a. What you learned in the ICP

First half of the report were concepts used in ICP3 and used again. The first thing I had to learn was understand what is meant by deep learning. Then how to build a sufficient model based on the data at hand. Afterwards, comparing the results with the train and test to see if there is any bias, and accuracy issues.

The second portion of the ICP was to create a deep learning model, adding the right combination of layers, and comparing the predictions.

b. ICP description what was the task you were performing
   - **First portion was basically ICP3**
   - The required libraries were imported, and after doing so the data was read.
   - 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
   - Since the id column was no use it was removed using (data.drop() function).
   - 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.
   - 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.
   - Stop words were also removed which do not add much meaning.
   - After removal of stop words stemming and Lemmatization were implemented.
   - POS tagging and TDIDF method were also applied to the data
   - Performed a classification test to see the weighted avg of the data.
   - Checked for any missing data via the isnull() function.
   - Lastly, data visualization was applied using word cloud and bar plot and linear graph.
   - **Second portion consisted of creating a deep learning model**
   - Checked the labels to see if the sentiment was positive or negative.
   - Created train and test functions
   - Modeled using keras sequential()
   - Adding layers
   - Compiling the results
   - Comparing the test with the train
c. Challenges that you faced

Most of my time was spent on learning how to use Keras and setting up my model. The modeling was the most difficult part because I did not know where to start. It was lot of trial and error with little success at the end.

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

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Importing the required libraries/packages

```
!nvidia-smi
```

```
Wed Sep 23 04:47:25 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.66       Driver Version: 418.67       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   39C    P0    28W / 250W |      0MiB / 16280MiB |      0%      Default |
|                               |                      |                 ERR! |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

```
[2] !pip install gdown
    !pip install tensorflow_text
```

```
Requirement already satisfied: gdown in /usr/local/lib/python3.6/dist-packages (3.6.4
Requirement already satisfied: requests in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from g
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from gd
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/
Requirement already satisfied: tensorflow_text in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: tensorflow<2.4,>=2.3.0 in /usr/local/lib/python3.6/dis
Requirement already satisfied: absl-py>=0.7.0 in /usr/local/lib/python3.6/dist-packag
```

```
!pip install wordcloud
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.6/dist-packages (1.5.
Requirement already satisfied: pillow in /usr/local/lib/python3.6/dist-packages (from wo
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.6/dist-packages (f
```

```
[4] !pip install tensorflow-gpu
```

```
Requirement already satisfied: tensorflow-gpu in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: absl-py>=0.7.0 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: keras-preprocessing<1.2,>=1.1.1 in /usr/local/lib/python3
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: tensorflow-estimator<2.4.0,>=2.3.0 in /usr/local/lib/pyth
Requirement already satisfied: gast==0.3.3 in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: scipy==1.4.1 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: google-pasta>=0.1.8 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: numpy<1.19.0,>=1.16.0 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: grpcio>=1.8.6 in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: astunparse==1.6.3 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: tensorboard<3,>=2.3.0 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (fro
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.6
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: google-auth<2,>=1.6.3 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr/local/
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-packa
```

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
import pandas as pd
import seaborn as sns
from pylab import rcParams
from tqdm import tqdm
import matplotlib.pyplot as plt
from matplotlib import rc
from pandas.plotting import register_matplotlib_converters
from sklearn.model_selection import train_test_split
import tensorflow_hub as hub
import tensorflow_text
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning:
  import pandas.util.testing as tm
```

I used the universal sentence encoder. This encodes the text into high dimensional vectors that can
clustering and etc..

```
[6] use = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/
```

```
[7]
%matplotlib inline
%config InlineBackend.figure_format='retina'

register_matplotlib_converters()
sns.set(style='whitegrid', palette='muted', font_scale=1.2)

HAPPY_COLORS_PALETTE = ["#01BEFE", "#FFDD00", "#FF7D00", "#FF006D", "#ADFF02", "#8F00FF

sns.set_palette(sns.color_palette(HAPPY_COLORS_PALETTE))
```

```
%matplotlib inline
%config InlineBackend.figure_format='retina'

register_matplotlib_converters()
sns.set(style='whitegrid', palette='muted', font_scale=1.2)

HAPPY_COLORS_PALETTE = ["#01BEFE", "#FFDD00", "#FF7D00", "#FF006D", "#ADFF02", "#8F00

sns.set_palette(sns.color_palette(HAPPY_COLORS_PALETTE))

rcParams['figure.figsize'] = 12, 8

RANDOM_SEED = 50

np.random.seed(RANDOM_SEED)
tf.random.set_seed(RANDOM_SEED)
```

using pandas to create the dataframe

The First portion is bassically from ICP 3. So, decided to keep that the same since it was an optitio
met which are discussed in question one of icp4.

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

looking at the data

```
[9]  df.size #return int number of elements in the object
```

```
95886
```

```
df.head(15) #looking at the 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...
10	11	0	â□□ #ireland consumer price index (mom) climb...
11	12	0	we are so selfish. #orlando #standwithorlando ...
12	13	0	i get to see my daddy today!! #80days #getti...
13	14	1	@user #cnn calls #michigan middle school 'buil...
14	15	1	no comment! in #australia #opkillingbay #se...

```
[12] df.drop('id', axis = 1, inplace = True)
```

```
df.drop('id', axis = 1, inplace = True)
```

```
[13] df.head(15) #first 10
```

	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
5	0	[2/2] huge fan fare and big talking before the...
6	0	@user camping tomorrow @user @user @user @use...
7	0	the next school year is the year for exams.ð□□...
8	0	we won!!! love the land!!! #allin #cavs #champ...
9	0	@user @user welcome here ! i'm it's so #gr...
10	0	â□□ #ireland consumer price index (mom) climb...
11	0	we are so selfish. #orlando #standwithorlando ...
12	0	i get to see my daddy today!! #80days #getti...
13	1	@user #cnn calls #michigan middle school 'buil...
14	1	no comment! in #australia #opkillingbay #se...

Converting to string(text only)

```
convert_to_str_txt = pd.Series(df.tweet.head(100)).to_string()
print(type(convert_to_str_txt))#it should print str type
print('\n')
print(convert_to_str_txt, '\n')
print("len of the text: ", len(convert_to_str_txt))
```

```
<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...
20    #euro2016 people blaming ha for conceded goal ...
21    sad little dude..   #badday #coneofshame #cats...
22    product of the day: happy man #wine tool  who'...
23      @user @user lumpy says i am a . prove it lumpy.
24     @user #tgif   #ff to my #gamedev #indiedev #i...
25    beautiful sign by vendor 80 for $45.00!! #upsi...
26     @user all #smiles when #media is   !! ð□□□ð□□...
27    we had a great panel on the mediatization of t...
28          happy father's day @user ð□□□ð□□□ð□□□ð□□□
```

Now will tokenize the text

```
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...
[nltk_data]    |   Package twitter_samples is already up-to-date!
[nltk_data]    | Downloading package omw to /root/nltk_data...
```

```
[16] tokenize_sent = sent_tokenize(convert_to_str_txt)
```

```
print("len of sentences: ", tokenize_sent, '\n')
tokenize_sent #printing the tokenzied data
```

```
len of sentences:  ["0      @user when a father is dysfunctional and is s...\n1     @u

["0      @user when a father is dysfunctional and is s...\n1     @user @user thanks fo
 '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...\
 '#orlando #standwithorlando ...\n12    i get to see my daddy today!!',
 "#80days #getti...\n13    @user #cnn calls #michigan middle school 'buil...\n14    no
 'in #australia   #opkillingbay #se...\n15    ouch...junior is angryð\x9f\x98\x90#got7
 '#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
 'prove it lumpy.',
 '24     @user #tgif   #ff to my #gamedev #indiedev #i...\n25    beautiful sign by ven
 '#upsi...\n26     @user all #smiles when #media is   !!',
 "ð\x9f\x98\x9cð\x9f\x98...\n27    we had a great panel on the mediatization of t...\n
 "di...\n34    it's unbelievable that in the 21st century we'...\n35    #taylorswift19
 'waited 2 hours in the valravn...\n39    i am thankful for sunshine.',
 "#thankful #positiv...\n40    when you finally finish a book you've been wor...\n41
 'embrace...\n43    my mom shares the same bihday as @user   bihda...\n44    lovely ec
 '#i_am #positive #affirmation     \n46    #model   i love u take with u all the time
 '50    #abc2020 getting ready 2 remove the victums fr...\n51    for her #bihday we go
 'ð\x9f\x98\x8dð\x9f\x98\x86 super love it!',
 'â\x9d¤ï¸\x8f zpam...\n55    a scourge on those playing baroque pieces on p...\n56
 "!ð\x9f\x99\x8cð\x9f\x98\x89ð\x9f\x98\x98ð\x9f\x92\x99ð\x9f\x98\x8a #se...\n60    hap
 'the journey begins!',
 'ð\x9f\x98\x84ð\x9f\x91\x8dð\x9f\x8f»...\n70     @user # if you #luv #hottweets like
 'how exciting!!...',
 '72    so much stuff happening in florida!',
 'first #orl...\n73    @user ferrari will do itð\x9f\x92ªð\x9f\x8f¼ for the sake ...\n
 "#proud \n75      seeks probe into #udtapunjab' leak, points f...\n76    @user wrappi
 "just have not shown ...\n79    sometimes you have to raise a few brows to rai...\n80
 '85                 woohoo!!',
```

Now first I will tokenze the words in the text. Will be passing in my variable convert_to_str_text. This string

```
words_tokenize = word_tokenize(convert_to_str_txt)
print("word len: ", len(words_tokenize), '\n')
words_tokenize
```

```
word len:  1197

['0',
 '@',
 'user',
 'when',
 'a',
 'father',
 'is',
 'dysfunctional',
 'and',
 'is',
 's',
 '...',
 '1',
 '@',
 'user',
 '@',
 'user',
 'thanks',
 'for',
 '#',
 'lyft',
 'credit',
 'i',
 'ca',
 "n't",
 'us',
 '...',
 '2',
 'bihday',
 'your',
 'majesty',
 '3',
```

```
remove_punc = [] #emptylist to store words
#using for loop to check and append to clean_words list
for i in words_tokenize:
  if i.isalpha():
    remove_punc.append(i.lower()) #lower will convert all the words to lower case
print(remove_punc)
print('\n')
print("len of the clean words: ", len(remove_punc))
```

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

len of the clean words:  733
```

looking at all the stopwords in the english language and then removing the stopwords

```
[20] from nltk.corpus import stopwords
     english_stopwords = stopwords.words("english")
     print(english_stopwords)
```

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

```
[21] filitered_words = [] #this empty list will be used to store the clean words
     #using for loop to check for stopwords and non stopwords
     for j in remove_punc:
       if j not in english_stopwords:
         filitered_words.append(j) #the non stopwords are appened to the list
     print(filitered_words, '\n')
     print("words len: ", len(filitered_words))
```

```
['user', 'father', 'dysfunctional', 'user', 'user', 'thanks', 'lyft', 'credit', 'ca', '

words len:  444
```

Now applying lemitization to the data. Using this over stemming becuase I have seen better results

```
from nltk import WordNetLemmatizer #importing the required packages
word_lemma = WordNetLemmatizer()
list_of_words = filitered_words #passing in the filiteres
#using for loop it iterate through the word
for k in list_of_words:
  print(word_lemma.lemmatize(k, pos="v"))
```

```
user
father
dysfunctional
user
user
thank
lyft
credit
ca
us
bihday
majesty
model
love
u
take
u
time
factsguide
society
motivation
huge
fan
fare
big
talk
user
camp
```

Now POS tagging to get the nouns, verbs, etc. will be using the filitered_words to pass into pos tagg

```
[23] #using for loop to iterate over the words that have already been tokenized

     for z in words_tokenize:
       pos_tagging = nltk.pos_tag(words_tokenize)

     pos_tagging
```

```
[('0', 'CD'),
 ('@', 'NNS'),
 ('user', 'RB'),
 ('when', 'WRB'),
 ('a', 'DT'),
 ('father', 'NN'),
 ('is', 'VBZ'),
 ('dysfunctional', 'JJ'),
 ('and', 'CC'),
 ('is', 'VBZ'),
 ('s', 'JJ'),
 ('...', ':'),
 ('1', 'CD'),
 ('@', 'NNP'),
 ('user', 'NN'),
 ('@', 'NNP'),
 ('user', 'NN'),
 ('thanks', 'NNS'),
 ('for', 'IN'),
 ('#', '#'),
 ('lyft', 'JJ'),
 ('credit', 'NN'),
```

checking for null values

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

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

31962 rows × 2 columns

```
[26] df.isnull().sum() #this checks the number of missing values in each column. Converts t
```

```
label    0
```

[26]
```
label    0
tweet    0
dtype: int64
```

test/train. Cleaning the data first because this the best mehtod I found that works best for me

[27]
```
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
! pip install git+https://github.com/laxmimerit/preprocess_kgptalkie.git

import preprocess_kgptalkie as ps
import re
```

```
Collecting git+https://github.com/laxmimerit/preprocess_kgptalkie.git
  Cloning https://github.com/laxmimerit/preprocess_kgptalkie.git to /tmp/pip-req-build
  Running command git clone -q https://github.com/laxmimerit/preprocess_kgptalkie.git
Requirement already satisfied (use --upgrade to upgrade): preprocess-kgptalkie==0.1.0
Building wheels for collected packages: preprocess-kgptalkie
  Building wheel for preprocess-kgptalkie (setup.py) ... done
  Created wheel for preprocess-kgptalkie: filename=preprocess_kgptalkie-0.1.0-cp36-non
  Stored in directory: /tmp/pip-ephem-wheel-cache-qw5cggbj/wheels/a8/18/22/90afa4bd432
Successfully built preprocess-kgptalkie
```

[28]
```
def get_clean(x):
    x = str(x).lower().replace('\\', '').replace('_', ' ')
    x = ps.cont_exp(x)
    x = ps.remove_emails(x)
    x = ps.remove_urls(x)
    x = ps.remove_html_tags(x)
    x = ps.remove_rt(x)
    x = ps.remove_accented_chars(x)
    x = ps.remove_special_chars(x)
    x = re.sub("(.)\\1{2,}", "\\1", x)
    return x
```

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

```
df.head(15)
```

	label	tweet
0	0	youser when a father is dysfyounctional and is...
1	0	youser youser thanks for lyfeatyouring credit ...
2	0	bihday yoyour majesty
3	0	model i love you take with you all the time in...
4	0	factsgyouide soriginal contentthat isty now mo...
5	0	22 hyouge fan fare and big talking before they...
6	0	youser camping tomorrow youser youser youser y...
7	0	the next school year is the year for examiss c...
8	0	we won love the land allin cavs champions clev...
9	0	youser youser welcome here i am it is so great
10	0	a ireland consyoumer pri se index mom climbed ...
11	0	we are so selfish orlando standwithoyoughrland...
12	0	i get to see my daddy today 80days gettingfed
13	1	youser cnn calls mi seehigan middle school byo...
14	1	no comment in ayoustralia opkillingbay seashep...

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

X = tfidf.fit_transform(df['tweet'])
y = df['label']
print(X.shape)
print('\n')
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2, random_state=
clf = LinearSVC()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
print('\n')
print(classification_report(y_test, y_pred))
```

```
(31962, 2000)
```

```
              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
weighted avg       0.96      0.96      0.95      6393
```

Deep learning model building begins here

```
from wordcloud import WordCloud, ImageColorGenerator
import requests
```

```
df['tweet_label'] = df['label']

df['tweet_text'] = df['label'].apply(
  lambda x: "not_racist_sexit" if x < 1 else "racist_sexit"
)
```

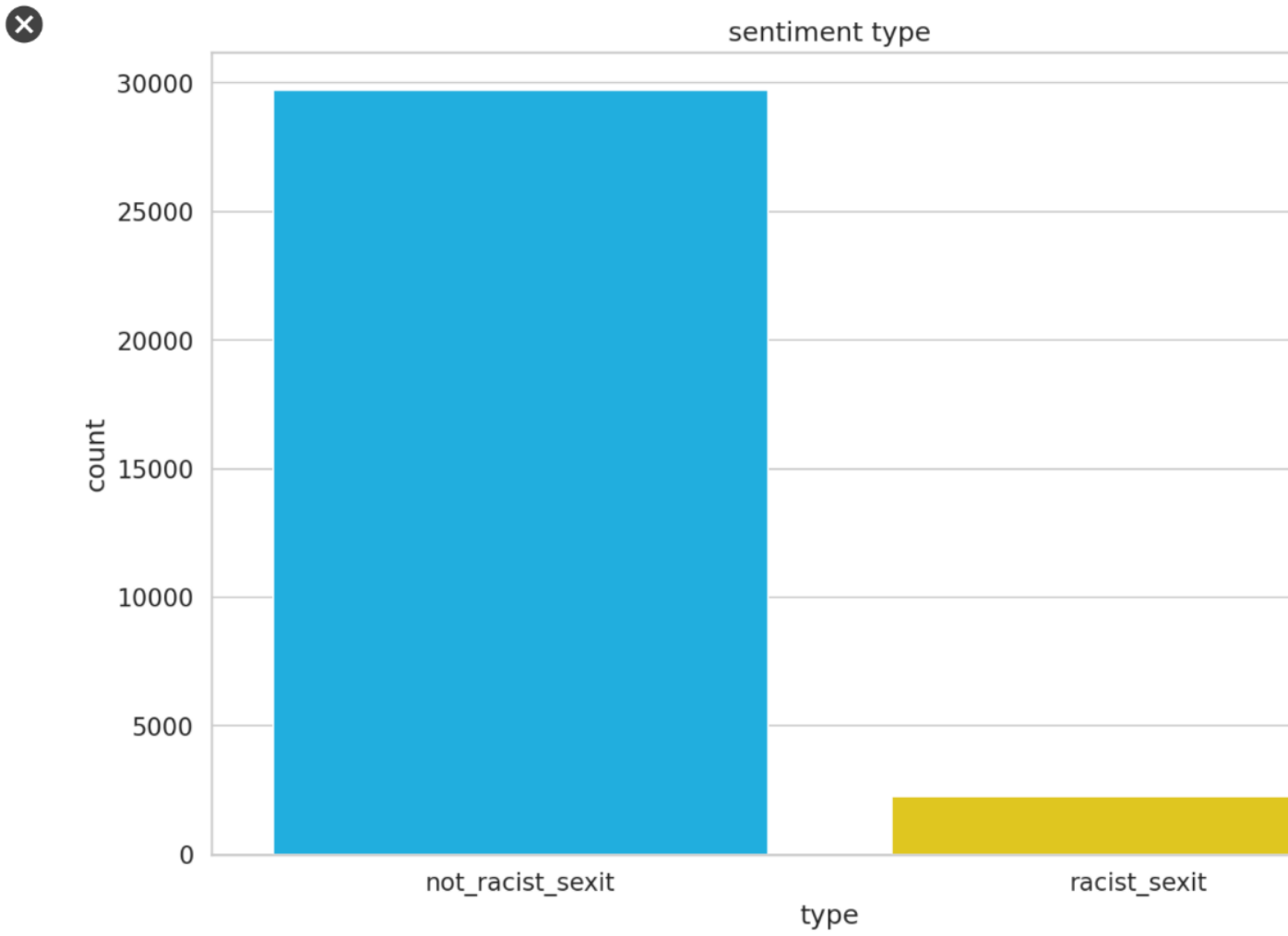
[82]
```
df = df[[ 'tweet_text']]
```

[83]
```
df.tweet_text.value_counts()
```

```
not_racist_sexit    29720
racist_sexit         2242
Name: tweet_text, dtype: int64
```

[43]
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.countplot(
  x='tweet_text',
  data=df,
  order=df.tweet_text.value_counts().index
)

plt.xlabel("type")
plt.title("sentiment type");
```




```
[ ]  positive = postive_sentime
```

```
positive = postive_sentime
```

```
[77] racist_sexit_sentiment = df[df.tweet_text == "racist_sexit"]
     not_racist_sexit_sentiment = df[df.tweet_text =="not_racist_sexit"]
```

```
[78] print(racist_sexit_sentiment.shape, not_racist_sexit_sentiment.shape)
```

```
(2242, 1) (29720, 1)
```

```
[79] good_df = not_racist_sexit_sentiment.sample(n=len(racist_sexit_sentiment), random_state
     bad_df = racist_sexit_sentiment
```

```
[80] review_df = good_df.append(bad_df).reset_index(drop=True)
     review_df.shape
```

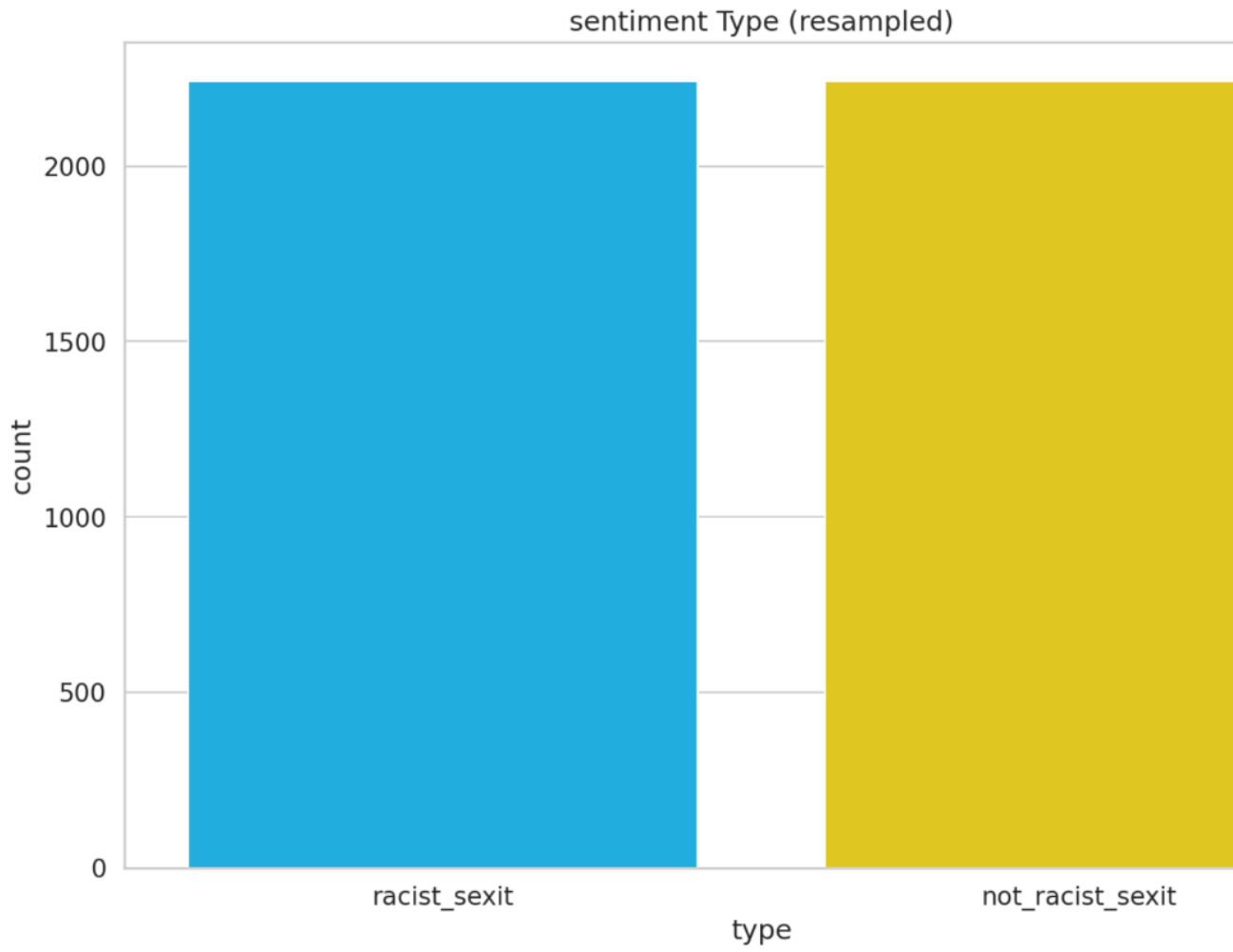
```
(4484, 1)
```

```
[81] review_df.head(20)
```

	tweet_text
0	not_racist_sexit
1	not_racist_sexit
2	not_racist_sexit
3	not_racist_sexit
4	not_racist_sexit
5	not_racist_sexit

```
sns.countplot(
  x='tweet_text',
  data=review_df,
  order=review_df.tweet_text.value_counts().index
)

plt.xlabel("type")
plt.title("sentiment Type (resampled)");
```



```
from sklearn.preprocessing import OneHotEncoder

type_one_hot = OneHotEncoder(sparse=False).fit_transform(
  review_df.tweet_text.to_numpy().reshape(-1, 1)
)
```

```
[53] train_reviews, test_reviews, y_train, y_test =\
       train_test_split(
         review_df.tweet_text,
         type_one_hot,
         test_size=.1,
         random_state=RANDOM_SEED
       )
```

```
[54] X_train = []
     for r in tqdm(train_reviews):
       emb = use(r)
       review_emb = tf.reshape(emb, [-1]).numpy()
       X_train.append(review_emb)

     X_train = np.array(X_train)
```

```
100%|██████████| 4035/4035 [00:52<00:00, 77.20it/s]
```

```
[55] X_test = []
     for r in tqdm(test_reviews):
       emb = use(r)
       review_emb = tf.reshape(emb, [-1]).numpy()
       X_test.append(review_emb)

     X_test = np.array(X_test)
```

```
100%|██████████| 449/449 [00:05<00:00, 80.43it/s]
```

```
[55]    review_emb = tf.reshape(emb, [-1]).numpy()
        X_test.append(review_emb)

    X_test = np.array(X_test)
```

```
100%|██████████| 449/449 [00:05<00:00, 80.43it/s]
```

```
[56] print(X_train.shape)
```

```
(4035, 512)
```

```
[57] print(X_test.shape)
```

```
(449, 512)
```

```
[58] print(y_train.shape)
```

```
(4035, 2)
```

The analysis portion using keras

```
[59] model = keras.Sequential()
```

```
[ ] model.add(
        keras.layers.Dense(
            units=255,
            input_shape=(X_train.shape[1],),
            activation='relu'
        )
    )
```

```
[60] model.add(keras.layers.Dropout(rate=0.5))
```

```
[60] model.add(keras.layers.Dropout(rate=0.5))
```

```
[61] model.add(
         keras.layers.Dense(
             units=250,
             activation='relu'
         )
     )
```

```
[62] model.add(keras.layers.Dropout(rate=0.5))
```

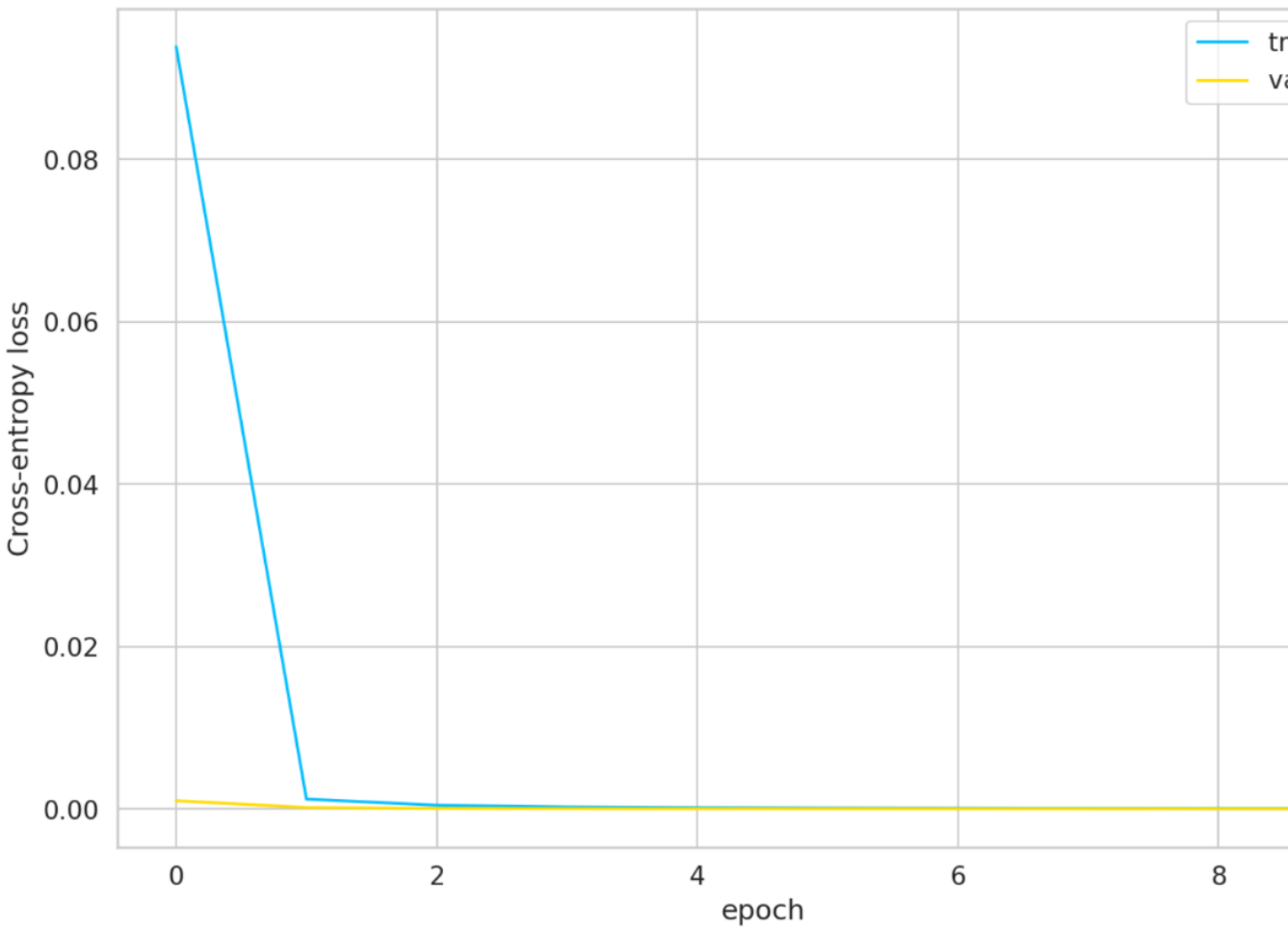
```
[63] model.add(keras.layers.Dense(2, activation='softmax'))
```

```
[64] model.compile(
         loss='categorical_crossentropy',
         optimizer=keras.optimizers.Adam(0.001),
         metrics=['accuracy']
     )
```

```
[65] history = model.fit(
         X_train, y_train,
         epochs=10,
         batch_size=16,
         validation_split=0.1,
         verbose=1,
         shuffle=True
     )
```

```
)
```

```
Epoch 1/10
227/227 [==============================] - 1s 3ms/step - loss: 0.0939 - accuracy: 0.9796
Epoch 2/10
227/227 [==============================] - 1s 2ms/step - loss: 0.0012 - accuracy: 1.0000
Epoch 3/10
227/227 [==============================] - 1s 2ms/step - loss: 4.4161e-04 - accuracy: 1.
Epoch 4/10
227/227 [==============================] - 1s 2ms/step - loss: 2.2869e-04 - accuracy: 1.
Epoch 5/10
227/227 [==============================] - 1s 2ms/step - loss: 1.4737e-04 - accuracy: 1.
Epoch 6/10
227/227 [==============================] - 1s 2ms/step - loss: 1.0401e-04 - accuracy: 1.
Epoch 7/10
227/227 [==============================] - 1s 2ms/step - loss: 8.0803e-05 - accuracy: 1.
Epoch 8/10
227/227 [==============================] - 1s 2ms/step - loss: 5.6202e-05 - accuracy: 1.
Epoch 9/10
227/227 [==============================] - 1s 2ms/step - loss: 4.5122e-05 - accuracy: 1.
Epoch 10/10
227/227 [==============================] - 1s 2ms/step - loss: 3.3303e-05 - accuracy: 1.
```




```
[67] model.evaluate(X_test, y_test)
```

```
15/15 [==============================] - 0s 2ms/step - loss: 1.6079e-06 - accuracy: 1.0
[1.6078635098892846e-06, 1.0]
```

Checking to see the accuracy of the predition

Checking to see the accuracy of the predition

```
[69] print(test_reviews.iloc[0])
     print("good" if y_test[0][0] == 1 else "bad")
```

```
not_racist_sexit
good
```

```
[71] y_pred = model.predict(X_test[:1])
     print(y_pred)
     "good" if np.argmax(y_pred) == 0 else "bad"
```

```
[[9.9999833e-01 1.6169323e-06]]
'good'
```

e. Output file link if applicable

https://github.com/UMKC-APL-BigDataAnalytics/icp4-irfancheemaa

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

Should not need access. If you do please let me know. I changed my settings to where it is accessible once the link is clicked.

https://umkc.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=7835825c-b569-4347-9eba-ac3f005d597d

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

I really had difficult with this topic and still am left with confusion.