[NSF] Sentiment Analysis_Polarity Analysis

@Author : Woojin Park, Nidhi Bhaskar

@Copyright : 2020, Neolth NSF grant NLP project

@Email : woojinpa@andrew.cmu.edu , nidhibha@andrew.cmu.edu

@Status : In-Progress

In [1]:

```
### Import Relevant Libraries
import os
import pandas as pd
import numpy as np
import collections
import datetime as dt
import requests
import json
import re
import time

import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns
from scipy.stats import norm

import string
import re
import nltk
from nltk.util import ngrams
from nltk import pos_tag,word_tokenize
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer,PorterStemmer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from textblob import TextBlob

from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
```

In [65]:

```
### Build a get_date function to convert date format
#### Build a data_creation function to read json data into pandas dataframe

def get_date(created):
    return dt.datetime.fromtimestamp(created)

def data_creation(subreddit) :
    with open('submissions_'+subreddit+'.json') as f:
        data = json.loads("[" +
            f.read().replace("}\n{", "},\n{") +
        "]")
    data =pd.DataFrame(data)
    reddit_data = data[['author','over_18','title','selftext','num_comments', 's
core', 'full_link','created_utc']]
    reddit_data = reddit_data.dropna()
    _timestamp = reddit_data["created_utc"].apply(get_date)
    reddit_data = reddit_data.assign(timestamp = _timestamp)
    reddit_data['over_18'] = reddit_data['over_18'].astype('str')
    reddit_data['subreddit']= subreddit
    # Build column have title + selftext
    reddit_data['title_with_selftext']= reddit_data['title'] +" " + reddit_data[
'selftext']

    # Do one more extra cleaning : keep updating this part
    reddit_data=reddit_data[~reddit_data['title_with_selftext'].isin([ '[removed
]', '[deleted]',''])]
    subreddit = reddit_data

    return subreddit

def empty_words_clean(text):
    text = text.replace('[removed]','')
    text= text.replace('[deleted]','')
    text= text.replace('\n','')
    return (text)
```

In [66]:

```
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
```

In [67]:

```
### Dataframing 4 subreddit Datasets
SuicideWatch_df = data_creation('SuicideWatch')
depressed_df = data_creation('depressed')
happy_df = data_creation('happy')
selfimprovement_df = data_creation('selfimprovement')

### Concat all 4 dataframes into one merged file
all_subreddit_df = pd.concat([SuicideWatch_df,depressed_df,happy_df,selfimprovem
ent_df])
all_subreddit_df.head(2)
```

Out[67]:

	author	over_18	title	selftext	num_comments	score	
0	DespressoCafe	False	I don't know where to go or what to do. I can'...	Let's make it quick. I'm almost 20. I've been ...	5	1	https://www.reddit.com/r/
1	LifeisCrumbling	False	I'm having an existencial crisis	If I only helped people either as a defense me...	1	1	https://www.reddit.com/r/

Sentiment Analysis

Polarity Analysis

Sentiment analysis is basically the process of determining the attitude or the emotion of the writer, i.e., whether it is positive or negative or neutral.

The sentiment function of textblob returns polarity. Polarity is float which lies in the range of [-1,1] where 1 means positive statement , -1 means a negative statement and 0 means a neutral statement.

0.Data Preparation

In [68]:

```
### Text Preprocessing by following pipeline :
### Raw text => Tokeninze/lowercase => Remove stop words => Remove non-alphabeti
c characters =>
### Remove Extra Punctuations => Lemmatization => Build Custom Stop words dictio
nary
```

