a. What you learned in the ICP

What I learned from this ICP was how to clean up my data. Few steps had to be considered before it could be analyzed. One for the first things I learned was how to remove unnecessary columns of data. Then learning how to remove Twitter handles, punctuations, and special characters . By following these steps, I was able to shrink the data size and only have the information I would need. Few other concepts were also applied like tokenization. Tokenization is breaking up sequence of strings or a sentence into just single words, symbols, phrases, or any other element that could be present. Also, lemmatization was applied to the data. The purpose of lemmatization is to reduce a word into its original dictionary root. Another concept I learned was TF-IDf which stands for "Term Frequency – Inverse Document Frequency". The technique is used to quantify the words in a document and then compute the weight of each word, therefore signifying the importance of the word. The concept of POS tagging was also utilized to understand the concept. Also, learning how to check for missing values, and training, testing, and splitting of data. Few data visualization and analysis were also learned.

b. ICP description what was the task you were performing

The following is general outline of the tasks performed.

1. The required libraries were imported, and after doing so the data was read.
2. After reading the data few commands were used to see the basic information about what was in the data. EX: data.head(10) to see the columns, labels, and ID
3. Since the id column was no use it was removed using (data.drop() function).
4. After successful completion of column removal, the first 20 tweets were converted into string. By doing so removes any ambiguity in the data. Therefore, that entire data is one data type, and making it easier to work with.

5. Then any special charters or elements that were present were removed from the tweets. The reason for doing so enables us to shrink the data size and remove unwanted data which as no meaning. A frequency method was applied to see the most common and then plotted.
6. Stop words were also removed which do not add much meaning.
7. After removal of stop words stemming and Lemmatization were implemented.
8. POS tagging and TDIDF method were also applied to the data
9. Performed a classification test to see the weighted avg of the data.
10. Checked for any missing data via the isnull() function.
11. Lastly, data visualization was applied using word cloud and bar plot and linear graph.

c. Challenges that you faced

One of the main challenges that I faced was how to apply a predictive modeling to my data. Its something I am not very familiar with and hope to learn more about it as the semester goes on.

d. Screen shots that shows the successful execution of each required step of your code

First thing is to install the required libaries

```
[1]  import pandas as pd

     import nltk
     from nltk import sent_tokenize
     from nltk import word_tokenize


     import nltk
     nltk.download("popular")
```

```
[nltk_data] Downloading collection 'popular'
[nltk_data]    |
[nltk_data]    | Downloading package cmudict to /root/nltk_data...
[nltk_data]    |   Package cmudict is already up-to-date!
[nltk_data]    | Downloading package gazetteers to /root/nltk_data...
[nltk_data]    |   Package gazetteers is already up-to-date!
[nltk_data]    | Downloading package genesis to /root/nltk_data...
[nltk_data]    |   Package genesis is already up-to-date!
[nltk_data]    | Downloading package gutenberg to /root/nltk_data...
[nltk_data]    |   Package gutenberg is already up-to-date!
[nltk_data]    | Downloading package inaugural to /root/nltk_data...
[nltk_data]    |   Package inaugural is already up-to-date!
[nltk_data]    | Downloading package movie_reviews to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package movie_reviews is already up-to-date!
[nltk_data]    | Downloading package names to /root/nltk_data...
[nltk_data]    |   Package names is already up-to-date!
[nltk_data]    | Downloading package shakespeare to /root/nltk_data...
[nltk_data]    |   Package shakespeare is already up-to-date!
[nltk_data]    | Downloading package stopwords to /root/nltk_data...
[nltk_data]    |   Package stopwords is already up-to-date!
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Package treebank is already up-to-date!
[nltk_data]    | Downloading package twitter_samples to
[nltk_data]    |     /root/nltk_data...
```

Reading the data csv file mentioned in the ICP 3 outline

```
[2] data = pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_Analysis/master/train.csv')
```

