

CS5542 Big Data Apps and Analytics

In Class Programming –3
11th September 2020

Submit ICP Feedback in Class. : [Lnk to Feed back Form](#)

NLP:

Use the same data (that we obtained by in source code

`Data = pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_Analysis/master/train.csv')`) and perform the sentiment analysis task on this data using one of the scikit learn classifier for text.

ICP Requirements:

- 1) Data cleaning and preprocessing (at minimum have the following: Removing unnecessary columns or data, Removing Twitter Handles(@user), Removing punctuation, numbers, special characters, Removing stop words, Tokenization, and Stemming, TFIDF vectors, POS tagging, checking for missing values , train/test split of data). (70 points)
- 2) Data Visualization and analysis for critical steps (WordCloud, Bar plots, etc) (10 points)
- 3) Model building and successfully executing the model to make prediction. (10 points)
- 4) Code quality, Pdf Report quality, video explanation (10 points)

Submission Guidelines:

Same as ICP 2.

ICP Report:

What I learned in the ICP:

I learned more about Data Frames and how need to think differently with dealing with immutable data. I continue to learn more about lambda function, but I feel I need to spend some time and focus on them because of how they are leveraged with transforming the RDDs. I learn some about Natural Language processing and some of the libraries that are available for Python.

Description of what task I was performing:

Use the given input file and perform tasks for cleaning analyzing the data.

Challenges I faced:

Had to figure out how to make the Bag of Words without crashing CoLab.

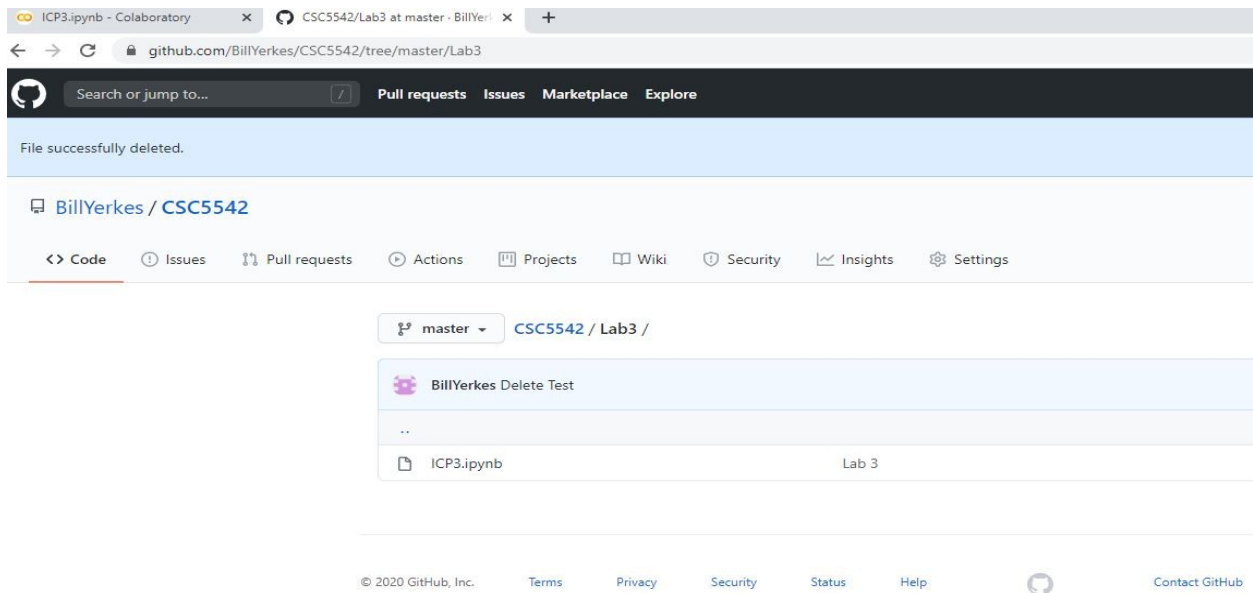
Figuring out how to update the Data Frame after cleaning up the data.