In [69]:

```
# Build function that takes a word and returns true if it consists only of non-a
lphabetic characters
def alpha_filter(w):
    pattern = re.compile('^[^a-z]+$')
    if (pattern.match(w)):
        return True
    else:
        return False

# Build data preparation function including all the necessary 7 steps :
def clean_words(text):
    # lower text & tokenizing
    text =text.lower()
    text = [word for word in text.split(" ")]
    # remove stop words
    nltk_stopwords = set(stopwords.words('english'))
    review_lower_stop = [x for x in text if not x in nltk_stopwords]
    # remove punctuations
    review_lower_stop_pun = [y for y in review_lower_stop if not alpha_filter(y)
]
    review_lower_stop_pun_extra = [''.join(x for x in par if x not in string.pun
ctuation) for par in review_lower_stop_pun]
    # Lemmatization
    porter = WordNetLemmatizer()
    review_lower_stop_pun_extra_lemmatized = []
    for a in review_lower_stop_pun_extra :
        review_lower_stop_pun_extra_lemmatized.append(porter.lemmatize(a))
    # buid custom stop words dictionary
    cachedStopWords = set(stopwords.words("english"))

    ####Keep Updating custom stop words
    cachedStopWords.update(('nt', 'wo', 're', 'im', 'yall','u','ca','ive', 'wan'
,'na','gon','nov','x200b','amp',\
                        'wwwyoutubecomwatch','http','vbjkbl5olvm8','lt', 'br', '
gt', 'amp','tsp','tbsp','nbsp'))

    review_lower_stop_pun_extra_lemmatized_stop = [x for x in review_lower_stop_
pun_extra_lemmatized\
                                                  if not x in cachedStopWords]
    text = " ".join(review_lower_stop_pun_extra_lemmatized_stop)
    #### Do extra cleaning remove \n sign
    text = text.replace('\n','')
    text = text.replace('[removed]','')
    text= text.replace('[deleted]','')
    return (text)

def detect_polarity(text):
    return TextBlob(text).sentiment.polarity
```

In [ ]:

```

```

1.SuicideWatch

In [70]:

```
### Because of relatively huge dataset, we need to perform random sampling of 50
% for now
sampleSuicideWatch_list = SuicideWatch_df.sample(frac=0.5, replace=True, random_
state=1)
```

In [71]:

```
sampleSuicideWatch_list["title_with_selftext_clean"] = sampleSuicideWatch_list["
title_with_selftext"].apply(lambda x: clean_words(x))
```

In [72]:

```
#Print out the example cases
```

In [73]:

```
sampleSuicideWatch_list.tail(1)
```

Out[73]:

	author	over_18	title	selftext	num_comments	score	
45942	arialamia	False	coping with genital self harm	Since I can't ever be sure of anything I post ...	4	1	https://www.reddit.com/r/Suic

In [74]:

```
sampleSuicideWatch_list["title_with_selftext_clean"].tail(3).tolist()
```

Out[74]:

```
['keep going removed',
 'ending misery hello  year old walmart employee losing vision kerat
oconus fading memory gerd anxiety disorder habit smile mental pain b
ad childhood mother care almost dropped sophomore year truancy tryin
g deal adult problem believe seizure sleep could reason drool badly
happy anymore feel like putting body misery advice',
 'coping genital self harm since cant ever sure anything post would
ever appropriate self harm sub post insteadi know take much want thi
ng gone forever wish afab least intersex everyday hate much fucking
ugly shit destroyed much life permanently profoundly mangled body wa
nt one thing body thats validating thing physically harm sick disgus
ted see blood make feel real like foreign parasite part body tear fo
llow make feel like actually feelingsi want nightmare end care mean
ill never sexual experience another person peace far important somet
hing trite orgasm cosmic justice hate giant lumbering moron fucking
hopelessit make heart break never stop breaking']
```

In [75]:

```
sampleSuicideWatch_list['polarity'] = sampleSuicideWatch_list['title_with_selfte
xt_clean'].apply(detect_polarity)
```

In [76]:

```
sampleSuicideWatch_list.tail(3)
```

Out[76]:

	author	over_18	title	selftext	num_comments	score	
44562	SuchRound	False	How do you keep going	[removed]	0	1	https://www.reddit.com/r/
28371	LeanneJo	False	Ending my misery.	Hello. I'm a 19 year old Walmart employee. I'...	0	1	https://www.reddit.com/r/
45942	arialamia	False	coping with genital self harm	Since I can't ever be sure of anything I post ...	4	1	https://www.reddit.com/r/

In [77]:

```
# Check what is the most negative postings looks like
```

In [78]:

```
extreme_example =sampleSuicideWatch_list[sampleSuicideWatch_list['polarity']==-1
.0]
```

In [99]:

