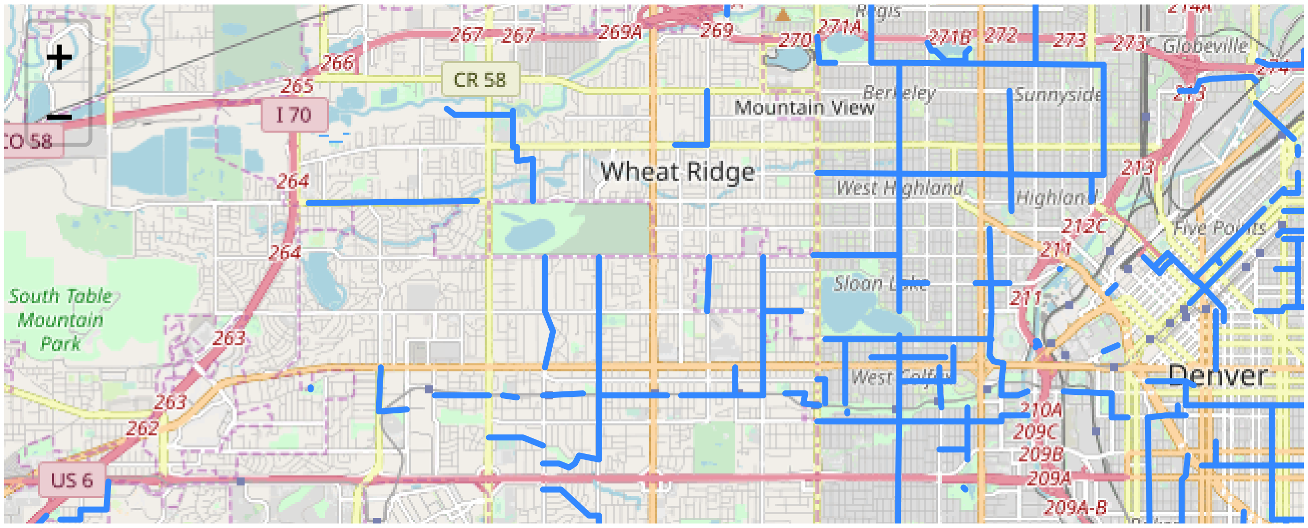
Part 1 Denver Bike Map

Load Denver Map

```
!pip install git+https://github.com/python-visualization/folium
!pip install geopandas

import folium
import geopandas as gpd
import json
import os
import pandas as pd
```

Load Bike Routes

```
coords_den=(39.7, -105.0)
layer_fac='Bike Facility.json'
df1 = gpd.read_file(layer_fac)

len(df1) #10080 lines/bike route records
df11=df1[0:800]


map_den = folium.Map(location=coords_den,zoom_start=12)
folium.GeoJson(df11).add_to(map_den)
map_den

###reference: https://blog.dominodatalab.com/creating-interactive-crime-maps-with-fc
```