Figuring out how to clean the data so as to reduce the size of the Bag of Words.

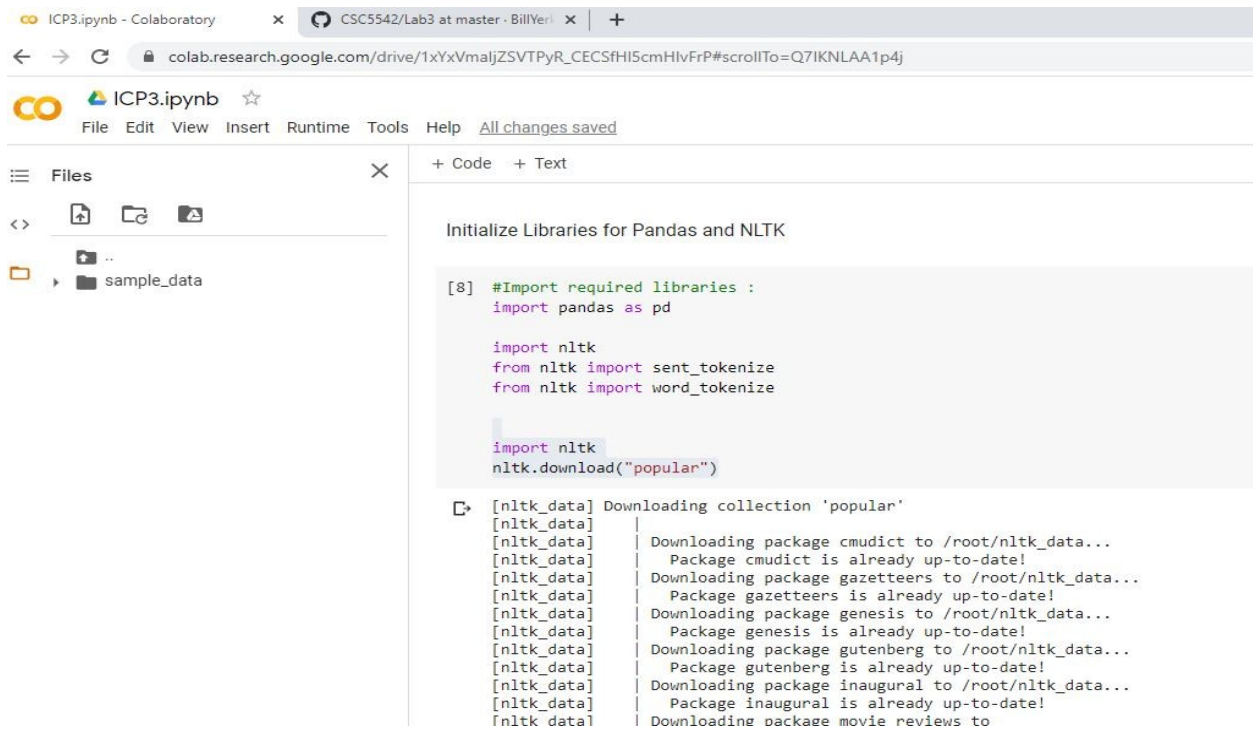
Figuring out how to get the Word Cloud to work, re-watching the lecture was a big help.

Screen Shots

GitHub Repository



Initialize Libraries for Pandas and NLTK



Init DataFrame

Read the CSV file with the Data from the Cloud

Reduce the size of the Dataframe, because of the issue with running out of RAM.