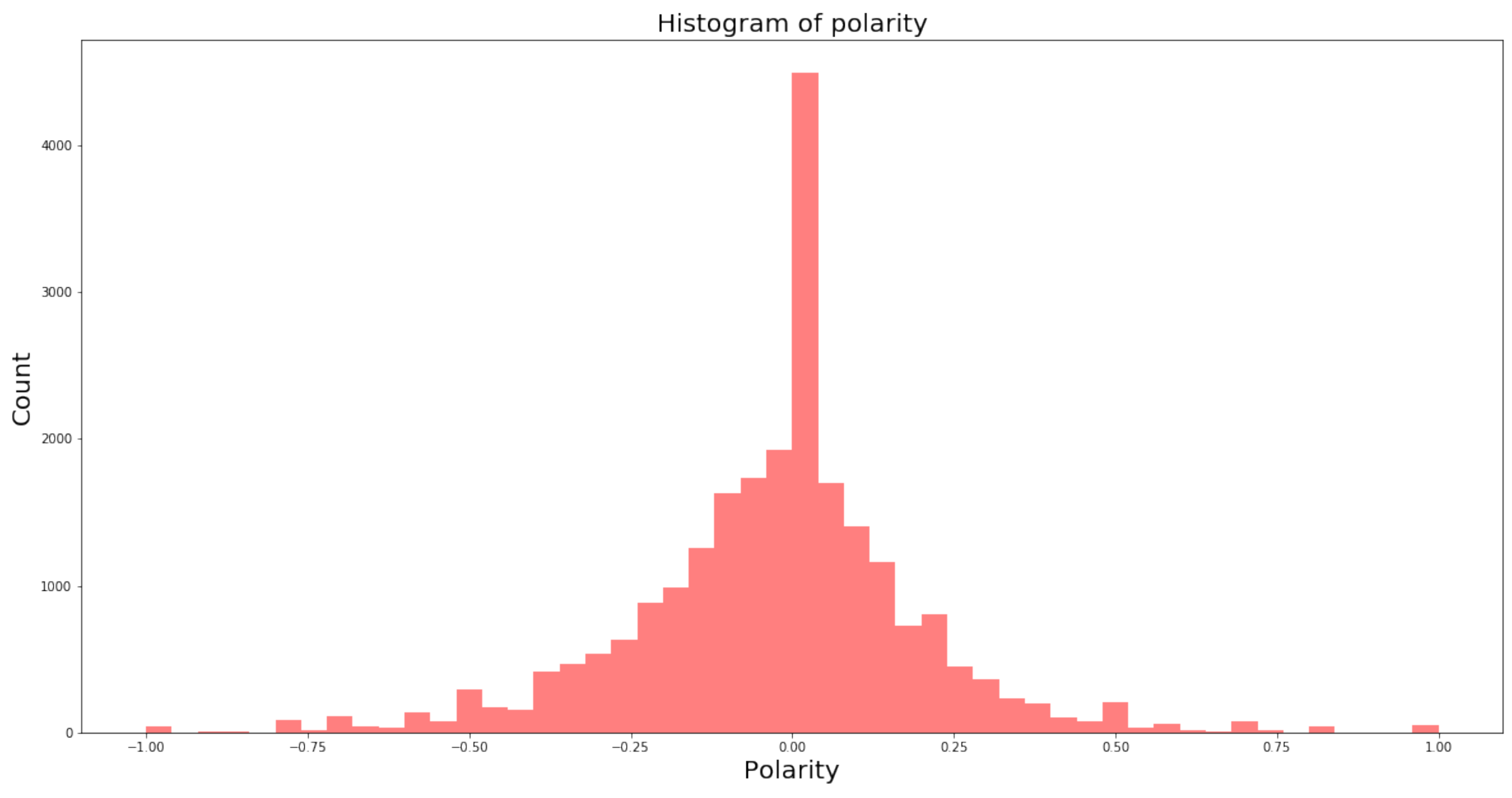
```
extreme_example['title'].head(5)
```

Out[99]:

```
2160                   Night terrors are terrifying
39630                     Literally want to give up
31608                          I’m a pathetic loser
33648    I can’t take it anymore I’m going insane
4338                                   im pathetic
Name: title, dtype: object
```

In [84]:

```
num_bins = 50
plt.figure(figsize=(20,10))
n, bins, patches = plt.hist(sampleSuicideWatch_list.polarity, num_bins, facecolo
r='red', alpha=0.5)
plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
plt.title('Histogram of polarity',fontsize =20)
plt.show()
```