The first few codes will be looking at the data. To see what is all in there.

```
[3] data.size #Return an int representing the number of elements in this object.
```

```
95886
```

```
data #displays the data in the csv file
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
...	...	...	...
31957	31958	0	ate @user isz that youuu?ðððððð...
31958	31959	0	to see nina turner on the airwaves trying to...
31959	31960	0	listening to sad songs on a monday morning otw...
31960	31961	1	@user #sikh #temple vandalised in in #calgary,...
31961	31962	0	thank you @user for you follow

31962 rows × 3 columns

If I want to look at only the first 10

```
[5] data.head(10) #looking at first 10
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
5	6	0	[2/2] huge fan fare and big talking before the...
6	7	0	@user camping tomorrow @user @user @user @use...
7	8	0	the next school year is the year for exams.ð...
8	9	0	we won!!! love the land!!! #allin #cavs #champ...
9	10	0	@user @user welcome here ! i'm it's so #gr...

Removing the a column. I will remove the id label and would like to only see the label and tweets only.

```
[6] #axis meaning:  Whether to drop labels from the index (0 or 'index') or columns (1 or 'columns').
    #inplace meaning:   If True, do operation inplace and return None.
    data.drop('id',axis=1,inplace=True) #Will drop the column . The axis one means it will drop the column and axis = 0 means it
```

```
[7] data.head(5) # now looking at the data after the id column has been removed
```

	label	tweet
0	0	@user when a father is dysfunctional and is s...
1	0	@user @user thanks for #lyft credit i can't us...
2	0	bihday your majesty
3	0	#model i love u take with u all the time in ...
4	0	factsguide: society now #motivation

Converting the tweet text to string

```
[8] # it is good to convert to string because the vartion of data types that could be present.
    tweet_text = pd.Series(data.tweet.head(20)).to_string() #It read the 20 values from the column and stored into variable tw
    print(type(tweet_text))
    print('\n')
    print(tweet_text) #printing the data
    print('\n')
```

```
<class 'str'>


0      @user when a father is dysfunctional and is s...
1     @user @user thanks for #lyft credit i can't us...
2                                  bihday your majesty
3     #model   i love u take with u all the time in ...
4               factsguide: society now    #motivation
5     [2/2] huge fan fare and big talking before the...
6      @user camping tomorrow @user @user @user @use...
7     the next school year is the year for exams.ð...
8     we won!!! love the land!!! #allin #cavs #champ...
9      @user @user welcome here !  i'm   it's so #gr...
10     â #ireland consumer price index (mom) climb...
11    we are so selfish. #orlando #standwithorlando ...
12    i get to see my daddy today!!   #80days #getti...
13    @user #cnn calls #michigan middle school 'buil...
14    no comment!  in #australia   #opkillingbay #se...
15    ouch...junior is angryð#got7 #junior #yugyo...
16    i am thankful for having a paner. #thankful #p...
17                               retweet if you agree!
18    its #friday! ð smiles all around via ig use...
19    as we all know, essential oils are not made of...
```

```
[9] sentences = sent_tokenize(tweet_text) #using a build in function to tokenize tweet_text


    print ("# sentences: ", len(sentences))
    print('\n')


    sentences
```

```
# sentences:  10

["0      @user when a father is dysfunctional and is s...\n1     @user @user thanks for #lyft credit i can't us...\n2
 'love the land!!!',
 '#allin #cavs #champ...\n9      @user @user welcome here !',
 "i'm   it's so #gr...\n10     â\x86\x9d #ireland consumer price index (mom) climb...\n11    we are so selfish.",
 '#orlando #standwithorlando ...\n12    i get to see my daddy today!!',
 "#80days #getti...\n13    @user #cnn calls #michigan middle school 'buil...\n14    no comment!",
 'in #australia   #opkillingbay #se...\n15    ouch...junior is angryð\x9f\x98\x90#got7 #junior #yugyo...\n16    i am thankful for having
 '#thankful #p...\n17                                retweet if you agree!',
 '18    its #friday!',
 'ð\x9f\x98\x80 smiles all around via ig use...\n19    as we all know, essential oils are not made of...']
```

```
10]
    words = word_tokenize(tweet_text) #this will create a of list of each word. using build in fucntion

    #How many words are there? :
    print (len(words))
    print("\n")

    #Print words :
    print (words)
```

```
246

['0', '@', 'user', 'when', 'a', 'father', 'is', 'dysfunctional', 'and', 'is', 's', '...', '1', '@', 'user', '@', 'user', 'thanks', 'for',
```