DataFrame Properties

Display portions of the DataFrame for visual inspection of the Data

Formally, given sample of tweets and labels, where label ‘1’ denotes the tweet is racist/sexist and label ‘0’ denotes the tweet is not racist/sexist.

id : The id associated with the tweets in the given dataset.

tweets : The tweets collected from various sources and having either positive or negative sentiments associated with it.

label : A tweet with label '0' is of positive sentiment while a tweet with label '1' is of negative sentiment

ICP3.ipynb - Colaboratory

CSCS542/Lab3 at stoner · BillYer · x +

→ ↻ 🔒 colab.research.google.com/drive/1xYxVmaljZSVTPyR_CECSfHl5cmHlvFrP#scrollTo=Q7IKNLAA1pj4

ICP3.ipynb ☆
File Edit View Insert Runtime Tools Help All changes saved

Files
 sample_data

+ Code + Text

```
[nltk_data] Done downloading collection popular
True

Read the CSV file with the Data from the Cloud

Reduce the size of the DataFrame, because of the issue with running out of RAM


[9] #get the Data used :
Data = pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_Analysis/master/train.csv')

Data2 = Data.sample(frac=0.1, replace=True)



Display portions of the DataFrame for visual inspection of the Data

Data Set Description:

Formally, given sample of tweets and labels, where label '1' denotes the tweet is racist/sextist and label '0' denotes the tweet is n
id : The id associated with the tweets in the given dataset.

tweets : The tweets collected from various sources and having either positive or negative sentiments associated with it.

label : A tweet with label '0' is of positive sentiment while a tweet with label '1' is of negative sentiment


▶ Data2
```

	id	label	tweet
20045	20046	1	white nationalist leader reveals 5 of his most...
19986	19987	0	@user yours was better because on the road to ...
17777	17778	0	anton yelchin rip dude #star #starek #trek #st...
25	26	0	beautiful sign by vendor 80 for \$45.00!! #ups!
15264	15265	0	holiday in 2 days 🎅🏠🍷🥂🍰👯🧖🌸🔪💰\$...
...
18465	18466	0	@user apparently he's not. @user

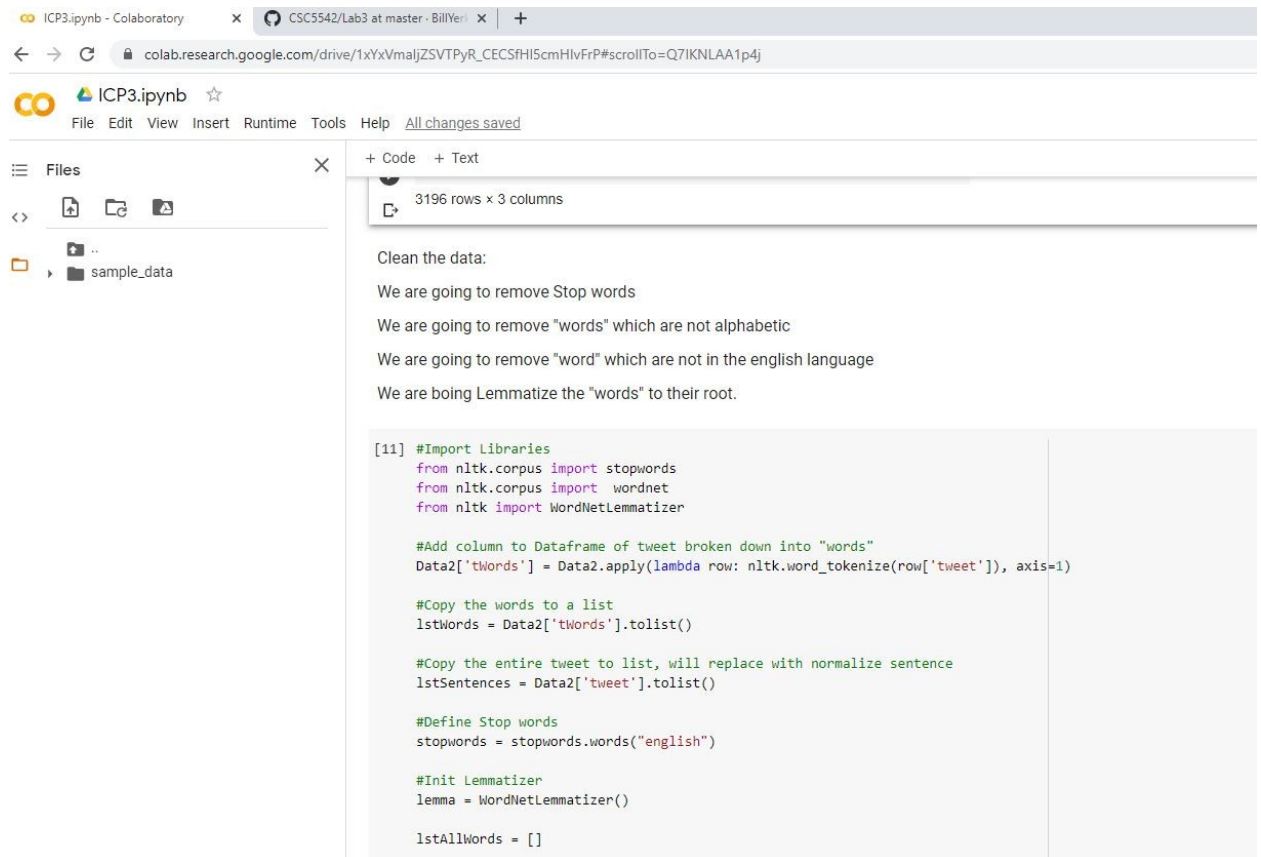
Clean the data:

