final-project

December 9, 2019

1 Introduction:

This project is mostly to analyze some potential political information from a Twitter dataset covering tweets coming from the United States and Canada. The questions I want to look at for this dataset are the general sentiment of the different tweets, the location popularity of certain tweets, time information, and the aspects of sentiment in this sample, ideally.

I ended up finding that determining gender based on the features from tweets was very difficult and often gave pretty low accuracy scores. I think this mostly resulted from the lower information used than was actually used for the project itself. I would possibly want to use computer vision and deeper analysis about groups of tweets than individual tweets themselves.

I referenced a GitHub repository called: https://github.com/shaypal5/awesome-twitter-data#tweet-datasets-labelled

From there, I chose to use twitter data that was classified to gender by a project on: https://data.world/crowdflower/gender-classifier-data

The following code is mostly import statements surrounding the analysis and usage of the data. I primarily highlighted dataframes coming from pandas and used other libraries to have surrounding information on the datasets.

```
In [330]: import csv
          import io
          import re
          import string
          import math
          import nltk
          import collections
          import random
          import glob
          import numpy as np
          import plotly
          import pandas as pd
          import plotly.graph_objs as go
          import sklearn as sk
          from sklearn.linear_model import LogisticRegression
          from sklearn.preprocessing import normalize
          import matplotlib.pyplot as plt
          from pandas.tools.plotting import table
```

2 Data Cleaning

I wanted to look through the data and see how much of the data ended up actually being useful. There were also a lot of extraneous columns that were not relevant for my actual analysis based on text data. Much of the dataset that I was looking at was related to the determination of gender from the original project plan. As a result, information such as profile picture, color of links/profile, and mostly null/empty values were removed.

```
In [361]: df = pd.read_csv('gender_classifier_dfe_791531.csv')
          print(len(df))
          #print(df["user_timezone"].value_counts())
```

```
20050
```

```
In [332]: col_names = ["unit_id", "golden", "unit_state", "trusted_judgments", "last_judgment_at
```

```
In [362]: for name in col_names:
              if name not in ["gender", "created", "fav_number", "name", "retweet_count", "text"
                  del df[name]
          #print(df)
```

```
In [363]: filtered = df[df.user_timezone.isin(["Pacific Time (US & Canada)", "Central Time (US &

          #print(filtered)
```

3 Demographics

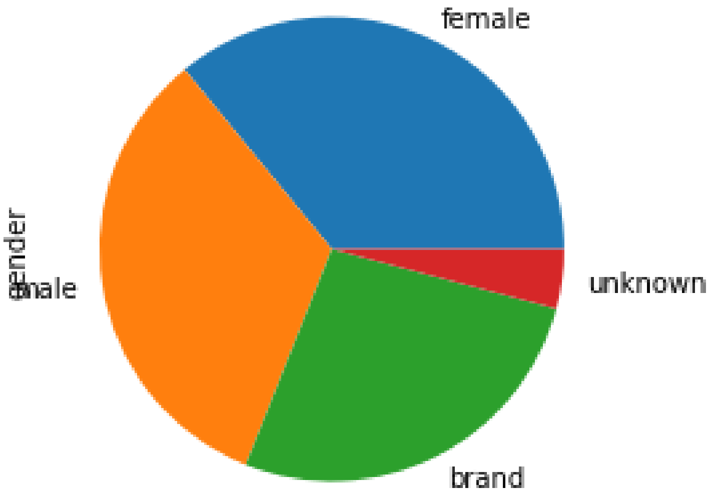
It was interesting to see the actual predicted genders of most of these tweets (many were also found to be brands).

I also wanted to get some confirmation on the areas that we were looking at for these data points.

```
In [335]: print(filtered.gender.value_counts())
          print("Of a total: " + str(len(filtered)))

          filtered.gender.value_counts().plot(kind='pie')
```

```
female     2721
male       2500
brand      2039
unknown     312
Name: gender, dtype: int64
Of a total: 7588
```