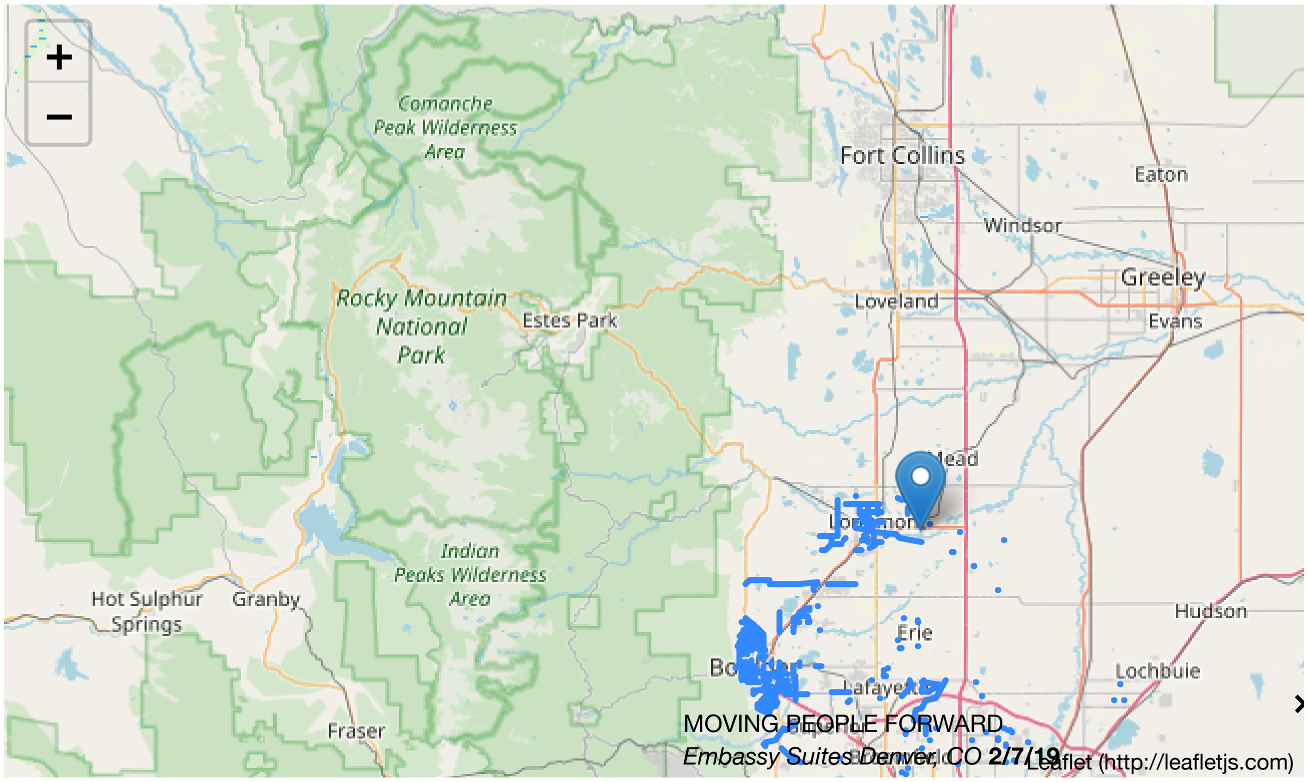


Load Events and Map Markers

```
file_event = 'DenverEventDecFeb.csv'
df_event = pd.read_csv(file_event)
df_event
```

	Date	Location	Lat	Lon	Title	
0	12/1/18	Salisbury Equestrian Park and Sports Complex P...	39.501161	-104.775139	COLORADO STATE CX CHAMPIONSHIPS	https://www.withoutlimits.
1	12/8/18	Sandstone Ranch Longmont, CO	40.156807	-105.042667	ROCKY MOUNTAIN CX REGIONAL CHAMPIONSHIP	https://www.withoutlimits.c
2	2/7/19	Embassy Suites Denver, CO	39.707776	-104.955089	MOVING PEOPLE FORWARD	https://www.bicyclecolorado
3	2/8/19	Denver,	39.764519	-104.995194	INTERNATIONAL WINTER BIKE TO	https://wir

```
df_event["Details"]=df_event["Title"]+'<br /><i>'+df_event["Location"]+'  </i> <b>'+
for i in range(0, len(df_event)):
    folium.Marker((df_event.iloc[i]["Lat"],df_event.iloc[i]["Lon"]), popup=df_event.
map_den.save('index.html')
map_den
```