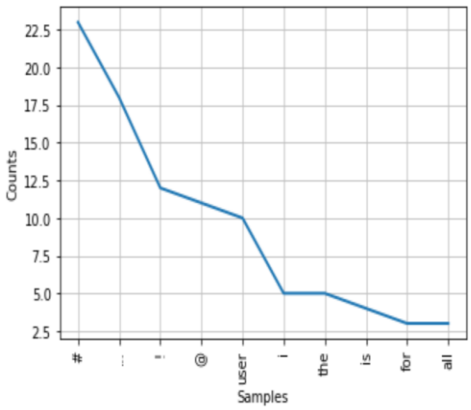
[11]

```
from nltk.probability import FreqDist #using pre build functino to see the distrubution


fdist = FreqDist(words) # this determine the frequency

fdist.most_common(10) #getting the most common
```

```
[('#', 23),
 ('...', 18),
 ('!', 12),
 ('@', 11),
 ('user', 10),
 ('i', 5),
 ('the', 5),
 ('is', 4),
 ('for', 3),
 ('all', 3)]
```

[12]

```
import matplotlib.pyplot as plt #plotting the data

fdist.plot(10)
```



Cleaing data Now i will show two ways to remove the uncessary character from the file.

The first demo

```
[13] import re
     tweet_text_2 = re.sub("[^A-Za-z]+"," ",tweet_text) # the character are removed and space i
     print(tweet_text_2)
     print('\n')
```

```
 user when a father is dysfunctional and is s user user thanks for lyft credit i can t us
```

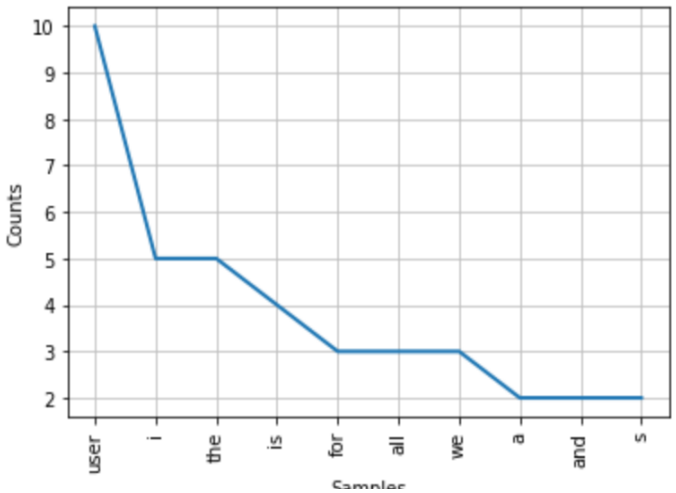
```
#Tokenize the text with words :
words_2 = word_tokenize(tweet_text_2) #passing in tweet_text_2

#How many words are there? :
print (len(words_2))
print("\n")

#Print words :
print (words_2)
```

```
151


['user', 'when', 'a', 'father', 'is', 'dysfunctional', 'and', 'is', 's', 'user', 'user', '
```

```
[15] #Import required libraries :
     from nltk.probability import FreqDist

     #Find the frequency :
     fdist_2 = FreqDist(words_2)

     #Print 10 most common words :
     fdist_2.most_common(10)
```

```
[('user', 10),
 ('i', 5),
 ('the', 5),
 ('is', 4),
 ('for', 3),
 ('all', 3),
 ('we', 3),
 ('a', 2),
 ('and', 2),
 ('s', 2)]
```

```
[16] #Plot the graph for fdist :
     import matplotlib.pyplot as plt

     fdist_2.plot(10)
```



second demo

An empty list is created and the words are stored. Then a for loop is used to check if all characters in each string
Z). Whitespace or any other character occurrence in the string would return false, but if there is a complete nume
NaN.

```
[17] #Empty list to store words:
     words_no_punc = []

     #Removing punctuation marks :
     for w in words:
         if w.isalpha():
             words_no_punc.append(w.lower())

     #Print the words without punctution marks :
     print (words_no_punc)

     print ("\n")

     #Length :
     print (len(words_no_punc))
```

```
['user', 'when', 'a', 'father', 'is', 'dysfunctional', 'and', 'is', 's', 'user', 'user', 't

143
```