We are going to remove Stop words

We are going to remove "words" which are not alphabetic

We are going to remove "word" which are not in the english language

We are going to Lemmatize the "words" to their root.



The screenshot shows a Google Colaboratory notebook interface. The browser address bar displays the URL: `colab.research.google.com/drive/1xYxVmaljZSVTPyR_CECsfHl5cmHlvFrP#scrollTo=Q7IKNLAA1p4j`. The notebook is titled "ICP3.ipynb" and has a menu bar with options: File, Edit, View, Insert, Runtime, Tools, Help, and [All changes saved](#). On the left, a file explorer shows a folder named "sample_data". The main area contains a code cell with the following text and code:

```
[11] #Import Libraries
      from nltk.corpus import stopwords
      from nltk.corpus import wordnet
      from nltk import WordNetLemmatizer

      #Add column to Dataframe of tweet broken down into "words"
      Data2['tWords'] = Data2.apply(lambda row: nltk.word_tokenize(row['tweet']), axis=1)

      #Copy the words to a list
      lstWords = Data2['tWords'].tolist()

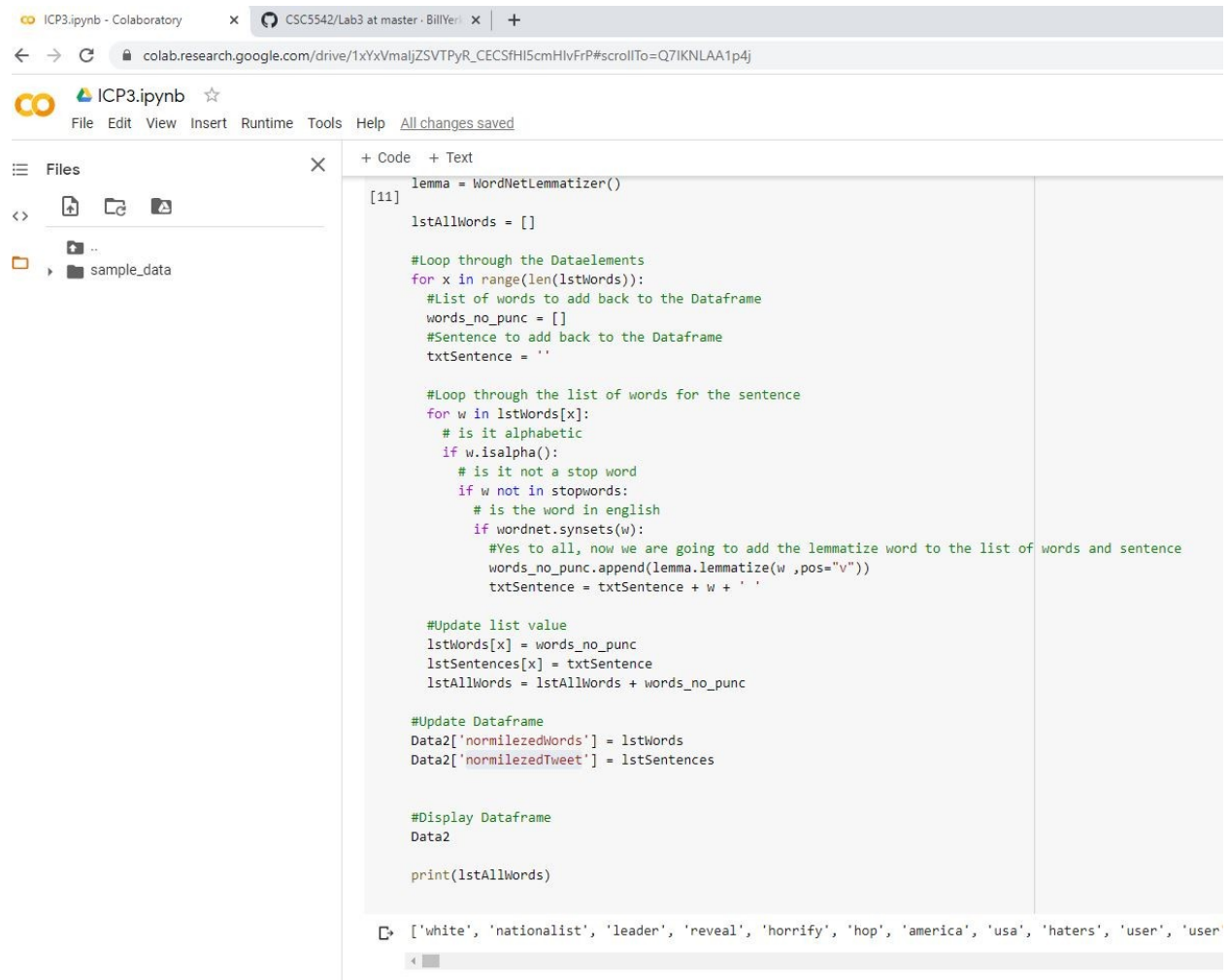
      #Copy the entire tweet to list, will replace with normalize sentence
      lstSentences = Data2['tweet'].tolist()

      #Define Stop words
      stopwords = stopwords.words("english")

      #Init Lemmatizer
      lemma = WordNetLemmatizer()

      lstAllWords = []
```

Looping through the Data and cleaning it