2.Depressed

In [15]:

```
### Because of relatively huge dataset, we need to perform random sampling of 80
% for now
depressed_list = depressed_df.sample(frac=0.8, replace=True, random_state=1)
```

In [16]:

```
depressed_list["title_with_selftext_clean"] = depressed_list["title_with_selftex
t"].apply(lambda x: clean_words(x))
```

In [17]:

```
depressed_list.head(1)
```

Out[17]:

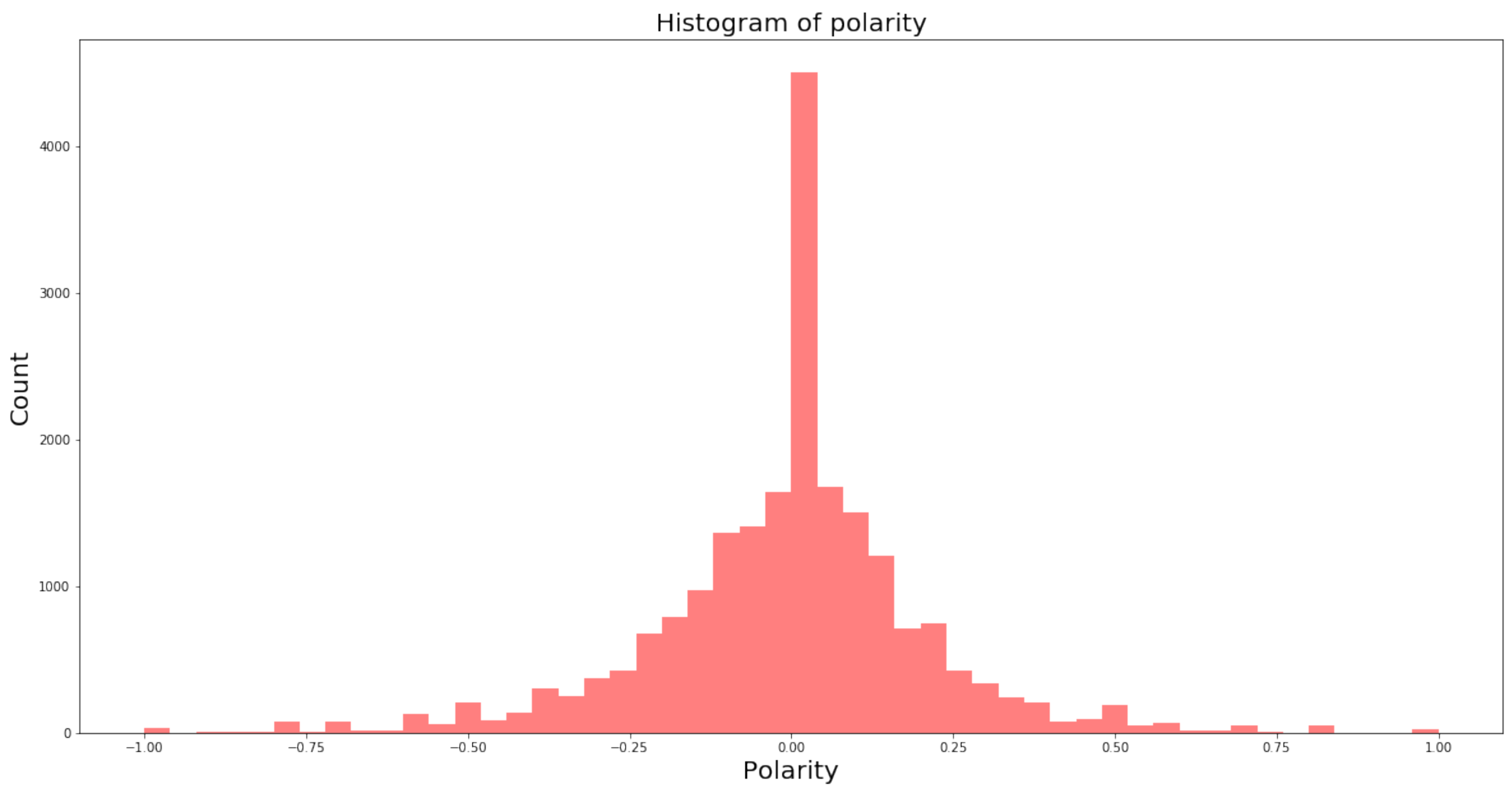
	author	over_18	title	selftext	num_comments	score	
236	L0st_W1sd0m	False	Just help me please	I just feel horrible constantly but I’m forced...	2	1	https://www.reddit.com/r/