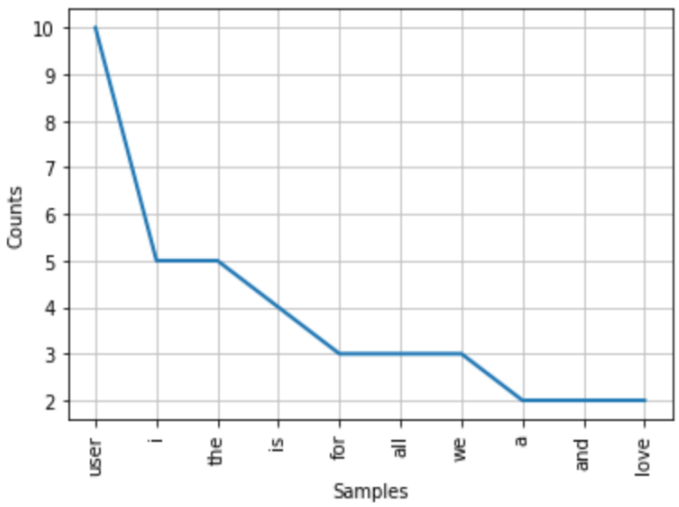
Now checking the frequency distribution

```
[18] #Frequency distribution :
     fdist = FreqDist(words_no_punc)

     fdist.most_common(10)
```

```
[('user', 10),
 ('i', 5),
 ('the', 5),
 ('is', 4),
 ('for', 3),
 ('all', 3),
 ('we', 3),
 ('a', 2),
 ('and', 2),
 ('love', 2)]
```

Now repeating the process again and show the graph analysis

```
[19]
     #Plot the most common words on grpah:

     fdist.plot(10)
```



link textNow I will be removing **stopwords**. Stopwords are words that are used frequently used in the english lang
meaning ...

Will need to import nltk. This give us easy access to a prebuild library which contains the stopwords. using this w
stopwords and check what is considered a stopword

```
[20] import nltk
     stopwords = nltk.corpus.stopwords.words("english")
     stopwords[0:10] #only looking at the first 10
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're"]
```

Now to remove the stopwords. A function is created

First will need a store the clean words. Then

```
[21] filtered_list = [] # creating a blank list which the filitered stopwords
     for word in words_no_punc: #will check for each word in words_no_punc
       if word not in stopwords:# check to see it is present or not then it wil be appened to t
         filtered_list.append(word)
```

```
[22] filtered_list #looking at the stopwords that have been removed.
```

```
 'year',
 'love',
 'land',
 'allin',
 'cavs',
 'champ',
 'user',
 'user',
 'welcome',
 'gr',
 'ireland',
 'consumer',
```