The screenshot shows a Google Colaboratory notebook interface. The top bar includes the Colaboratory logo, the notebook name 'ICP3.ipynb', and a star icon. Below the bar is a menu with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. The left sidebar shows a file explorer with a folder named 'sample_data'. The main area is a code editor with a Python script. The script uses the WordNetLemmatizer from nltk to clean a list of words. It iterates through the list, removing non-alphabetic characters, stop words, and words not in English. The cleaned words are then used to create a DataFrame with columns 'normalizedWords' and 'normalizedTweet'. The final output is a list of cleaned words: ['white', 'nationalist', 'leader', 'reveal', 'horrify', 'hop', 'america', 'usa', 'haters', 'user', 'user'].

```
[11] lemma = WordNetLemmatizer()

lstAllWords = []

#Loop through the Dataelements
for x in range(len(lstWords)):
    #List of words to add back to the Dataframe
    words_no_punc = []
    #Sentence to add back to the Dataframe
    txtSentence = ''

    #Loop through the list of words for the sentence
    for w in lstWords[x]:
        # is it alphabetic
        if w.isalpha():
            # is it not a stop word
            if w not in stopwords:
                # is the word in english
                if wordnet.synsets(w):
                    #Yes to all, now we are going to add the lemmatize word to the list of words and sentence
                    words_no_punc.append(lemma.lemmatize(w ,pos="v"))
                    txtSentence = txtSentence + w + ' '

    #Update list value
    lstWords[x] = words_no_punc
    lstSentences[x] = txtSentence
    lstAllWords = lstAllWords + words_no_punc

#Update Dataframe
Data2['normalizedWords'] = lstWords
Data2['normalizedTweet'] = lstSentences

#Display Dataframe
Data2

print(lstAllWords)
```