Part 2 Text Analysis

Initialize Environement

```
# This fix for a new bug appeared in Novermber is needed to use spaCy.
# https://github.com/explosion/spaCy/issues/2995
!pip install -U spacy==2.0.13
!pip install "msgpack==0.5.6"
import spacy
!spacy download en_core_web_sm
nlp = spacy.load('en_core_web_sm')
import thinc
import thinc.neural.ops

!pip install tweepy

import matplotlib.pyplot as plt
import nltk
import numpy as np
import re
import string
import time
import tweepy

from sklearn.cluster import KMeans
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer

nltk.download('punkt')
nltk.download('stopwords')
```

```
os.environ['TWITTER_CONSUMER_KEY']='cdbYllmYYfWOlTxS3CxxrDaZW'
os.environ['TWITTER_CONSUMER_SECRET']='49QbMIl6yI2sLX0xyHEb7iuaTNL67MwFAHa8fmqIF4kEf
os.environ['TWITTER_ACCESS_TOKEN_KEY']='194388233-nC2LPMq9eTVDDSuv6FMIfQbPqv31SL69XM
os.environ['TWITTER_ACCESS_TOKEN_SECRET']='rYz1ThRLUo2QsHrq38y3Teurxr85r8t6JODq25Fln
```

```
# Accessing environment variables
os.environ['PATH']
consumer_key = os.environ.get('TWITTER_CONSUMER_KEY')
consumer_secret = os.environ.get('TWITTER_CONSUMER_SECRET')
access_token_key = os.environ.get('TWITTER_ACCESS_TOKEN_KEY')
access_token_secret = os.environ.get('TWITTER_ACCESS_TOKEN_SECRET')
```

```
# Connect
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token_key, access_token_secret)

# Authorize API without too many requests
## Reference: https://stackoverflow.com/questions/41786569/twitter-error-code-429-wi
api = tweepy.API(auth,wait_on_rate_limit=True)
```

Collect Tweets - Denver Bike/Bicycle/Cycling

```
from tweepy import Cursor
def collect_tweets(loc):
    tweets = [tweet._json['text'] for tweet in Cursor(api.search,"bicycle OR bike OF
    return tweets
tweets = collect_tweets('39.7392,-104.9847,20mi')
```

Clean Tweets

```
stopwords = nltk.corpus.stopwords.words('english')
stopwords = set(stopwords + ['rt','video','get','say','make','think','like','bike','

# new approach to cleaning text, take a minute and review this code
# simple clean text function -- spacy lowercases, removes stopwords, lemmatizes
def clean_text(docs):
    # remove punctuation and numbers
    # I do this before lemmatizing, so things like "act's" turn into 'act' instead c
    print('removing punctuation and digits')
    table = str.maketrans({key: None for key in string.punctuation + string.digits})
    docs = [d.translate(table) for d in docs]
    docs[:5]
    print('replace newlines with spaces')
    docs = [re.sub('[\r\n]+','', d) for d in docs]
    docs[:5]
    print('replace web links with spaces')
    docs = [re.sub('https[\w]*',' ', d) for d in docs]
    docs[:5]
    print('replace spaces with one space')
    docs = [re.sub('\s\s*',' ', d) for d in docs]
    docs[:5]
    print('remove weird characters')
    clean_docs = [d.encode('ascii', errors='ignore').decode('ascii') for d in docs]
    clean_docs = [d.encode('ascii', errors='ignore').decode('ascii') for d in docs]
    clean_docs[:5]
    print('spacy nlp...')
    # nlp() function uses spacy's neural network to preform lemmatization on the col
    nlp_docs = [nlp(d) for d in clean_docs]
    clean_docs =[]
    # keep the word if it's a pronoun, otherwise use the lemma
    # otherwise spacy substitutes '-PRON-' for pronouns
    print('getting lemmas')
    for d in nlp_docs:
        temp_doc = []
        for w in d:
            if w.lemma_ in stopwords or len(w.lemma_)>10 or len(w.lemma_)<2:
                continue
            elif w.lemma_ !='-PRON-':
              temp_doc.append(w.lower_)
        # join tokens back into a string with each word separated by a space
        clean_docs.append(' '.join(temp_doc))

    return clean_docs


clean_tweets = clean_text(tweets)
clean_tweets[:5]
```

```
removing punctuation and digits
replace newlines with spaces
replace web links with spaces
replace spaces with one space
remove weird characters
spacy nlp...
getting lemmas
['sometimes goals need benefit time happen come fruition know rush fath',
 'friends great friendships priceless thank monarchflys thoughtful awesome ch
 'framed marquette alloy mountain sram suntour raidon fork normal price',
 'fuji nevada er mountain onsale fujibike er',
 'rock climbing amp riding sunday appreciate golden colorado']
```

Checking Point

```
filename = 'clean_den'+ time.strftime("%Y%m%d-%H%M%S")+'.txt'

def write_tweets(filename, twtext):
  with open(filename,'w') as out_file:
      for line in twtext:
        out_file.write(line + '\n')

def read_tweets(filename):
    return pd.read_csv(filename, index_col=False, header=None, names=['tweet'])

write_tweets(filename,clean_tweets)

clean_tweets =read_tweets(filename)
clean_tweets.head()
```

	tweet
0	sometimes goals need benefit time happen come ...
1	friends great friendships priceless thank mona...

PART 3 Advanced Text Analysis

```
!pip install wordcloud
nltk.download('vader_lexicon')

import nltk.collocations as nc


from nltk.sentiment.vader import SentimentIntensityAnalyzer as SIA
from wordcloud import WordCloud
```

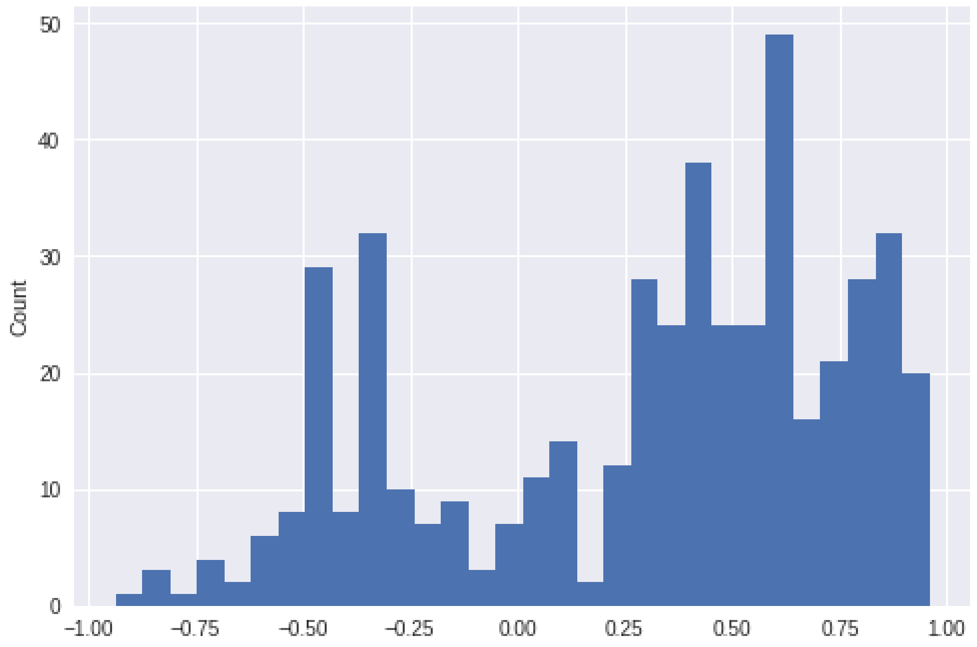
```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: pillow in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: olefile in /usr/local/lib/python3.6/dist-packa
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
/usr/local/lib/python3.6/dist-packages/nltk/twitter/__init__.py:20: UserWarni
  warnings.warn("The twython library has not been installed. "
```

```
sia = SIA()

def score(tweets):
    tweets['polarity'] =tweets['tweet'].apply(sia.polarity_scores)
    polarity_df = tweets['polarity'].apply(pd.Series)
    tweets = tweets.join([polarity_df])
    tweets = tweets.drop(['polarity'],axis=1)
    return tweets

clean tweets = score(clean_tweets)
clean_tweets.head()
```

	tweet	compound	neg	neu	pos
0	sometimes goals need benefit time happen come ...	0.4588	0.0	0.769	0.231
1	friends great friendships priceless thank mona...	0.9584	0.0	0.136	0.864
2	framed marquette alloy mountain sram suntour r...	0.0000	0.0	1.000	0.000
3	fuji nevada er mountain onsale fujibike er	0.0000	0.0	1.000	0.000
4	rock climbing amp riding sunday appreciate gol...	0.4019	0.0	0.722	0.278

```
not_neutral = clean_tweets[clean_tweets['neu']!=1]
plt.hist(not_neutral['compound'],bins=30)
plt.xlabel('Sentiment')
plt.ylabel('Count')
```

```
Text(0,0.5,'Count')
```



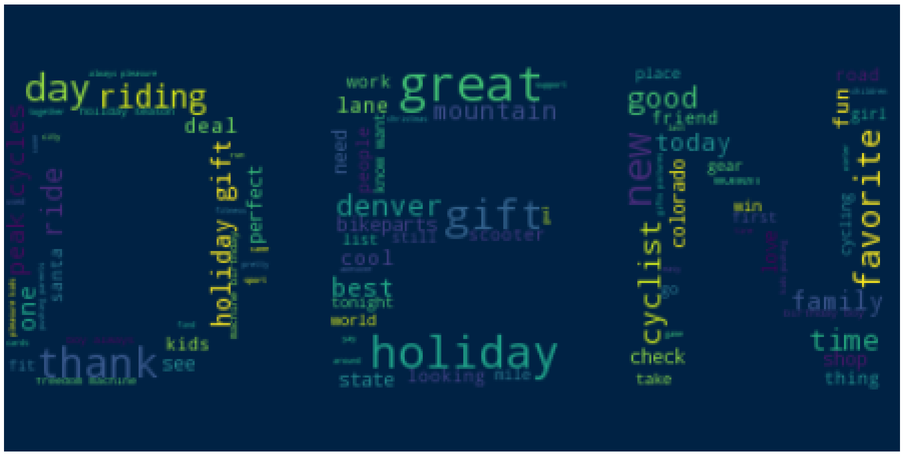
```
from nltk import FreqDist, word_tokenize
from itertools import chain
def flatten_word_list(series):
    return list(chain.from_iterable([word_tokenize(line) for line in series]))


def word_commonpos(tweets):
    pos = tweets[tweets['pos']>0.2]
    pos_words = flatten_word_list(pos['tweet'])
    pos_fd = FreqDist(pos_words)
    return pos_fd.most_common(10)

word_denPos = word_commonpos(clean_tweets)
word_denPos
```

```
[('great', 37),
 ('holiday', 30),
 ('bikeparts', 24),
 ('new', 22),
 ('good', 20),
 ('holidays', 19),
 ('want', 19),
 ('time', 18),
 ('thanks', 18),
 ('best', 18)]
```

Word Cloud

```
fulltext = ''
pos = clean_tweets[clean_tweets['pos']>0.2]
for i in range (len(pos)):
    fulltext = fulltext + pos["tweet"].iloc[i]
fulltext

from PIL import Image
file_mask='DENmask.png'
mask =np.array(Image.open(file_mask))

# hint: construct a wordcloud text you extracted from the book, not your tokens
# see the wordcloud documentation to understand how to configure your cloud

# Generate a word cloud image
wordcloud = WordCloud(background_color='#002244', mask=mask).generate(fulltext)

# Display the generated image:
# the matplotlib way:
import matplotlib.pyplot as plt
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
```

```
(-0.5, 399.5, 199.5, -0.5)
```



PART 4 Data Comparison

What about other cities? We will examine Wausau (WI), Santa Monica (CA) and Washington(D.C.)

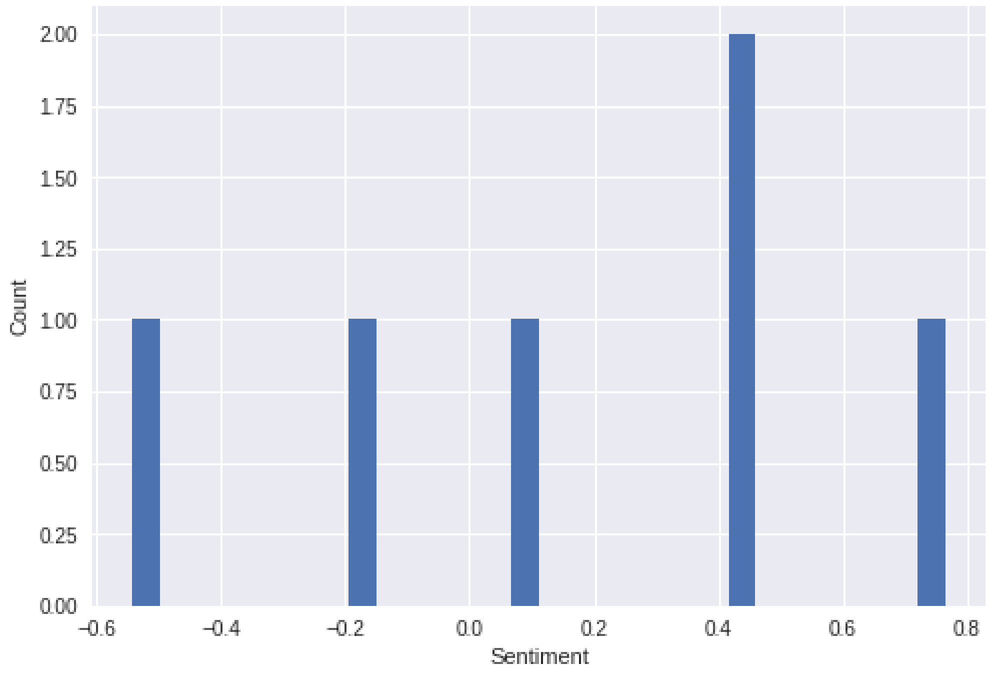
https://usa.streetsblog.org/2018/05/03/a-new-way-to-rank-americas-best-cities-for-bicycling/

```
# Wausau, WI

tweets = collect_tweets('44.9624539,-89.6958796,20mi')
clean_tweets = clean_text(tweets)

filename = 'clean_Wausau'+ time.strftime("%Y%m%d-%H%M%S")+'.txt'
write_tweets(filename,clean_tweets)
df_tweets = read_tweets(filename)
df_tweetscores = score(df_tweets)
word_WausauPos = word_commonpos(df_tweetscores)
```

```
    ['fuck this. I'm getting a bike.'
```

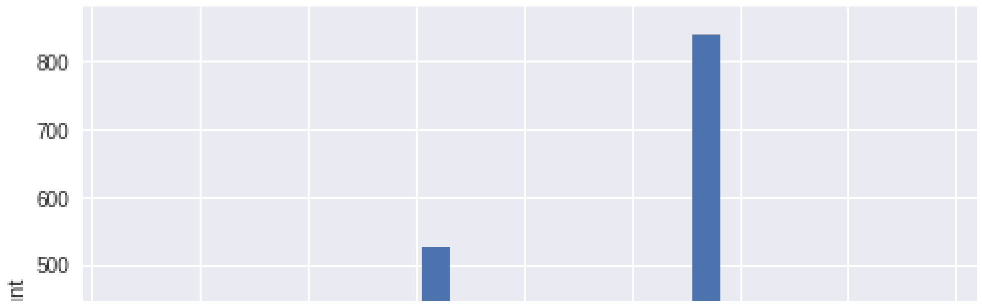
```
not_neutral = df_tweetscores[df_tweetscores['neu']!=1]
plt.hist(not_neutral['compound'],bins=30)
plt.xlabel('Sentiment')
plt.ylabel('Count')

# Note that how little tweets have been created with in the same radius.
```

```
Text(0,0.5,'Count')
```



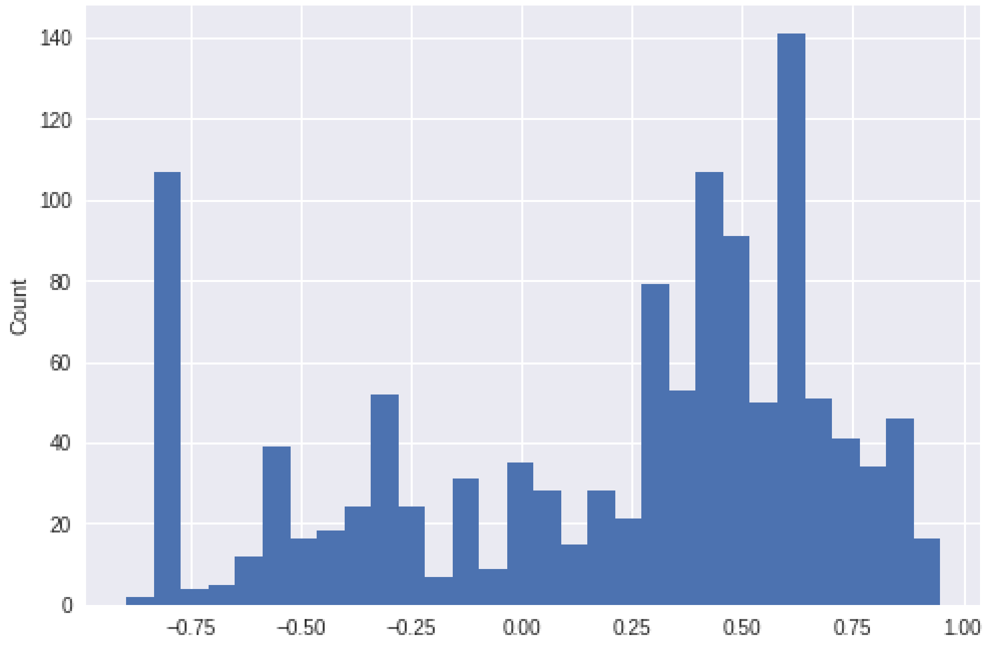
```
# Santa Monica, CA

tweets = collect_tweets('34.0218948,-118.4983079,20mi')
clean_tweets = clean_text(tweets)

filename = 'clean_SantaMonica'+ time.strftime("%Y%m%d-%H%M%S")+'.txt'
write_tweets(filename,clean_tweets)
df_tweets = read_tweets(filename)
df_tweetscores = score(df_tweets)
word_SantaMonicaPos = word_commonpos(df_tweetscores)

not_neutral = df_tweetscores[df_tweetscores['neu']!=1]
plt.hist(not_neutral['compound'],bins=30)
plt.xlabel('Sentiment')
plt.ylabel('Count')

# The most tweets and also unhappy tweets.
```

```
removing punctuation and digits
replace newlines with spaces
replace web links with spaces
replace spaces with one space
remove weird characters
spacy nlp...
getting lemmas
Text(0,0.5,'Count')
```



```
# Washington, D.C.

tweets = collect_tweets('38.893709,-77.0847872,20mi')
clean_tweets = clean_text(tweets)

filename = 'clean_DC'+ time.strftime("%Y%m%d-%H%M%S")+'.txt'
write_tweets(filename,clean_tweets)
df_tweets = read_tweets(filename)
df_tweetscores = score(df_tweets)
word_DCPos = word_commonpos(df_tweetscores)

not_neutral = df_tweetscores[df_tweetscores['neu']!=1]
plt.hist(not_neutral['compound'],bins=30)
plt.xlabel('Sentiment')
plt.ylabel('Count')
```

```
removing punctuation and digits
replace newlines with spaces
replace web links with spaces
replace spaces with one space
remove weird characters
spacy nlp...
getting lemmas
Text(0,0.5,'Count')
```


Count