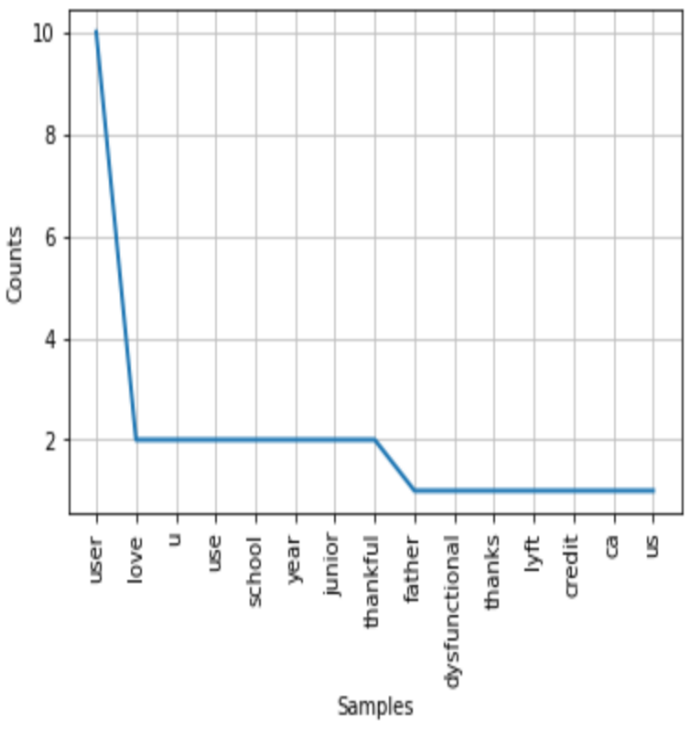
now looking at the distrbution

```
#Frequency distribution :
fdist = FreqDist(filtered_list)

fdist.most_common(10)
```

```
[('user', 10),
 ('love', 2),
 ('u', 2),
 ('use', 2),
 ('school', 2),
 ('year', 2),
 ('junior', 2),
 ('thankful', 2),
 ('father', 1),
 ('dysfunctional', 1)]
```

plotting the distribution

```
[24] fdist.plot(15)
```



Now word stemming. This method will allow us to bring the words to their root form.

```
25] from nltk.stem import PorterStemmer #Import stemming library
    porter_stemmer = PorterStemmer()
    input_data_test = "0      @user when a father is dysfunctional and is so selfish he drags
    input_data_test = nltk.word_tokenize(input_data_test)
    #input_data = sentences
    #input_data = nltk.word_tokenize(input_data[0])
    for word in input_data_test:
      print(porter_stemmer.stem(word))
```

```
0
@
user
when
a
father
is
dysfunct
and
is
so
selfish
he
drag
hi
kid
into
hi
dysfunct
.
'
```

```python
#now just passing in the sentence based on the index
from nltk.stem import PorterStemmer #Import stemming library
porter_stemmer = PorterStemmer()
input_data = sentences
input_data = nltk.word_tokenize(input_data[0])
for word in input_data:
  print(porter_stemmer.stem(word))
```

```
time
in
...
4
factsguid
:
societi
now
#
motiv
5
[
2/2
]
huge
fan
fare
and
big
talk
befor
the
...
6
@
user
camp
tomorrow
@
```

Now passing in the list of words

[27]
```
porter_stemmer_2 = PorterStemmer()

#Word-list for stemming :
word_list = words_no_punc

for word in word_list:
    print(porter_stemmer_2.stem(word))
```

```
user
when
a
father
is
dysfunct
and
is
s
user
user
thank
for
lyft
credit
i
ca
us
bihday
your
majesti
model
i
love
u
take
with
u
all
the
time
in
factsguid
```

Stemming. Wanted to see if there wre any variations in the output

This a test code to see how it works

```
from nltk import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
word_list = ["studies","leaving","decreases","plays", "playing","affrimation"]

for w in word_list:
    print(lemmatizer.lemmatize(w ,pos="v"))
```

```
study
leave
decrease
play
play
affrimation
```

Now using the list of words from the data

```
[29] from nltk import WordNetLemmatizer
     lemmatizer = WordNetLemmatizer()
     word_list2 = words_no_punc

     for word_ in word_list2:
         print(lemmatizer.lemmatize(word_ ,pos="v"))
```

```
to
see
my
daddy
today
getti
user
cnn
call
michigan
middle
school
no
```