In [18]:

```
depressed_list['polarity'] = depressed_list['title_with_selftext_clean'].apply(d
etect_polarity)
```

In [19]:

```
num_bins = 50
plt.figure(figsize=(20,10))
n, bins, patches = plt.hist(depressed_list.polarity, num_bins, facecolor='red',
alpha=0.5)
plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
plt.title('Histogram of polarity',fontsize =20)
plt.show()
```



3.Happy

In [20]:

```
### Because of relatively huge dataset, we need to perform random sampling of 50
% for now
happy_list = happy_df.sample(frac=0.5, replace=True, random_state=1)
```

In [21]:

```
happy_list["title_with_selftext_clean"] = happy_list["title_with_selftext"].appl
y(lambda x: clean_words(x))
```

In [22]:

```
happy_list.head(1)
```

Out[22]:

	author	over_18	title	selftext	num_comments	score	
33424	meatballsubreddit	False	I’m happy	[removed]	1	1	https://www.reddit.c

In [23]:

```
happy_list['polarity'] = happy_list['title_with_selftext_clean'].apply(detect_po
larity)
```

In [ ]:

```
##positive example
```

In [100]:

```
extreme_example =happy_list[happy_list['polarity']==1.0]
```

In [105]:

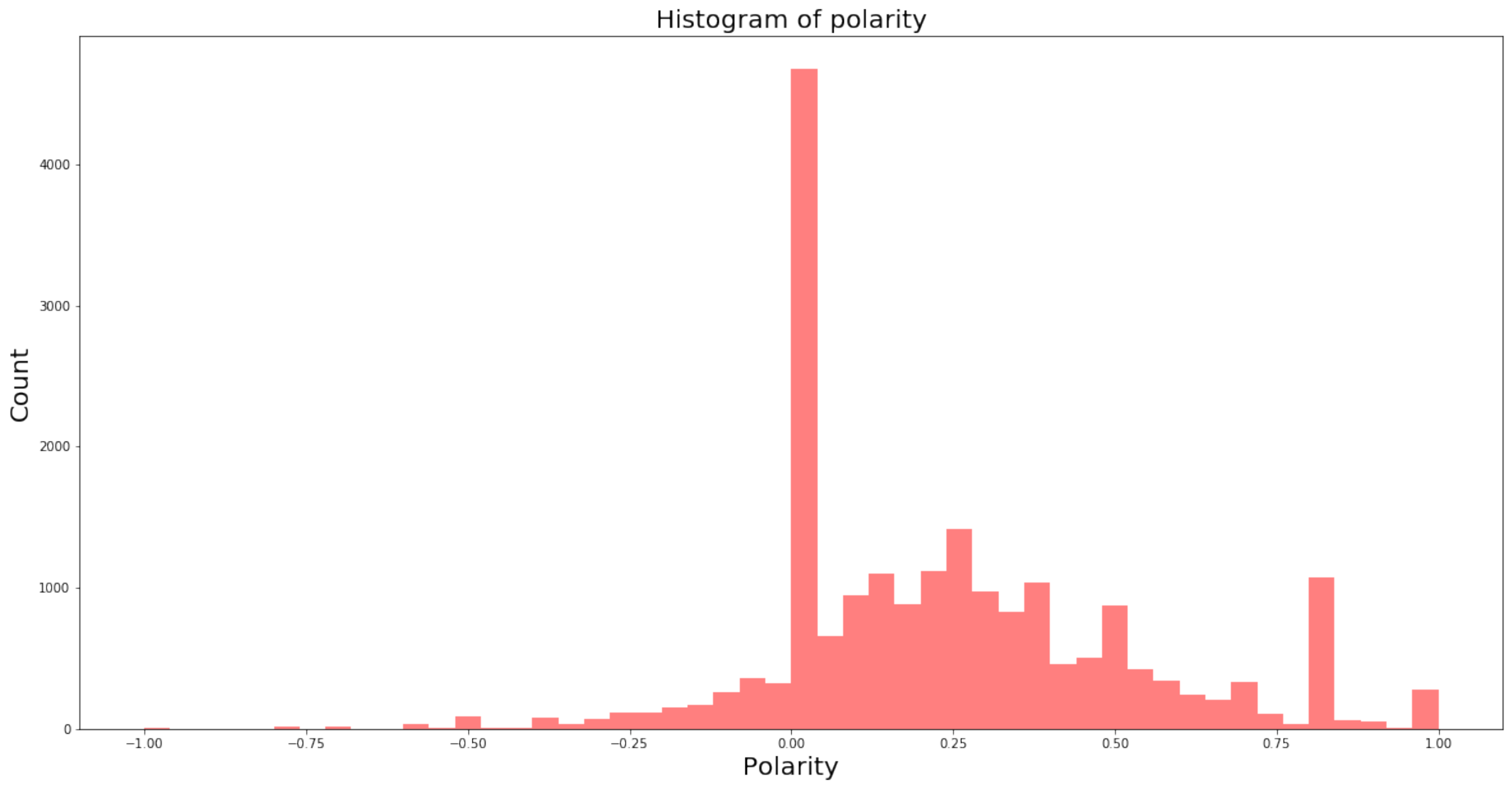
```
extreme_example['title'].tail(5).tolist()
```

