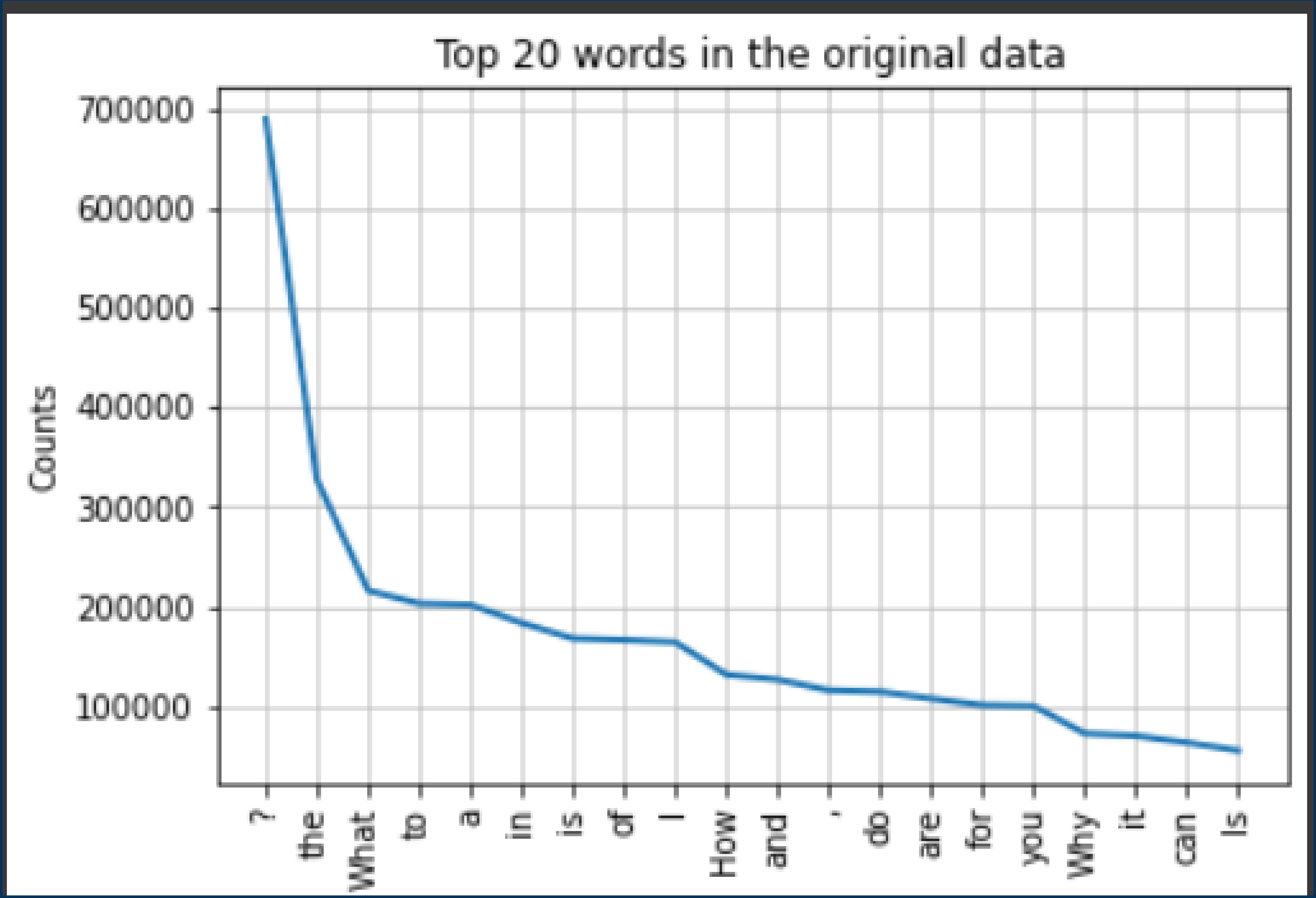
Negative_word
women
will
people
Muslim
think
liberal
make
want
country
men
India
American
Indian
world
know
Quora
US
Trump
America
much
many
Christian
one
girl
sex
Jew
say
true
black
Donald Trump
Chinese
Obama
right
white
time
Republican
stupid
Russia
look
atheist
come
guy
good
life
someone
love
Jewish
hate
Pakistani
President
feel
still
way
British
Europe
God
China
terrorist
wrong
black people
support
conservative
child
thing
religion
Modi
kill
live
mother
now
state
woman
feminist
toward
Pakistan
United State
rape
real
USA
Hindu
Democrat
government
question
believe
racist
instead
boy
every
see
Israel
show
European
happen
new
help
name
reason
Palestinian
culture
Russian
society
alway
become
better
never
white people
children
countries
Asian
mean
male
Islam
answer
media
year old
fact
year
call
money
person
friend
find
work
called
back
stop
gay
keep
Arab
law
human
world
let
trying
gun
war
care
race
going
really
even
without
day
UK
other
problem
anything
tell
less
seem

label=0

Postive_words
best
one
will
India
make
good
use
people
think
better
life
find
now
become
really
answer
difference
time
mean
many
work
know
much
still
need
want
even
name
possible
world
start
different
country
friend
Indian
made
first
person
without
take
someone
book
student
product
day
happen
say
year
change
cause
school
look
feel
go
way
come
used
help
job
girl
important
experience
relationship
new
sex
family
write
show
exam
phone
children
business
college
future
human
reason
type
buy
anyone
government
American
United States
class
number
play
value
thought
kind
idea
movie
game
well
system
house
state
effect
instead
problem
word
animal
love
great
create
food
normal
China
view
right
Canada
law
tell
service
question
learn
apply
purpose
true
Quora
UK
bad
car
order
guy
stop
back
got
around
history
online
other
kid
using
function
considered
example
something
company
group
give
app
home
called
program
USA
actually
man
top
given
child
point
America
benefit

GRAPH

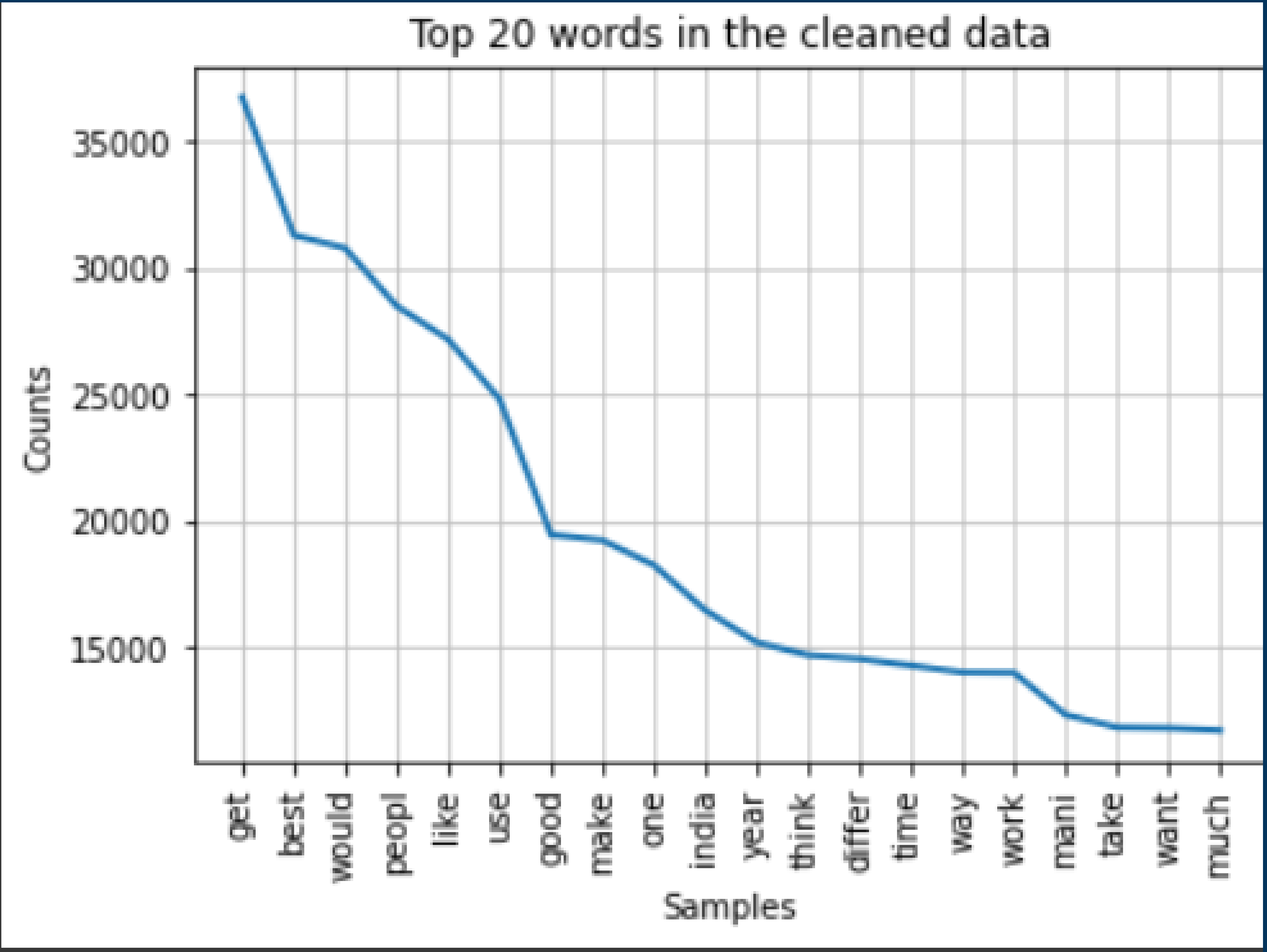
It gives us intuition depicting which words are used most often in original and cleaned data



Counts vs Words

IN ORIGINAL DATA

WE CAN SEE MANY STOPWORDS IN OUR DATA



Counts vs Words

IN CLEANED DATA

ALL THOSE STOP WORDS ARE REMOVED AFTER CLEANING

PREPROCESSING

CLEANING TEXT

STEMMING

OVER SAMPLING

VECTORIZER

CLEAN TEXT STEMMING

We see many stopwords and unncessary characters in our data entry and these are cleaned using the function clean text.

Some functionalities include:-

- remove special characters
- remove some webaddress
- remove stopwords
- uppercase to lowercase
- used stemming using snowballstemmer

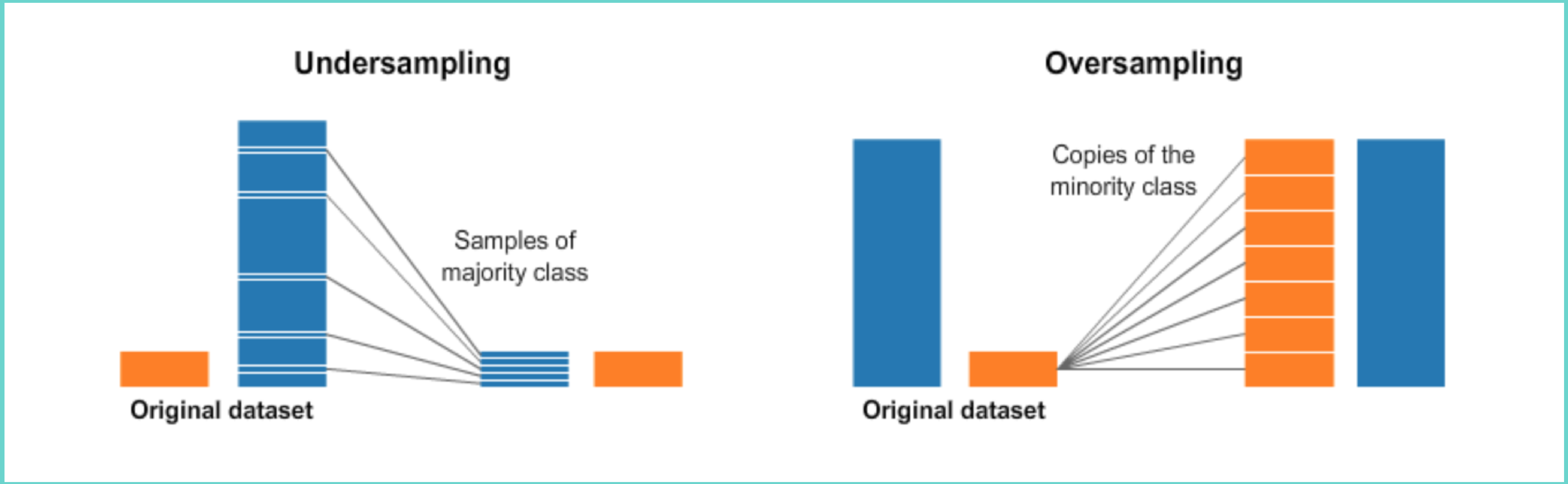
CLEAN TEXT

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
nltk.download(['punkt', 'stopwords'])
ss = nltk.SnowballStemmer("english")
STOPWORDS = set(stopwords.words('english'))
def clean_text(text):
    clean_text = re.sub(r'^.+@[^\.].*\.[a-z]{2,}$',' ',text)
    #remove some webaddress
    clean_text = re.sub(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',' ',clean_text)
    #some moneysymbols
    clean_text = re.sub(r'£|\$',' ',clean_text)
    #phone numbers
    clean_text = re.sub(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',' ',clean_text)
    #removed any number
    clean_text = re.sub(r'\d+(\.\d+)?',' ',clean_text)
    #remove punctuation
    clean_text = re.sub(r'[^\w\d\s]',' ',clean_text)
    # remove special characters
    clean_text = re.sub(r'[^0-9a-zA-Z]', ' ',clean_text)
    # remove extra spaces
    clean_text = re.sub(r'\s+', ' ', clean_text)
    # convert to lowercase
    clean_text = clean_text.lower()
    # remove stopwords
    clean_text = ' '.join( word.lower() for word in word_tokenize(clean_text)if word.isalpha() and word not in STOPWORDS)
    clean_text = ' '.join(ss.stem(term) for term in clean_text.split())
    return clean_text
```

OVERSAMPLING

Randomly duplicate examples in the minority class

This technique is effective we are experiencing a skewed distribution as shown in the pie chart



```python
#Resampling the dataset
from sklearn.utils import resample
zero_data = train_df[train_df["target"] == 0]
one_data = train_df[train_df["target"] == 1]
train_df = pd.concat([resample(zero_data, replace = True, n_samples = len(one_data)*6), one_data])
```

While doing oversampling, we had the opportunity to decide how many samples we have to add in the trainning dataset.

Out of all the possibilities, we came up with this particular no. of samples.

i.e n_samples = len(one_data) *6;

i.e 6 times the count of label 1 data.

VECTORIZER

It's a process to map words or phrases from vocabulary to a corresponding vector of real numbers which used to find word predictions, similarities. The process of converting words into numbers are called Vectorization

The Count Vectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
vectorizer = CountVectorizer(preprocessor=clean_text, ngram_range=(1, 3) , strip_accents = 'ascii')
```

MODELS

- NAIVE BAYES
- LOGISTIC REGRESSION
- XGBOOST
- ADABOOST
- RANDOMFOREST

NAIVE BAYES

```
from sklearn.naive_bayes import MultinomialNB
naive_bayes = MultinomialNB()
naive_bayes.fit(X_train, y_train)
```

```
MultinomialNB()
```

NAIVE BAYES CLASSIFIERS ARE A COLLECTION OF CLASSIFICATION ALGORITHMS BASED ON BAYES' THEOREM.

LOGISTIC REGRESSION

```
from sklearn import model_selection, metrics, linear_model
logistic = linear_model.LogisticRegression(solver='sag', max_iter = 1000)
logistic.fit(X_train, y_train)
```

```
LogisticRegression(max_iter=1000, solver='sag')
```

WE CHOSE SOLVER TO BE "SAG" CONSIDERING IT GAVE HIGHER ACCURACY

XGBOOST

```
from xgboost import XGBClassifier
# declare parameters
params = {
            'objective':'binary:logistic',
            'max_depth': 4,
            'alpha': 10,
            'learning_rate': 1.0,
            'n_estimators':100
        }
# instantiate the classifier
xgb_clf = XGBClassifier(**params, use_label_encoder =False)
xgb_clf.fit(X_train, y_train)
```

ADABOOST

```
from sklearn.ensemble import AdaBoostClassifier
ada_boost = AdaBoostClassifier(random_state = 96)
ada_boost.fit(X_train, y_train)
```

RANDOM FOREST

```
from sklearn.ensemble import RandomForestRegressor
random_forest = RandomForestRegressor(n_estimators = 1000, random_state = 42)
random_forest.fit(X_train, y_train)
```

ACCURACY

NAIVE BAYES	0.7064
LOGISTIC REGRESSION	0.7301
XGBOOST	0.6666
ADABOOST	0.6878
RANDOMFOREST	06945

KAGGLE SCORE

0.62130

PUBLIC LEADERBOARD

0.63328

PRIVATE LEADERBOARD

CONCLUSION

- This project helped us gently get the idea of the domain of NLP.
- Out of all the models used, Logistic Regression model gave us the best accuracy.
- Hence, we used Logistic Regression model to predict which of the questions are the troll questions.
- We also faced some challenges; eventually, those problems only helped us learn new things and apply them

CONTRIBUTION

We did the whole project as a team. We discussed everything and implemented everything together.

SHIVYANSH

Worked upon the base code to improve accuracy until 0.70

Helped in EDA of the code and formation of wordcloud

SAURABH

Provided the base code to work upon with accuracy 0.68.

Helped in preprocessing of the data.

Worked together to tune the model to recieve the best accuracy

THANKS

MERRY CHRISTMAS
AND
HAPPY NEW YEAR