pos tagging

```
[30] import nltk
     from nltk import word_tokenize, pos_tag
     nltk.download('universal_tagset')
```

```
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Package universal_tagset is already up-to-date!
True
```

```
[31]
     (sentences[0])
```

```
'0      @user when a father is dysfunctional and is s...\n1     @user @user thanks for #l
he time in ...\n4                factsguide: society now    #motivation\n5     [2/2] huge
is the year for exams.ð\x9f\x98...\n8     we won!!!'
```

```
[32] word_3 =word_tokenize(sentences[0])
     for words in word_3:
         tagged_words = nltk.pos_tag(word_3,tagset= 'universal')
     tagged_words
```

```
 ('time', 'NOUN'),
 ('in', 'ADP'),
 ('...', '.'),
 ('4', 'NUM'),
 ('factsguide', 'NOUN'),
 (':', '.'),
 ('society', 'NOUN'),
 ('now', 'ADV'),
 ('#', '.'),
 ('motivation', 'NOUN'),
 ('5', 'NUM'),
 ('[', 'NOUN'),
 ('2/2', 'NUM'),
 (']', 'ADJ'),
 ('huge', 'ADJ'),
 ('fan', 'NOUN'),
```

TDIDF

```
[33] from sklearn.feature_extraction.text import TfidfVectorizer

     sentences_2 = data # this was used previously and has the special charcater remvoed
     vectorizer = TfidfVectorizer(norm = None) #Create an object :

     X = vectorizer.fit_transform(sentences_2).toarray() #Generating output for TF_IDF :


     print(vectorizer.vocabulary_)#Total words with their index in model :
     print("\n")

     print(vectorizer.get_feature_names())
     print("\n")

     #Show the output :
     print(X)
```

```
{'label': 0, 'tweet': 1}


['label', 'tweet']


[[1.40546511 0.        ]
 [0.         1.40546511]]
```

Decided to try something different here to understand some of the conepts and do some testing on the data.

```
[34] import pandas as pd
     import numpy as np
```

```
[35] df = pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_Analysis/master
```

```
[36] df.head(5)
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

```
[37] df['label'].value_counts()
```

```
0    29720
1     2242
Name: label, dtype: int64
```

```
[38] ! pip install git+https://github.com/laxmimerit/preprocess_kgptalkie.git
```

```
Collecting git+https://github.com/laxmimerit/preprocess_kgptalkie.git
  Cloning https://github.com/laxmimerit/preprocess_kgptalkie.git to /tmp/pip-req-build-ans2q
  Running command git clone -q https://github.com/laxmimerit/preprocess_kgptalkie.git /tmp/p
Requirement already satisfied (use --upgrade to upgrade): preprocess-kgptalkie==0.1.0 from g
Building wheels for collected packages: preprocess-kgptalkie
```

```
[41] df['tweet'] = df['tweet'].apply(lambda x: get_clean(x))
```

```
[42] df.head(5)
```

	id	label	tweet
0	1	0	youser when a father is dysfyounctional and is...
1	2	0	youser youser thanks for lyfeatyouring credit ...
2	3	0	bihday yoyour majesty
3	4	0	model i love you take with you all the time in...
4	5	0	factsgyouide soriginal contentthat isty now mo...

TFDIF and predicting

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import LinearSVC
from sklearn.metrics import classification_report
```

```
[44] tfidf = TfidfVectorizer(max_features=2000,ngram_range=(1,3), analyzer='char')
```

```
[45] X = tfidf.fit_transform(df['tweet'])
```

```
[46] y = df['label']
```

```
[47] X.shape
```

```
(31962, 2000)
```

```
[48] X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2, random_state=0)
```

```
[49] clf = LinearSVC()
     clf.fit(X_train, y_train)
```

```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
          verbose=0)
```

```
[50] y_pred = clf.predict(X_test)
```

```
[51] print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      5985
           1       0.81      0.48      0.60       408

    accuracy                           0.96      6393
   macro avg       0.89      0.74      0.79      6393
```

Doing a simple check to see if there are any missing values. I will use isnull() to do that.

```
[52] df.isnull() #checking to see if there are any missing values.
```

	id	label	tweet
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
31957	False	False	False
31958	False	False	False
31959	False	False	False
31960	False	False	False
31961	False	False	False