Out[105]:

```
['Best motivational video to stay motivated',
 'Best friend rings',
 'Expecting our firstborn! During a time that seems hopeless, it’s w
onderful to have something to look forward to!',
 'I got my best friend back!!!',
 'My roommate and I just had the best laugh']
```

In [24]:

```
num_bins = 50
plt.figure(figsize=(20,10))
n, bins, patches = plt.hist(happy_list.polarity, num_bins, facecolor='red', alph
a=0.5)
plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
plt.title('Histogram of polarity',fontsize =20)
plt.show()
```



4.Self-improvement

In [25]:

```
### Because of relatively huge dataset, we need to perform random sampling of 50
% for now
selfimprovement_list = selfimprovement_df.sample(frac=0.5, replace=True, random_
state=1)
```

In [26]:

```
selfimprovement_list["title_with_selftext_clean"] = selfimprovement_list["title_
with_selftext"].apply(lambda x: clean_words(x))
```

In [27]:

```
selfimprovement_list.head(1)
```

Out[27]:

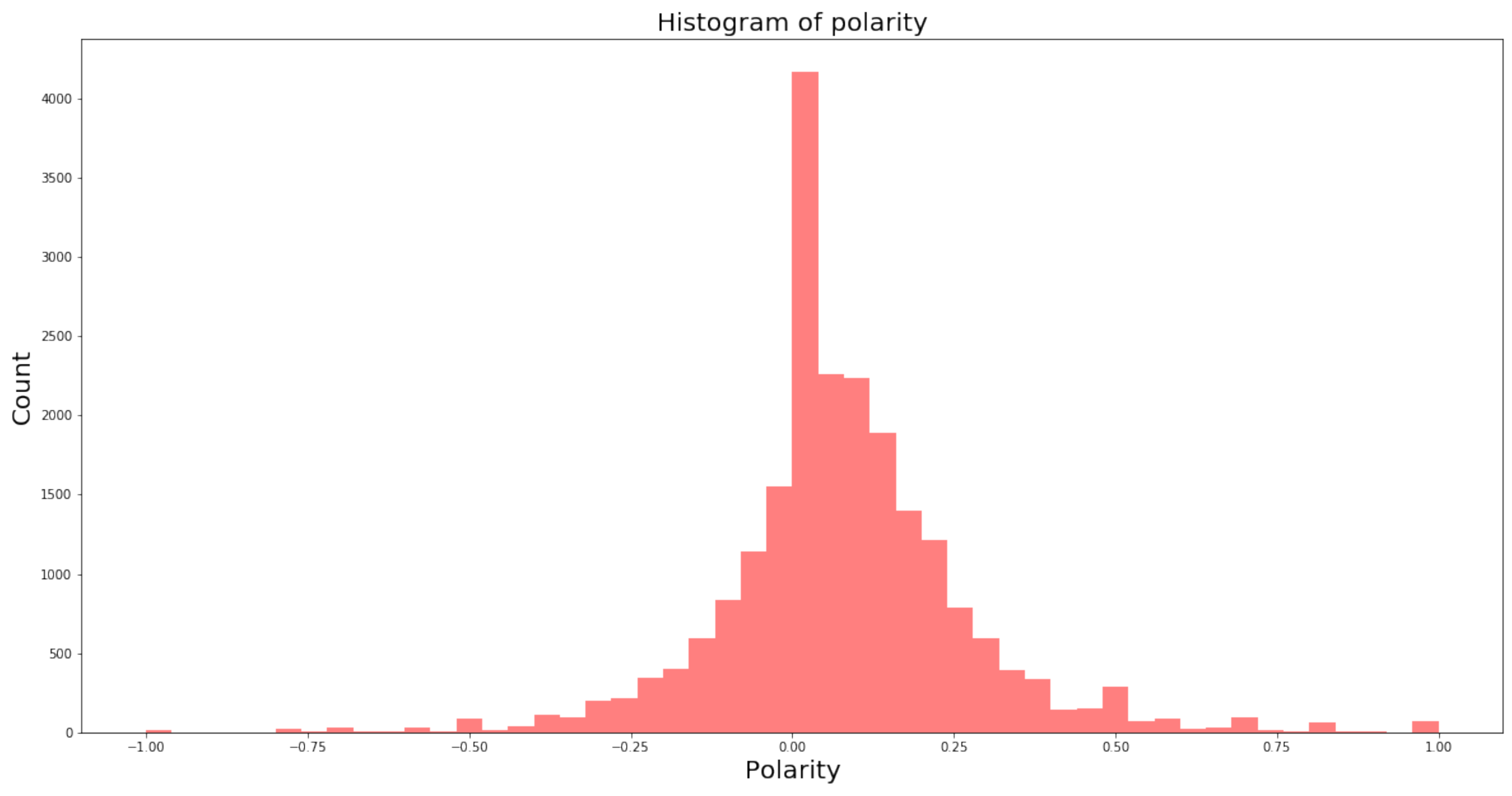
	author	over_18	title	selftext	num_comments	score	
33176	ParsnipParadise	False	Books or Podcasts for learning how to assert y...	I'm sitting here listening to Outliers by Malc...	0	1	https://www.reddit.c

In [28]:

```
selfimprovement_list['polarity'] = selfimprovement_list['title_with_selftext_cle
an'].apply(detect_polarity)
```

In [29]:

```
num_bins = 50
plt.figure(figsize=(20,10))