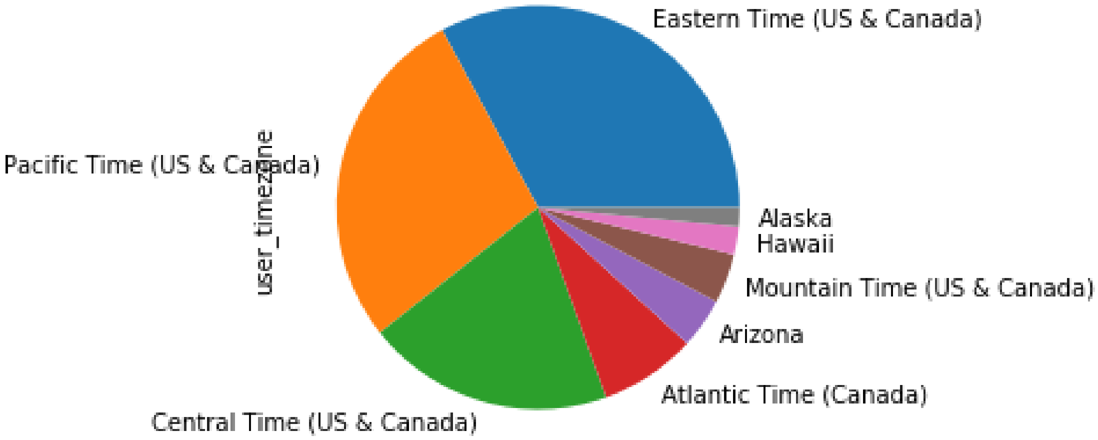
```
Out[335]: <matplotlib.axes._subplots.AxesSubplot at 0x7efc64bf47b8>
```



```
In [336]: print(filtered.user_timezone.value_counts())
          print("Of a total: " + str(len(filtered)))
          filtered.user_timezone.value_counts().plot(kind='pie')
```

```
Eastern Time (US & Canada)     2496
Pacific Time (US & Canada)     2106
Central Time (US & Canada)     1505
Atlantic Time (Canada)          589
Arizona                         306
Mountain Time (US & Canada)     301
Hawaii                          170
Alaska                          115
Name: user_timezone, dtype: int64
Of a total: 7588
```

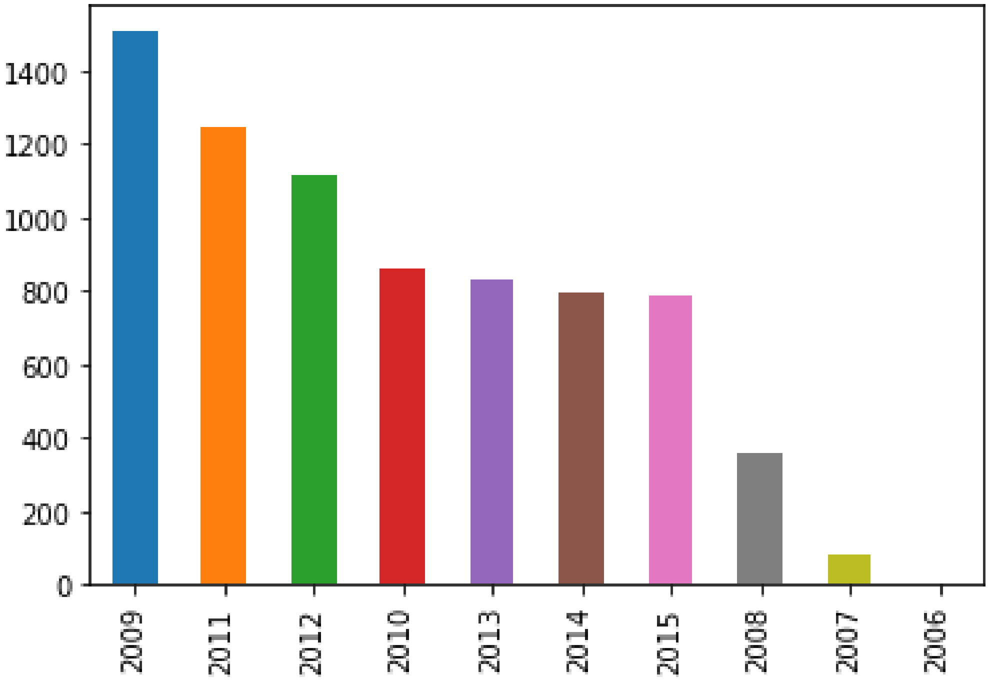
```
Out[336]: <matplotlib.axes._subplots.AxesSubplot at 0x7efc64f9d550>
```



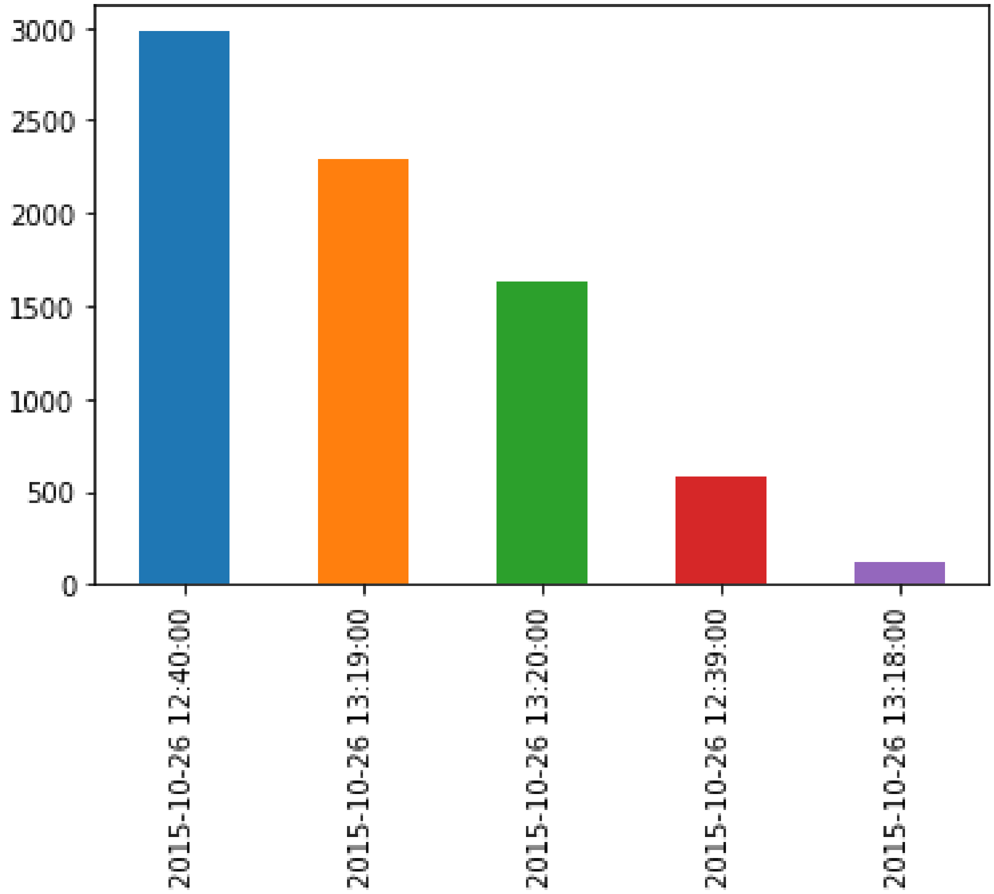
```
In [337]: filtered["created"] = pd.to_datetime(filtered['created'])

          filtered["created"].dt.year.value_counts().plot(kind = "bar")
```

```
/usr/lib/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
```

```
Out[337]: <matplotlib.axes._subplots.AxesSubplot at 0x7efc622d1940>
```



```
In [338]: filtered["tweet_created"] = pd.to_datetime(filtered['tweet_created'])

          filtered["tweet_created"].value_counts().plot(kind = "bar")

/usr/lib/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#



Out[338]: <matplotlib.axes._subplots.AxesSubplot at 0x7efc66745c88>
```



4 Commonalities in Words and Rankings

In order to explore the dataset a bit more, I wanted to look into the actual tweets and users behind the tweets. Here, I looked into some stats around the most popular words in the dataset, some of the highest favorited account's tweets, and same for retweets.

Graphs data can be found here and code for future information.

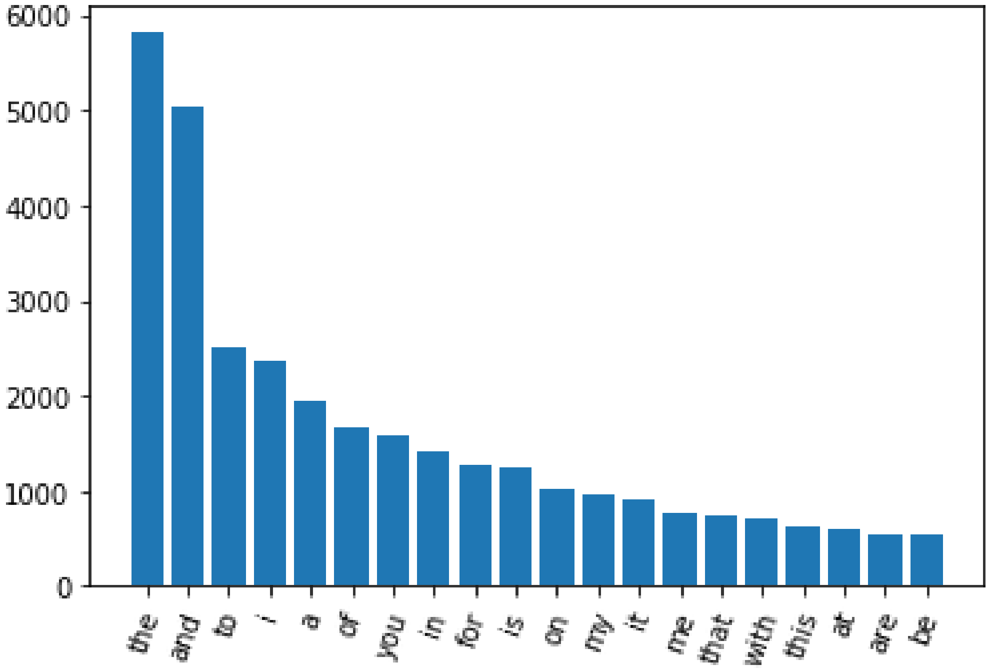
I mostly took care to add many counters surrouding the datasets and filtered a bit more deeply through data that was actually labelled with genders fairly confidently.

```
In [339]: wordcounts = collections.Counter()
          favorites = collections.Counter()
          retweets = collections.Counter()

          male_words = collections.Counter()
          female_words = collections.Counter()

          word_counters = []
```

```
for index, row in filtered.iterrows():
    text = row['text']
    words = re.sub(u'['+string.punctuation+u']', '', text).split()
    counts = collections.Counter(w.lower() for w in words)
    wordcounts.update(counts)
    if row['gender'] == "male":
        male_words.update(counts)
        word_counters.append(counts)
    if row['gender'] == "female":
        female_words.update(counts)
        word_counters.append(counts)
    favorites[row['name'] + "---" + text] = row['fav_number']
    retweets[row['name'] + "---" + text] = row['retweet_count']
```

```
In [340]: high_freq_words = wordcounts.most_common(20)

          print(high_freq_words)

          plt.bar(range(len(high_freq_words)), [val[1] for val in high_freq_words], align='cente
          plt.xticks(range(len(high_freq_words)), [val[0] for val in high_freq_words])
          plt.xticks(rotation=70)
          plt.show()
```

```
[('the', 5811), ('and', 5031), ('to', 2503), ('i', 2363), ('a', 1940), ('of', 1674), ('you', 159
```



```
In [341]: males = high_freq_words = male_words.most_common(20)

          print(high_freq_words)
```

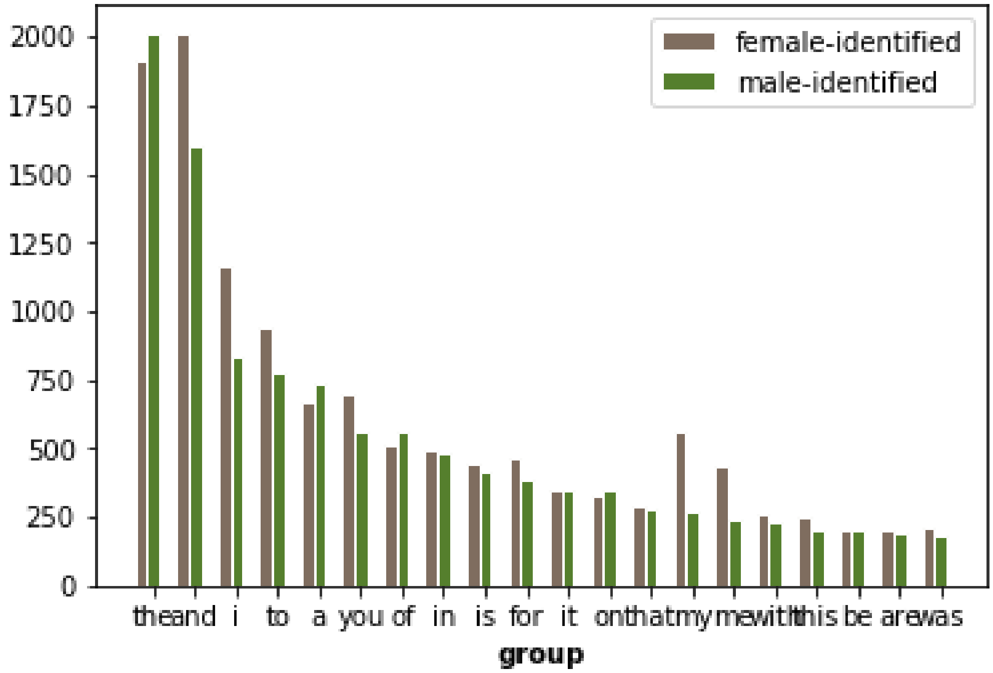
```
[('the', 2014), ('and', 1602), ('i', 839), ('to', 780), ('a', 736), ('you', 563), ('of', 563), (
```

```
In [342]: females = high_freq_words = female_words.most_common(20)

          print(high_freq_words)
```

```
[('and', 2008), ('the', 1918), ('i', 1161), ('to', 944), ('you', 696), ('a', 671), ('my', 558),
```

Here you can see some of the interesting comparisons between words that were very commonly used, an interesting thing to notice is that there was a higher prevelance of pronounces from the female-identified group, we could see words such as "i", "you", "me" very heavily used in the 'female' tweets.

```
In [343]: intersec = male_words & female_words

          men_common = []
          women_common = []
          words = []

          for pair in intersec.most_common(20):
              key = pair[0]
              women_common.append(female_words[key])
              men_common.append(male_words[key])
              words.append(key)

          barWidth = .3

          r1 = np.arange(len(women_common))
          r2 = [x + barWidth for x in r1]

          plt.bar(r1, women_common, color='#7f6d5f', width=barWidth, edgecolor='white', label='f
          plt.bar(r2, men_common, color='#557f2d', width=barWidth, edgecolor='white', label='mal

          plt.xlabel('group', fontweight='bold')
          plt.xticks([r + barWidth for r in range(len(women_common))], words)

          plt.legend()
          plt.show()
```



```
In [344]: high_fav = favorites.most_common(20)

          for fav in high_fav:
              print(fav)

("ariel_thethird---@MarziaPie and your hair looks so cute in the new Marzia's Time video _", 341
("NATSUH1---i love diane i'm glad we got her in court this episode and we got a little cary and 
('facepaulmrevere---OH GOD this reminds me of that time a former employee of the city of Chicago
('Love_bug1016---The fun thing about having a vagina is waking up not knowing if you need to get
('evankirstel---Flipping the Equation: The Mobile Enterprise in 2020 https://t.co/7mCbbEg7Gr @no
('13spencer---Depression comes when you least expect it, like the shittiest person planning a su
('RalphieeLove---In order to play The Joker on screen, one must completely lose themselves in th
('TommiesMommy14---@LaughWithUsBlog A5 Keep tabs on the kids  #TreatYourFamily', 142892)
('ThirdEyeYoukai---"I killed the rat!"\n\n"Actually there were two of them"\n\n"..." #JJBA', 139
("chafos---Maybe I'm the last human alive and everyone else is just a hallucination by a fairy t
('slasher48---@HALSTlEL I think hes easy to cosplay, and I have cosplayed him, but Im probably a
('cabEYomyeggo---omg the girls got a picture with justin..and is that miley !???! wow ! https://
('Camsboothang---@cameron_unicorn ID LIKE AND RV', 132462)
('bootymeats---im always sayin "u r so pretty" "ur beautiful" "ilysm" "u are the most beautiful 
("JonsCrazyTweets---@candybriones And when we have our successful shows and toys, we'll have a t
('johnIaurens---but Tesla, who also spent a winter digging ditches and crying, had the rage and 
('johnIaurens---imagine hamilton but all the dudes are played by cute girls,', 126805)
('TammaraMaiden1---@LilMsgss @Anomaly100 \n\nAnomaly is the best.__', 125456)
('sincerelyjonah---When u turn on the tv and girl meets world is on ____________________________
('JhonRules---When people choose to sit next to me on the bus I assume they think "that guy look
```

```
In [345]: high_rt = retweets.most_common(20)

          for rt in high_rt:
              print(rt)
```

```
('LittleMix---Love this by the amazingly talented @elenamartynyuk _ https://t.co/F7ky0y5Htr', 33
('Meghan_Trainor---#BETTERWHENIMDANCIN lyric video on @VEVO!! @PeanutsMovie characters are THE c
("sayingsforgirls---My problem is that I'll always be the one putting in more effort just to kee
('JivDude---If you want a new vine hop over there and like my most recent post :)', 20)
("HotTopic---Rainbow hair, don't care? Show us and you could win a year's supply of hair dye. LA
('nikesoccer---Force of nature.\n\nCR7 Savage Beauty #Mercurial, available soon exclusively in t
('billboard---.@TheWeeknd tops the #Hot100 (again!), but will @Adele be No. 1 next week? https:/
('AltPress---Big announcement happening tomorrow... Who will be the winner of the @VansWarpedTou
("ProjectBuddy---This is an abusive relationship and should not be romanticized. Don't let your 
('1DCharts---Spotify: "Perfect"\n\n13. Worldwide (=) \n21. United States (=)\n\n@onedirection Ar
('jonmorosi---Ruben Amaro Jr. is ambidextrous and will throw batting practice for the @RedSox bo
("Patriots---Who's up and who's down in the #Patriots Week 7 win: https://t.co/sCR02uere3 https:
('GriffinArnlund---My birthday was amazing ___ thank you for the support!  https://t.co/zuuqPSHo
('LovatoCrowd---"You\'ve been friends for so long, how does it feel to be going on tour together
('NHLFlames---"Hopefully we can find a away to get some greasy goals and spark the offence." - J
('Our5SOSUpdatesx---Ashton and Calum on snapchat this morning! https://t.co/PtSRv6mzQ7', 7)
('yo---The loudest way to open a bag of chips is to try and open them quietly', 6)
('RealTristan13---Tomorrow, we tip off a new season...and a new era. @MountainDew @NBA #DEWxNBA 
('WestsiiideTray---Without that badge, you a bitch and a half ___', 6)
('justphilanddan---sometimes i remember dan and phil slapped each other on camera before and lau
```

5 Running a similar experiment

I am now curious if we can manage to classify tweets into correct genders, I think it is not very likely given the features that I have maintained, but we can try. I wanted to format this similarly to the comparison we made between Jane Austen and Walter Scott. I don't think it is as likely for as unstructured text data to fall as neatly into distinguishable features when a lot of the content is going to be similar or related to just the thoughts of individuals.

```
In [346]: def matrix(cts, features) :
                  result = np.zeros((len(cts), len(features)))
                  for n, w in enumerate(features):
                      result[:,n] = (cts[w])/sum(cts.values())
                  return normalize(result, norm='l2')
```

```
In [347]: w = ['a', 'all', 'also', 'an', 'and', 'any', 'are', 'as', 'at',
                'be', 'been', 'but', 'by', 'can', 'do', 'down', 'even', 'every',
                'for', 'from', 'had', 'has', 'have', 'her', 'his', 'i' 'if', 'in',
                'into', 'is', 'it', 'its', 'may', 'more', 'must', 'my', 'no',
                'not', 'now', 'of', 'on', 'one', 'only', 'or', 'our',
                'shall', 'should', 'so', 'some', 'such', 'than', 'that',
                'the', 'their', 'then', 'there', 'thing', 'this', 'to',
                'up', 'upon', 'was', 'were', 'what', 'when', 'which',
                'who', 'will', 'with', 'would', 'your', 'you', 'me']

         matrices = [matrix(x, w)[0] for x in word_counters]
```

```
In [348]: actual_genders = []
          for x in filtered['gender']:
              if x == 'male':
                  actual_genders.append(0)
              if x == 'female':
                  actual_genders.append(1)
          print(matrices[0])
          print(actual_genders[0])
          print(len(matrices))
          print(len(actual_genders))
```

```
[0.         0.         0.         0.         0.40824829 0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.40824829 0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.40824829 0.         0.
 0.         0.         0.         0.         0.         0.
 0.40824829 0.40824829 0.         0.         0.         0.
 0.         0.40824829 0.         0.         0.         0.        ]
0
5221
5221
```

```
In [349]: X_all = matrices
          y_all = actual_genders

          classifier = \
          sklearn.linear_model.SGDClassifier(loss="log",
                                             penalty="elasticnet")

          _ = classifier.fit(X_all, y_all)
```

```
/usr/lib/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/stochastic_gradient.py:166:

max_iter and tol parameters have been added in SGDClassifier in 0.19. If both are left unset, th
```

```
In [350]: pred = classifier.predict(X_all)
```

```
In [351]: sklearn.metrics.accuracy_score(y_all, pred)
```

```
Out[351]: 0.5811147289791228
```

```
In [352]: print(sklearn.metrics.confusion_matrix(y_all, pred))
```

```
[[1159 1341]
 [ 846 1875]]
```

```
In [353]: folds = 5

          fold_size = math.floor(len(X_all) / folds)
          extra = len(X_all) % folds
          groups = np.array(fold_size * list(range(folds)) + list(range(extra)))
          np.random.shuffle(groups)
```

```
In [354]: def train_and_eval(i, X_train, y_train, X_test, y_test) :
              print("Fold {}: Training on {}, testing on {} items".format(
                  i,
                  len(y_train),
                  len(y_test)))
              classifier = \
          sklearn.linear_model.SGDClassifier(loss="log",
                                             penalty="elasticnet")

              _ = classifier.fit(X_train, y_train)
              pred = classifier.predict(X_test)
              print(sklearn.metrics.accuracy_score(y_test, pred))
              print(sklearn.metrics.confusion_matrix(y_test, pred))
```

6 Trying to gain sentiment from Twitter data

I was now curious if we could see any general differences in sentiment between the genders, given enough time, let's continue to look at the differences between other features in our dataset.

I will be using NLTK and specifically the Sentiment Intensity Analyzer in order to determine some information about the actual general positivity from each tweet.

```
In [355]: from nltk.sentiment.vader import SentimentIntensityAnalyzer
          import nltk
          nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /ilab/users/dp865/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

```
Out[355]: True
```

```
In [365]: sid = SentimentIntensityAnalyzer()
          positivity = []
          for index, row in filtered.iterrows():
              if not (row['gender'] == 'male' or row['gender'] == 'female'):
                  continue
              sentence = row['text']
              ss = sid.polarity_scores(sentence)
              positivity.append(ss['pos'])
          #     for k in sorted(ss):
          #         print('{0}: {1}, '.format(k, ss[k]), end='')
          #     print()
```

```
In [366]: mcount = 0
          mpos = 0
          fcount = 0
          fpos = 0
          for i in range(len(actual_genders)):
              gender = actual_genders[i]
              pos = positivity[i]
              if gender == 0:
                  mcount += 1
                  mpos += pos
              else:
                  fcount += 1
                  fpos += pos
```

```
In [367]: print(mcount, mpos, mpos/mcount)

2500 313.059 0.12522360000000002
```

```
In [368]: print(fcount, fpos, fpos/fcount)

2721 352.27499999999975 0.12946527012127884
```

7 Conclusion

I think that this project overall was very interesting. I ended up moving away from my initial project idea after finding this dataset and completing a mini project related to political sentiments. I wasn't able to really gather the information I wanted, but having worked with this dataset, I thought I could find other information surrounding it.

This was also as a result of inconsistencies in the dataset itself, not many tweets actually contained regular or defined timezones or coordinates, which made my initial problem harder. In addition, there were different languages included in the various tweets and it was hard to sample over a large period of time with some of the datasets that I found online.

I believe that there is a possibility of being able to classify tweets based on text data, but perhaps using my slightly modified "w70" wasn't strong enough. I would want to run more tests to see whether just using the most common words could be a better call in the long run.

After the attempted classification, I wanted to look at general sentiments between the genders and see if there were any glaring pieces of information that were surprising to me. The overal difference was not very large, but I believe it could be interesting to filter based on location or perhaps even by the relative popularity of certain Twitter users based on total tweet/favorite/retweet counts of an account.

Although it would be going away from Language as the data, I think involving the profile pictures would really help strengthen my analysis and perhaps finding a way to use the personal bio's/descriptions could lend to more accuracy. General accuracy was (52%~58%)

```
In [ ]:
```