31962 rows × 3 columns

```
[53] df.isnull().sum() #this checks the number of missing values in each column. Converts trues to 1s and 0s fa
```

```
id       0
label    0
tweet    0
dtype: int64
```

```
[54] df[df.label.isnull()] # to see if there are any missing values, and there are none.
```

	id	label	tweet

+ Code + Text

*Data Visualization and analysis *

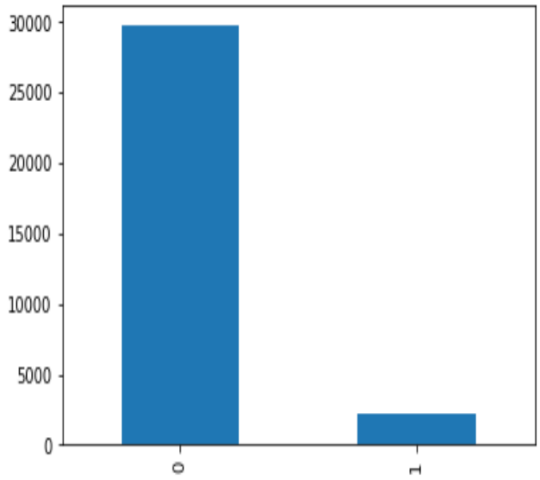
```
#Library to form wordcloud :
from wordcloud import WordCloud, ImageColorGenerator
import requests
import numpy as np
from PIL import  *


#Library to plot the wordcloud :
import matplotlib.pyplot as plt

# combining the image with the dataset
Mask = np.array(Image.open(requests.get('http://clipart-library.com/image_gallery2/Twitter-PNG-Image.png', stream=True).ra

# We use the ImageColorGenerator library from Wordcloud
# Here we take the color of the image and impose it over our wordcloud
image_colors = ImageColorGenerator(Mask)



#Generating the wordcloud :
wordcloud = WordCloud(background_color='black', height=1500, width=4000,mask=Mask).generate(tweet_text_2) #important to pr

#Plot the wordcloud :
plt.figure(figsize = (12, 12))
plt.imshow(wordcloud)

#To remove the axis value :
plt.axis("off")
plt.show()
```

```
[58] df['label'].value_counts() #counting the 0 and 1 for the label
```

```
0    29720
1     2242
Name: label, dtype: int64
```

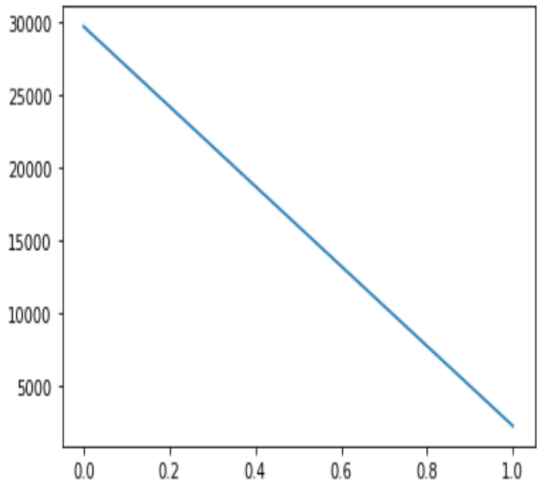
```
[59] df['label'].value_counts().plot.bar() #ploting the data
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f29fd37ef28>
```




```
[60] df['label'].value_counts().plot.line() # to see what a line graph would look like
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f29fd1ff470>
```



e. Output file link if applicable

f. Video link (YouTube or any other publicly available video platform)

The link should. I opened the access where anyone with the link can view It. If counter any issues please let me know and I will change the setting to where you can view it. Thank you.

https://umkc.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=fab7e9f4-a88c-4249-9f6c-ac37002bf092

g. Any inside about the data or the ICP in general

I really enjoyed working on this ICP. The lecture was very helpful in understanding the core concepts, which allowed me better grasp the topics like stemming, tokenization, and few other discussed for this ICP. I think the due dates should be extended to this timeframe for future ICPs, this allows me to better understand the data and method I am working with. I talked to some of my classmates about this and they think the same. Just a consideration thought I add. Thank you