n, bins, patches = plt.hist(selfimprovement_list.polarity, num_bins, facecolor='
red', alpha=0.5)
plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
plt.title('Histogram of polarity',fontsize =20)
plt.show()
```



Polarity Descriptive Statistics comparison plot by 4 different subreddit groups

In [30]:

```
sampleSuicideWatch_list['polarity'].describe()
```

Out[30]:

```
count    23850.000000
mean        -0.025652
std          0.222685
min         -1.000000
25%         -0.126977
50%          0.000000
75%          0.081699
max          1.000000
Name: polarity, dtype: float64
```

In [31]:

```
depressed_list['polarity'].describe()
```

Out[31]:

```
count    21270.000000
mean        -0.005429
std          0.214168
min         -1.000000
25%         -0.100000
50%          0.000000
75%          0.100000
max          1.000000
Name: polarity, dtype: float64
```

In [32]:

```
selfimprovement_list['polarity'].describe()
```

Out[32]:

```
count    22098.000000
mean         0.075214
std          0.195665
min         -1.000000
25%         -0.006250
50%          0.060000
75%          0.166549
max          1.000000
Name: polarity, dtype: float64
```

In [33]:

```
happy_list['polarity'].describe()
```

Out[33]:

```
count    20470.000000
mean         0.237013
std          0.282833
min         -1.000000
25%          0.000000
50%          0.200000
75%          0.400000
max          1.000000
Name: polarity, dtype: float64
```