['white', 'nationalist', 'leader', 'reveal', 'horrify', 'hop', 'america', 'usa', 'haters', 'user', 'user']

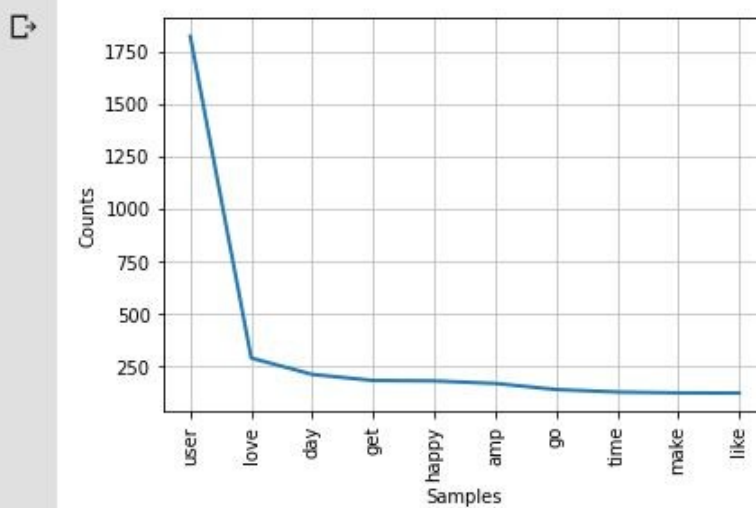
Show the distribution of the 10 most common words

Show the distribution of the 10 most common words

```
[14] #Import required libraries :  
      from nltk.probability import FreqDist  
  
      #Find the frequency :  
      fdist = FreqDist(lstAllWords)  
  
      #Print 10 most common words :  
      fdist.most_common(10)
```

```
[('user', 1821),  
 ('love', 287),  
 ('day', 209),  
 ('get', 180),  
 ('happy', 178),  
 ('amp', 166),  
 ('go', 137),  
 ('time', 125),  
 ('make', 121),  
 ('like', 120)]
```

```
▶ fdist.plot(10)
```



Make a Word Cloud

Make A word cloud

```
[18] #Library to form wordcloud .
import numpy as np
from wordcloud import WordCloud, ImageColorGenerator
import requests
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).raw))

# 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)

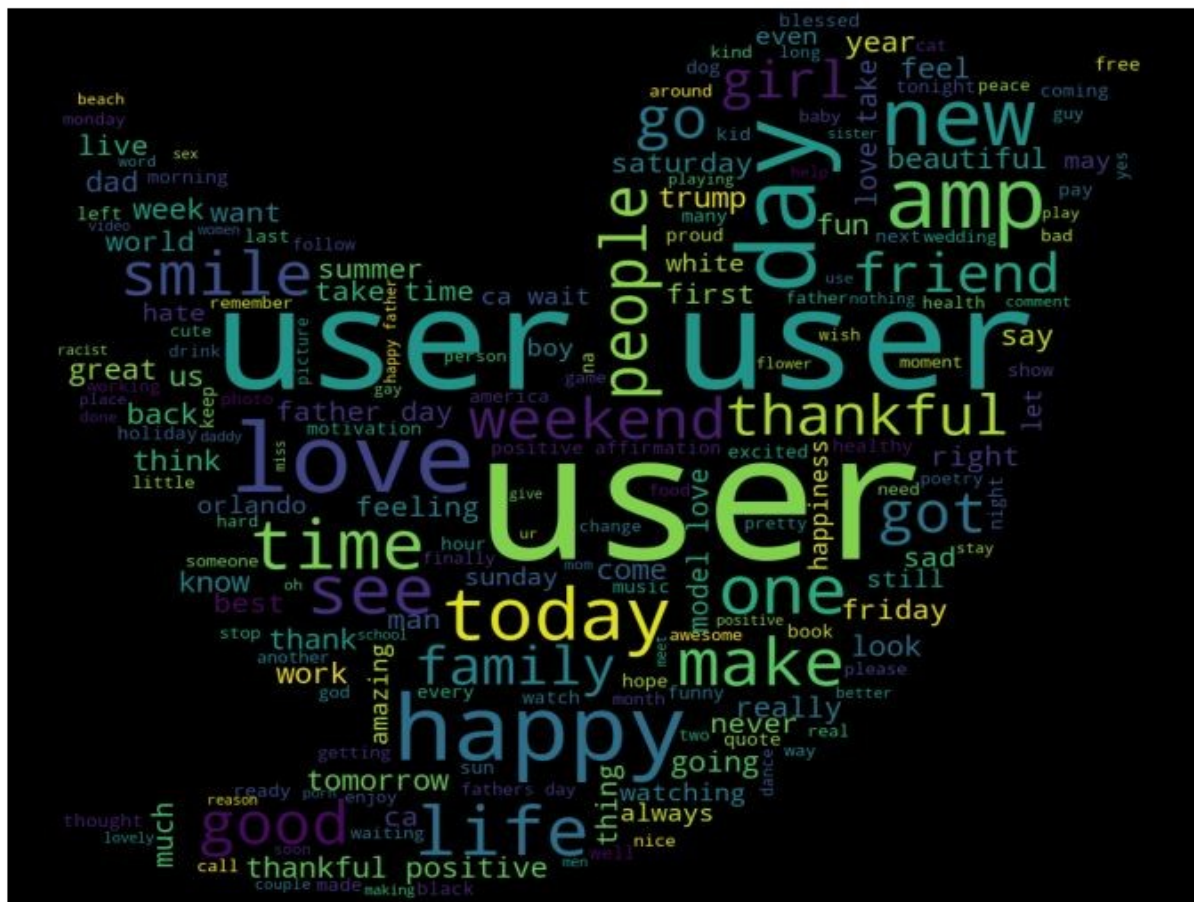
text = pd.Series(Data2.normalizedTweet).to_string()

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

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

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

Word Cloud



Make a Bag of Words

Make Bag of Words

This part here is memory intensive and will consume all of the RAM if the data set is too large and crash Co Lab.

```
[20] from sklearn.feature_extraction.text import CountVectorizer

#Text for analysis :

sentences = Data2['normilezedTweet'].tolist()

#Create an object :
cv = CountVectorizer()

#Generating output for Bag of Words :
B_O_W = cv.fit_transform(sentences).toarray()
```

Run the Test on the Data

Run the Test on the Data

```
[21] #Import svm model
from sklearn.model_selection import train_test_split
from sklearn import svm

X_train, X_test, y_train, y_test = train_test_split(B_O_W, Data2.label, test_size=0.3, random_state=109) # 70% training and 30% test

#Create a svm Classifier
clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics

# Model Accuracy: how often is the classifier correct?
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

➡ Accuracy: 0.9395203336809176

[Video Link](#)

Any in site about the data or the ICP in general

Data consisted of Tweet text, with the text being categorized as racist or not racist.

CoLab is great. The Python libraries for NLP seem very powerful.

The code did not run as fast as expected and was taken by surprised by running out of RAM while trying to figure out the lab.