Polarity Distribution comparison plot by 4 different subreddit groups

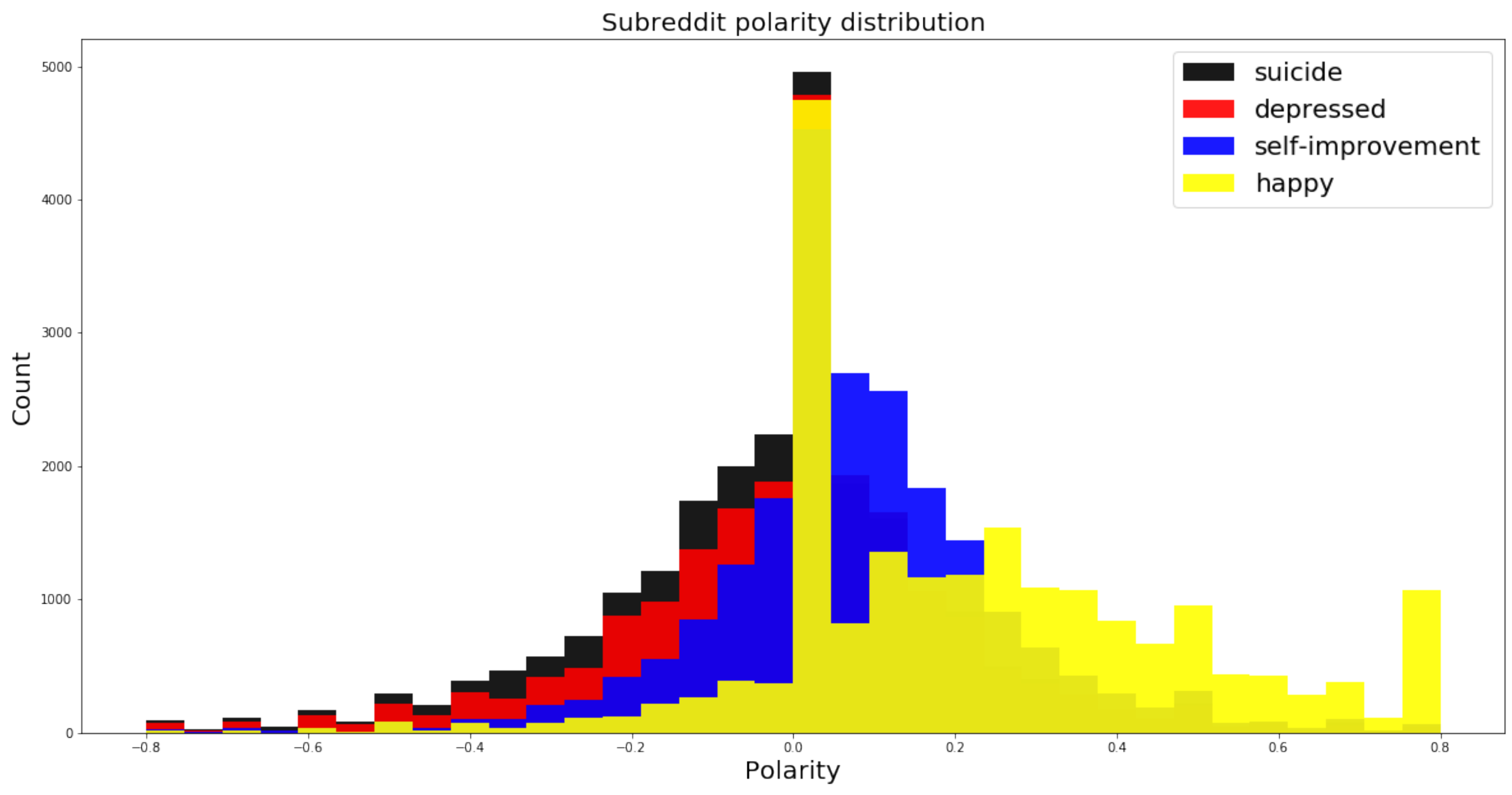
In [42]:

```
from matplotlib import pyplot
plt.figure(figsize=(20,10))
bins = np.linspace(-.8, .8, 35)

pyplot.hist(sampleSuicideWatch_list['polarity'], bins, alpha=0.9, label='suicide
',color='black')
pyplot.hist(depressed_list['polarity'], bins, alpha=0.9, label='depressed',color
='red')
pyplot.hist(selfimprovement_list['polarity'], bins, alpha=0.9, label='self-impro
vement',color='blue')
pyplot.hist(happy_list['polarity'], bins, alpha=0.9, label='happy',color='yellow
')

plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
pyplot.legend(loc='upper right',fontsize=20)
plt.title('Subreddit polarity distribution',fontsize =20)

pyplot.show()
```



In [ ]: