Report

NTU@India Connect Research Internship

Research Topic: **Immigration Reforms Sentiment Analysis**

Student details:

Name: Charvi Kusuma

Contents

Abstract

Can immigrants, who come in search of opportunity, freedom, and a better life, legally be permitted? Or worry that illegal entry into the country -- or even excessive legal entry -- will strain the economy and pose a threat to national security. Sentiment Analysis on the rise of anti-foreign/anti-immigrant sentiments all over the world, especially in the main immigrant-receiving countries like the United States, collected from Twitter to measure the inclination of people's opinions through text analysis and figure out how divided the opinions are on this issue. The report shall provide a clear visualization through various elements of the results obtained throughout the analysis.

Keywords: Sentiment Analysis, Immigration reforms, Twitter Data, Natural Language Processing, SenticNet APIs.

Introduction

The topic of immigration has always surfaced as a priority and a concern for many countries across the globe. Immigration which is the movement from one's country to another where you don't possess citizenship. It can either be legal or illegal depending on the channels one has gone through to migrate. It can contribute hugely to the ever-expanding diversity and economy of any given country. Immigration reforms are crucial since they can help a country move towards a unified nation.

Sentiment Analysis is a data analysis based on decision making techniques, which measures the inclination of people's opinions through Natural Language Processing (NLP), computational linguistics and text analysis. This technique is used to extract and analyse subjective information from the Web, mostly social media and similar sources. Since social media in one of the biggest sources of data, this document explores how Sentiment Analysis on text tweets can contribute to our implementation. Moreover, Anti-immigrant groups and activists use social media to demonize immigrants and get their messages across to the public.

The research focused on the understanding of anti-immigrant sentiment has previously been explored by various institutes and researchers, especially in the domain of economics and drawing up political decisions for taking the next step in reformation. The literature review will briefly describe the experiments conducted on this topic and details about the results drawn by them will help us better understand the procedures they followed.

Literature Review

The proliferation of social media platforms changed the way people interact online. Immigration is a long-simmering issue in the politics of many countries, including the United States. Several research experiments and articles have been posted on this topic. Below are a few examples which motivates to crawl and analyse the publicly available tweets based on the differences that are created, or the highlighted interests among people etc.

Pitropakis et. al. [1] collected and manually annotated an immigration-related dataset of tweets in UK, US, and Canadian English, explored anti-immigration speech detection utilizing various language features (word n-grams, character n-grams) and measured their impact on a number of trained classifiers. They also referred to the presence of anti-immigration sentiments both in private and public, are motivated by several reasons such as terrorist attacks being one of the extreme examples, often having profound influence on rising anti-immigration sentiments [2]. Politics and influencing factors, which not only can spark debates about immigration, but also openly display anti-immigration attitudes. They trigger an intense public debate, which in modern times are displayed through social networks. This leads us to social networking sites, such as Twitter which, distributes its data freely via Application Programming Interfaces (APIs).

Another study[3], analysed real public Twitter data, limited to anti-Islamic hashtag-related rhetoric. This work reveals that anti-Islamic sentiments were articulated in terms of complex identities referring not only to religion but also to ethnicity, politics, and gender. Basile et al. [4] examined both English and Spanish Tweets for anti-immigration rhetoric without focusing on any specific country under the title Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. Where the task was organized in two related classification subtasks, a main binary subtask for detecting the presence of hate speech, and a finer-grained one devoted to identifying further features in hateful contents such as the aggressive attitude and the target harassed, to distinguish if the incitement is against an individual rather than a group.

This report differentiates from previous efforts, as more of a data mining capability to investigate the expression of online users' opinions, by crawling and analysing publicly available Tweets. With a motivation to understand and visualize clearly the factors that are influencing the immigration laws, how divided are the opinions of the public and gain insights regarding the sentiments.

Furthermore, a 2007 Pew poll found that three-quarters of all U.S. citizens want to further restrict immigration. But what's behind such strongly held views?[5] Conventional wisdom holds that American attitudes toward immigrants are shaped by both economic and cultural considerations. In trying to explain the economic concerns of U.S. citizens, social scientists have pointed to few forms self-interest: Fear over increased competition for jobs, and resentment over having to pay for the social services used by immigrants and their families.

Another public-opinion research experiment by MIT political scientist Jens Hain Mueller and his Harvard colleague Michael Hiscox paints a very different picture. American citizens, they find, are not necessarily afraid of job competition or supporting public services. Instead, the striking thing about Americans' attitude toward immigration is that they collectively tend to prefer immigrant workers with refined job skills instead of those lacking good training: People seem to be much more in favour of high-skill immigrants because they think they contribute more to society. As a practical matter, that insight motivates us to identify the broad classification in which the topic can falls under. The results appear in a new paper, "Attitudes Toward Highly Skilled and Low Skilled Immigration: Evidence from a Survey Experiment,"[6].

Methodology

Data Collection and Preparation

Crawling the data from twitter using python's tweepy over 10k tweets posted since 2019. The hashtags that were included are: #immigration, #visa, #studentvisa, #immigrants, #immigrationlaw. Retweets and duplicated tweets were removed from the dataset. URLs were removed from the tweets.

Data Cleaning

The pre-processing of the text data is an essential step as it makes the raw text ready for mining, i.e., it becomes easier to extract information from the text. The objective of this step is to clean noise those are less relevant to find the sentiment of tweets such as punctuation, special characters, numbers, and terms which don't carry much weightage in context to the text. Data cleaning and pre-processing were performed on the collected Twitter data.

Mentions and hyperlinks were removed and the hex character (#) was removed from the hashtags. Made sure to keep only text after converting them to lower case and tokenized the raw text into tokens by taking into account phrases that are more commonly used in social media. Removal of meaningless words which do not occur in a dictionary. Cleaned the texts of all kinds of single letters and special characters. The tweets also contained lots of twitter handles (@user), that is how a Twitter user acknowledged on Twitter. These were also removed from the data as they don't convey much information. Duplicate tweets were then removed from the resulting dataset. After pre-processing, 4k unique tweets remained.

Sentiment Analysis

- ➢ Beginning with the visualization of the most frequent words from the dataset a word cloud is generated for a maximum of 1000 words, to understand the major contributing words in our dataset.
- ➢ Proceeding with the usage of the polarity classification API of SenticNet each tweet is labelled either positive, negative or neutral.
- ➢ To compare and differentiate the polarity labelling, Vader sentiment and Textblob polarity labelling were also used. Each of the value counts of the sentiment obtained from the various tools were plotted to visualize.
- ➢ Separate counts for the frequent words were visualized through bar graphs for each polarity.
- ➢ Majorly contributing Wordcounts for each polarity were plotted.
- ➢ Then utilizing SenticNet Concept Parsing API, we obtain the required concepts, segregate them based on the polarity into positive_conept_sets and negative_concept_sets respectively.
- ➢ Use the Gensim library which enables us to develop word embeddings by training our own word2vec models on the custom corpus.
- ➢ Finally understanding the differences of polarity by following the unsupervised approach of classification based on the concepts gathered from SenticNet.

Implementation

The complete setup and process flow of the project has been shown with source code and necessary outputs for an overview.

Crawling Dataset from twitter

```python
def printtweetdata(n, ith_tweet):
    print()
    print(f"Tweet {n}:")
    print(f"Tweet Text:{ith_tweet[0]}")


def scrape():
    db = pd.DataFrame(columns=['text'])
    tweets = tweepy.Cursor(api.search_tweets, q="#immigrationlaw", count=5000, lang="en",
                        since="2019-01-01",tweet_mode='extended').items()
    list_tweets = [tweet for tweet in tweets]
    i = 1
    for tweet in list_tweets:
        hashtags = tweet.entities['hashtags']
        try:
            text = tweet.retweeted_status.full_text
        except AttributeError:
            text = tweet.full_text
        hashtext = list()
        for j in range(0, len(hashtags)):
            hashtext.append(hashtags[j]['text'])
        ith_tweet = [text]
        db.loc[len(db)] = ith_tweet
        printtweetdata(i, ith_tweet)
        i = i+1
    filename = 'immigrationlaw_text_tweets.csv'
    db.to_csv(filename)
```

The query attribute was changed with different hashtags: #immigration, #visa, #studentvisa, #immigrants, #immigrationlaw.

```python
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_key, access_secret)
api = tweepy.API(auth)
scrape()
print('Scraping has completed!')
```

```
Output exceeds the size limit. Open the full output data in a text editor

Tweet 1:
Tweet Text:Good to see an attorney finally facing consequences for this well-known scam, even if there are
still plenty of others taking way too much from far too many to do the same. Ty to @sweetadelinevt for this
important story on one of the uglier open secrets of #immigrationlaw https://t.co/dHu3V2nFcb

Tweet 2:
Tweet Text:Omax Immigration Inc.
#immigration #visa #canada #studyabroad #ielts #immigrationlawyer #usa #studentvisa #immigrationconsultant
#canadaimmigration #immigrants #studyincanada #fypシ #canadavisa #studyvisa #immigrationlaw #workpermit
#citizenship #education #travel #greencard https://t.co/iJFbdgEnWd

Tweet 3:
Tweet Text:Sze Man “Angel” Lau, ’20, initially planned on returning to China to practice law, but changed
plans after finding a niche in #ImmigrationLaw. She passed the CA Bar Exam and started specializing in
immigration law in Sacramento. #McGeorgeInternational #LLM https://t.co/vFBOcBFb0k https://t.co/AQL0smN4Wf

Tweet 4:
Tweet Text:@MarshaBlackburn LegalImmigration,  Family based category #Legalimmigrants who are waiting for
many many years to enter to the US  #legally with respect to #Immigrationlaw from Front doors should Not be
#Abounded for illegal undocumented immigrants who came from walls borders backdoors illegaly.
https://t.co/zx6o1JPCpa
```

Separately merged all of the data collected to a single csv file keeping only the text column.

```python
df = pd.read_csv("merged.csv")
df
```

	text
0	In this video, I answer the question:\nApplyin...
1	Susan Rice and Ron Klain appear to be the prob...
2	Susan Rice and Ron Klain appear to be the prob...
3	Susan Rice and Ron Klain appear to be the prob...
4	Call us today for any of your immigration need...
...	...
21528	Local elections coming along on May 5th. Shoul...
21529	Disposal date for #Everify records has changed...
21530	Why is the government paying out performance b...
21531	Demography is destiny!\n\nThe maps below show ...
21532	Local elections coming along on May 5th. Shoul...

21533 rows × 1 columns

Text Pre-processing

Method for each text entry to remove @mentions, '#' symbols, Retweets, hyperlinks, stop words. And keep only meaningful words of text type in lower case.

```
def preprocess(raw_text):
    text_mention = re.sub(r'@[A-Za-z0-9]+',' ', raw_text) # Removed @mentions
    text_hashtag = re.sub(r'#', ' ', text_mention) #Removing the '#' symbol
    RT_text = re.sub(r'RT[\s]+', ' ', text_hashtag) # Removing RT
    text_http = re. sub(r'https?:\/\/\S+', ' ', RT_text)     # Remove the hyper link
    letters_only_text = re.sub("[^a-zA-Z0-9]", ' ', text_http)  # keep only specified text
    words = letters_only_text.lower().split()     # convert to lower case and split
    stopword_set = set(stopwords.words("english"))     # remove stopwords
    meaningful_words = [w for w in words if w not in stopword_set] #keep only meaningful
words
    cleaned_word_list = " ".join(meaningful_words)     # join the cleaned words in a list

    return cleaned_word_list
```

Definition to process each tweet:

```
def process_data(dataset):
    tweets_df = pd.read_csv(dataset)
    tweets_df = tweets_df.dropna()
    num_tweets = tweets_df.shape[0]
    print("Total tweets: " + str(num_tweets))
    cleaned_tweets = []

    print("Beginning pre-processing of tweets ")
    for i in range(num_tweets):
        cleaned_tweet = preprocess(tweets_df.iloc[i][0])
                   cleaned_tweets.append(cleaned_tweet)
        if(i % 100 == 0):
            print(str(i) + " tweets processed")
    print("Finished processing of tweets ")
    return cleaned_tweets
```

Drop all the duplicate tweets

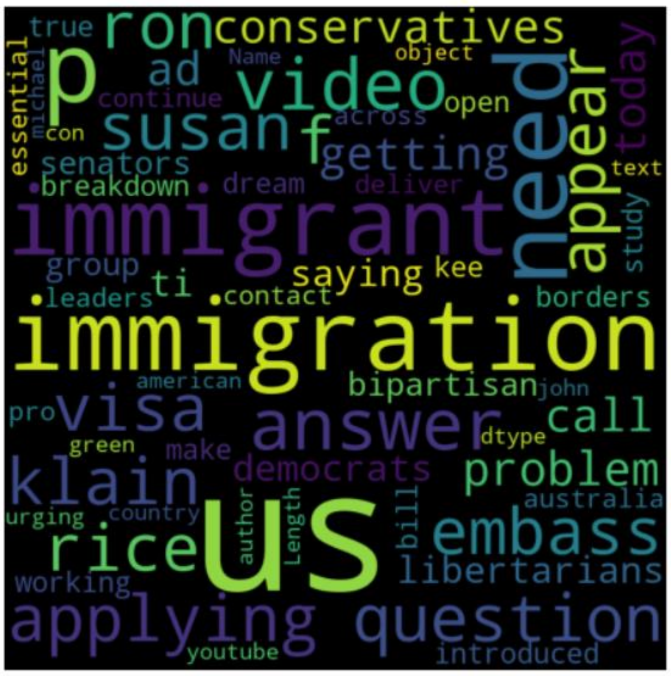
```
cleaned_data = process_data("merged.csv")
df = pd.DataFrame(cleaned_data)

df.drop_duplicates(subset=None, inplace=True)
df.to_csv('final_data.csv')
```

```
df = pd.read_csv("final_data.csv")
df
```

To get a visualization of the most frequent words on this topic:

```
stop_words = ['is','you','your','and', 'the', 'to', 'from', 'or', 'I', 'for', 'do', 'get',
'not',
             'here', 'in', 'im', 'have', 'on', 're', 'new', 'subject']
```

```
wordcloud = WordCloud(width = 800, height = 800, background_color = 'black', stopwords =
stop_words, max_words = 1000
                      , min_font_size = 20).generate(str(df['text']))

fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

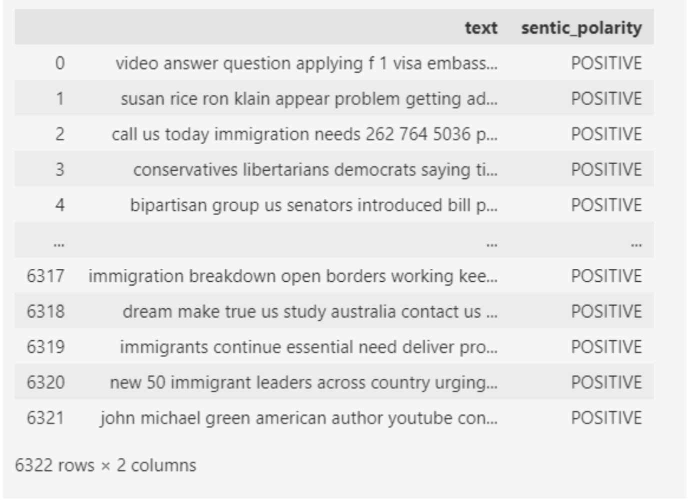


Use the Sentic Net Polarity Classification API

```
import requests

LANG = 'en'
APIKEY = '*********'
APIURL = 'https://sentic.net/api/' + LANG + '/' + APIKEY + '.py?text='

def process_data(tweets_df):
    tweets_df = tweets_df.dropna()
    num_tweets = tweets_df.shape[0]
    print("Total tweets: " + str(num_tweets))
    print("Beginning processing of tweets ")
    print(tweets_df.iloc[0][0])
    label_tweets=[]
    for i in range(num_tweets):
        label = polarity_api(tweets_df.iloc[i][0])
        label_tweets.append(label)
        if(i % 1000 == 0):
            print(str(i) + " tweets processed")
    print("Finished processing of tweets ")
    return label_tweets

def polarity_api(raw_text):
    text=raw_text
    for c in [';', '&', '#', '{', '}']: text = text.replace(c, ':')
    label = str(requests.get(APIURL + text).content)[2:-3]
    return label
labels = process_data(df)
df['sentic_polarity']=labels
df.head()
```

Report any encountered issues, e.g., mislabelled tweets

```
data=pd.read_csv('Polarity_Labelled_SenticNet.csv')
data
print(data['sentic_polarity'].unique())
print(data['sentic_polarity'].isna().sum())
```

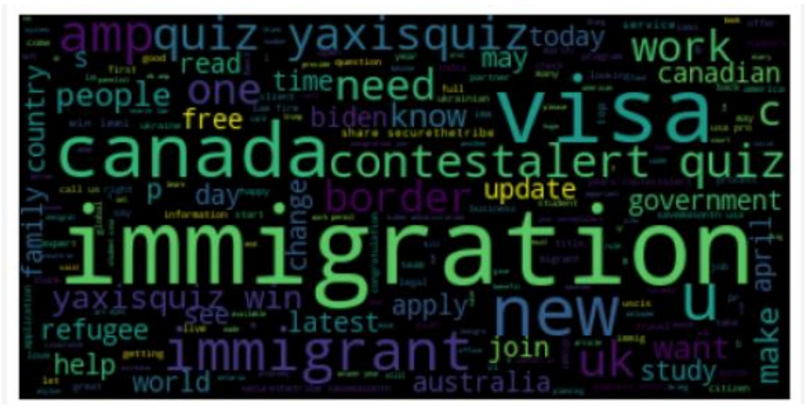


	text	sentic_polarity
0	video answer question applying f 1 visa embass...	POSITIVE
1	susan rice ron klain appear problem getting ad...	POSITIVE
2	call us today immigration needs 262 764 5036 p...	POSITIVE
3	conservatives libertarians democrats saying ti...	POSITIVE
4	bipartisan group us senators introduced bill p...	POSITIVE
...	...	...
6317	immigration breakdown open borders working kee...	POSITIVE
6318	dream make true us study australia contact us ...	POSITIVE
6319	immigrants continue essential need deliver pro...	POSITIVE
6320	new 50 immigrant leaders across country urging...	POSITIVE
6321	john michael green american author youtube con...	POSITIVE

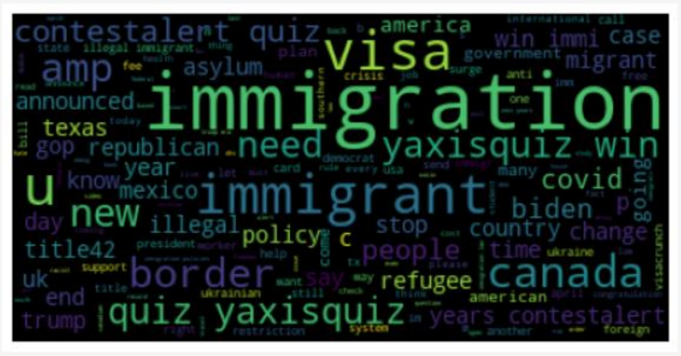
6322 rows × 2 columns

```
['POSITIVE' 'NEGATIVE' 'NEUTRAL']
0
```

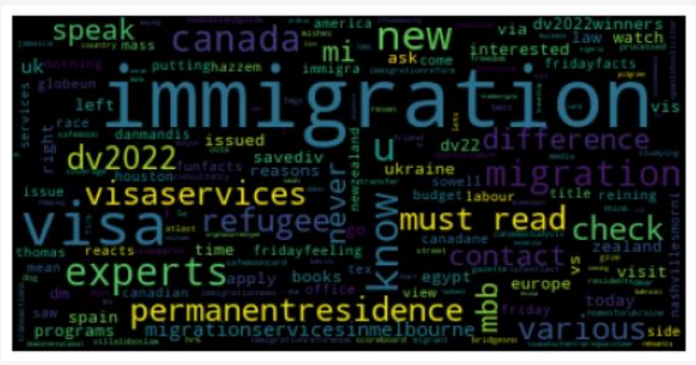
```
positive_sentiments = data[data['sentic_polarity']=='POSITIVE']
negative_sentiments = data[data['sentic_polarity']=='NEGATIVE']
neutral_sentiments = data[data['sentic_polarity']=='NEUTRAL']

wordcloud_pos =
WordCloud(background_color='black').generate(positive_sentiments['text'].to_string())
wordcloud_neg =
WordCloud(background_color='black').generate(negative_sentiments['text'].to_string())
wordcloud_neu =
WordCloud(background_color='black').generate(neutral_sentiments['text'].to_string())
fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud_pos)
plt.axis('off')
plt.show()
```



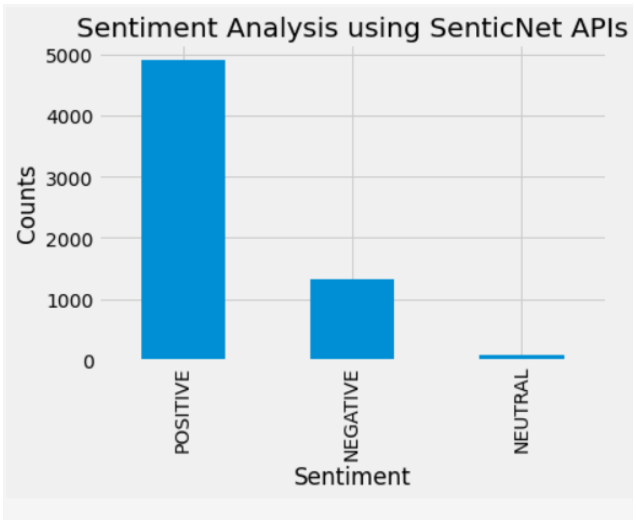
```
fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud_neg)
plt.axis('off')
plt.show()
```



```
fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud_neu)
plt.axis('off')
plt.show()
```



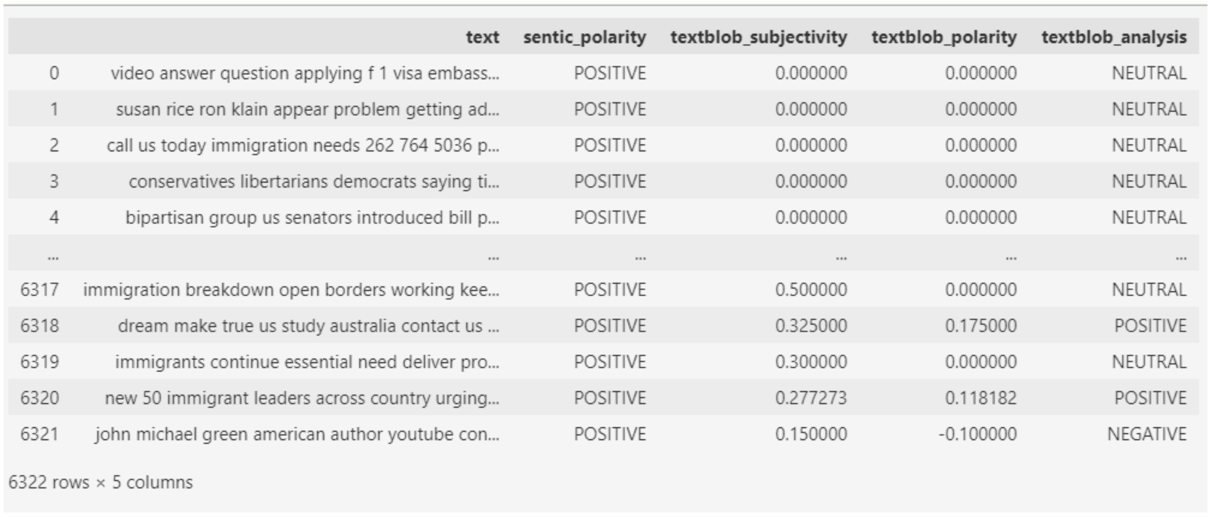
```
#Show the value counts
df['sentic_polarity'].value_counts()
#plot and visualize the counts
plt.title('Sentiment Analysis using Sentic Net APIs' )
plt.xlabel( 'Sentiment' )
plt.ylabel( 'Counts' )
df['sentic_polarity'].value_counts( ).plot(kind= 'bar')
plt.show( )
```



TextBlob Sentiment Analysis

Using **TextBlob** to calculate sentiment polarity which lies in the range of [-1,1] where 1 means positive sentiment and -1 means a negative sentiment.

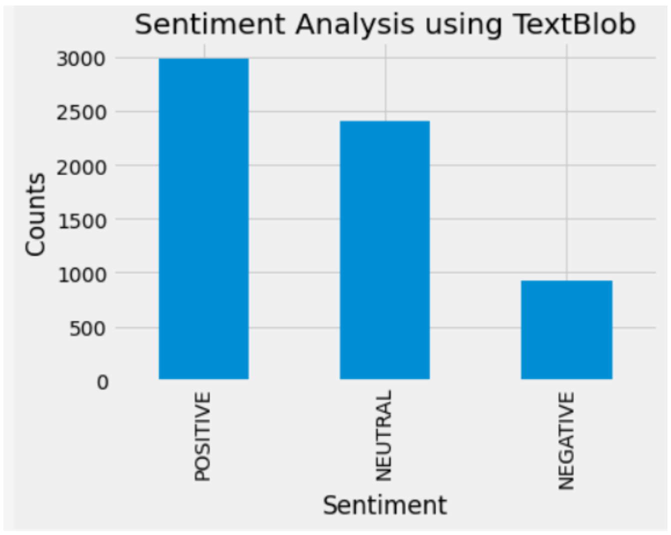
```
from textblob import TextBlob
# Create a function to get the subjectivity
def getSubjectivity (text):
    return TextBlob(text).sentiment.subjectivity

# Create a function to get the polarity
def getPolarity (text):
    return TextBlob(text).sentiment.polarity

#Create two new columns
data['textblob_subjectivity'] = data['text'].apply(getSubjectivity)
data['textblob_polarity'] = data['text'].apply(getPolarity)

#Create a function to compute the negative, neutral and positive analysis
def getAnalysis (score):
    if score < 0:
        return 'NEGATIVE'
    elif score == 0:
        return 'NEUTRAL'
    else: return 'POSITIVE'

data['textblob_analysis'] = data['textblob_polarity' ].apply(getAnalysis)
data
```



	text	sentic_polarity	textblob_subjectivity	textblob_polarity	textblob_analysis
0	video answer question applying f 1 visa embass...	POSITIVE	0.000000	0.000000	NEUTRAL
1	susan rice ron klain appear problem getting ad...	POSITIVE	0.000000	0.000000	NEUTRAL
2	call us today immigration needs 262 764 5036 p...	POSITIVE	0.000000	0.000000	NEUTRAL
3	conservatives libertarians democrats saying ti...	POSITIVE	0.000000	0.000000	NEUTRAL
4	bipartisan group us senators introduced bill p...	POSITIVE	0.000000	0.000000	NEUTRAL
...	...	...	...	...	...
6317	immigration breakdown open borders working kee...	POSITIVE	0.500000	0.000000	NEUTRAL
6318	dream make true us study australia contact us ...	POSITIVE	0.325000	0.175000	POSITIVE
6319	immigrants continue essential need deliver pro...	POSITIVE	0.300000	0.000000	NEUTRAL
6320	new 50 immigrant leaders across country urging...	POSITIVE	0.277273	0.118182	POSITIVE
6321	john michael green american author youtube con...	POSITIVE	0.150000	-0.100000	NEGATIVE

6322 rows × 5 columns

```python
#Show the value counts
data['textblob_analysis'].value_counts()
#plot and visualize the counts
plt.title('Sentiment Analysis using TextBlob' )
plt.xlabel( 'Sentiment' )
plt.ylabel( 'Counts' )
data['textblob_analysis'].value_counts( ).plot(kind= 'bar')
plt.show( )
```



```python
print('5 random reviews with the highest positive sentiment polarity: \n')
cl = data.loc[data.textblob_polarity == 1, ['text']].sample(5).values
for c in cl:
    print(c[0])
```

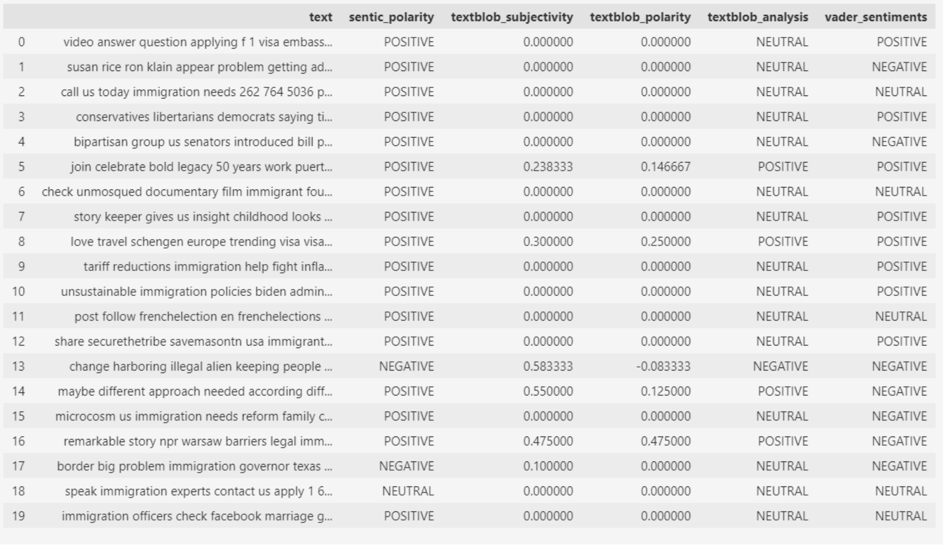
```
5 random reviews with the highest positive sentiment polarity:


awesome lgbtq immigrants autonomy
dubai city united arab emirate uae best tourist destination dubai dubaivisit rehmantravels visa visas islamabad pakistan
created privet group facebook people wanting learn ask questions join get hyper paycard invest best carpooling visa virtualcard
tech investingtips cryptotrading currency currencywallet
canada clicks bonus points whoever guess name marvelous place canada immigration immigrationcanada immigrationlawyer
immigratetocanada immigrationconsultant zsrimmigration
awesome idea immigration
```

Vader Sentiment

```python
import seaborn as sns
plt.style.use('fivethirtyeight')
cp = sns.color_palette( )
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()
```

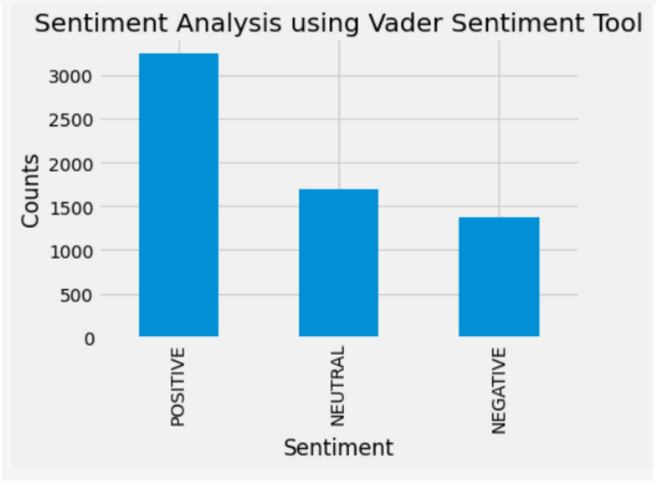
```python
emptyline=[]
for row in data ['text']:
    vs=analyzer.polarity_scores(row)
    emptyline. append(vs)

# Creating new dataframe with sentiments
df_sentiments=pd.DataFrame(emptyline)
df_sentiments.head()
```

```python
# Convert scores into positive and negative sentiments using some threshold
vader_sentiments = []
for i in df_sentiments.index:
    if df_sentiments['compound'][i] > 0:
       vader_sentiments.append('POSITIVE')
    elif df_sentiments['compound'][i] <0:
        vader_sentiments.append('NEGATIVE')
    else: vader_sentiments.append('NEUTRAL')
data['vader_sentiments']=vader_sentiments
data.head(20)
```



	text	sentic_polarity	textblob_subjectivity	textblob_polarity	textblob_analysis	vader_sentiments
0	video answer question applying f 1 visa embass...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
1	susan rice ron klain appear problem getting ad...	POSITIVE	0.000000	0.000000	NEUTRAL	NEGATIVE
2	call us today immigration needs 262 764 5036 p...	POSITIVE	0.000000	0.000000	NEUTRAL	NEUTRAL
3	conservatives libertarians democrats saying ti...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
4	bipartisan group us senators introduced bill p...	POSITIVE	0.000000	0.000000	NEUTRAL	NEGATIVE
5	join celebrate bold legacy 50 years work puert...	POSITIVE	0.238333	0.146667	POSITIVE	POSITIVE
6	check unmosqued documentary film immigrant fou...	POSITIVE	0.000000	0.000000	NEUTRAL	NEUTRAL
7	story keeper gives us insight childhood looks ...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
8	love travel schengen europe trending visa visa...	POSITIVE	0.300000	0.250000	POSITIVE	POSITIVE
9	tariff reductions immigration help fight infla...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
10	unsustainable immigration policies biden admin...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
11	post follow frenchelection en frenchelections ...	POSITIVE	0.000000	0.000000	NEUTRAL	NEUTRAL
12	share securethetribe savemasontn usa immigrant...	POSITIVE	0.000000	0.000000	NEUTRAL	POSITIVE
13	change harboring illegal alien keeping people ...	NEGATIVE	0.583333	-0.083333	NEGATIVE	NEGATIVE
14	maybe different approach needed according diff...	POSITIVE	0.550000	0.125000	POSITIVE	NEGATIVE
15	microcosm us immigration needs reform family c...	POSITIVE	0.000000	0.000000	NEUTRAL	NEGATIVE
16	remarkable story npr warsaw barriers legal imm...	POSITIVE	0.475000	0.475000	POSITIVE	NEGATIVE
17	border big problem immigration governor texas ...	NEGATIVE	0.100000	0.000000	NEUTRAL	NEGATIVE
18	speak immigration experts contact us apply 1 6...	NEUTRAL	0.000000	0.000000	NEUTRAL	NEUTRAL
19	immigration officers check facebook marriage g...	POSITIVE	0.000000	0.000000	NEUTRAL	NEUTRAL

```
#Show the value counts
data['vader_sentiments'].value_counts()
#plot and visualize the counts
plt.title('Sentiment Analysis using Vader Sentiment Tool' )
plt.xlabel( 'Sentiment' )
plt.ylabel( 'Counts' )
data['vader_sentiments'].value_counts( ).plot(kind= 'bar')
plt.show( )
```



```
N = 3
ind = np.arange(N)
width = 0.25

xvals =
[(data['sentic_polarity']=='POSITIVE').sum(),(data['vader_sentiments']=='POSITIVE').sum()
,(data['textblob_analysis']=='POSITIVE').sum()]
bar1 = plt.bar(ind, xvals, width, color = 'g')

yvals =
[(data['sentic_polarity']=='NEGATIVE').sum(),(data['vader_sentiments']=='NEGATIVE').sum()
,(data['textblob_analysis']=='NEGATIVE').sum()]
bar2 = plt.bar(ind+width, yvals, width, color='r')

zvals =
[(data['sentic_polarity']=='NEUTRAL').sum(),(data['vader_sentiments']=='NEUTRAL').sum()
,(data['textblob_analysis']=='NEUTRAL').sum()]
bar3 = plt.bar(ind+width*2, zvals, width, color = 'b')

plt.xlabel("Sentiment")
plt.ylabel('Counts')
plt.title("Sentiment Analysis")

plt.xticks(ind+width,['Sentic Net API', 'Vader Sentiment', 'TextBlob'])
plt.legend( (bar1, bar2, bar3), ('Positive', 'Negative', 'Neutral') )
plt.show()
```

<this plot is shown in error analysis section >

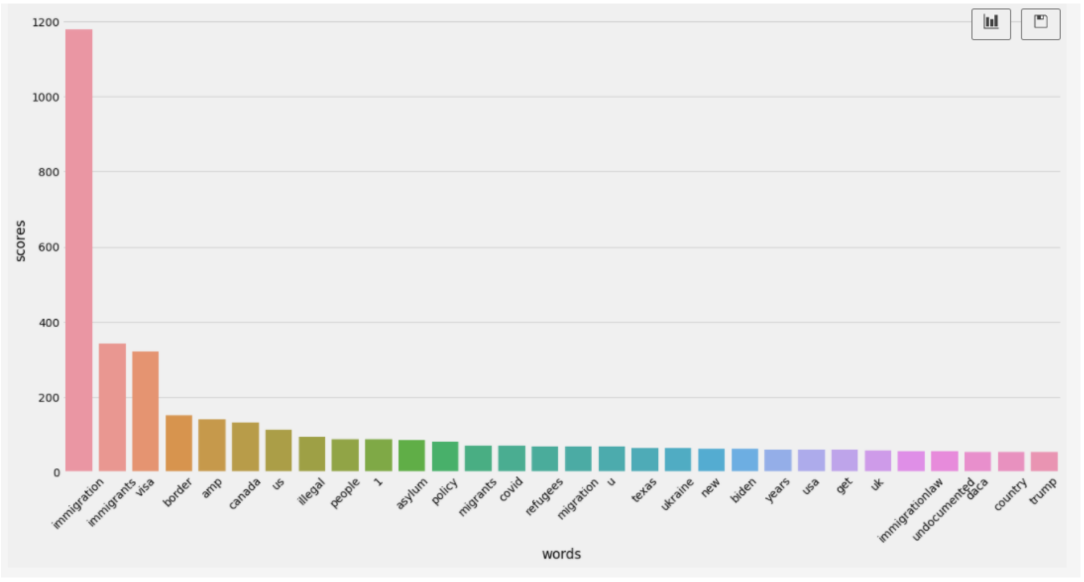
```
pos_tweet_sentic = data[data['sentic_polarity'] == 'POSITIVE']
neg_tweet_sentic = data[data['sentic_polarity'] == 'NEGATIVE']
neu_tweet_sentic = data[data['sentic_polarity'] == 'NEUTRAL']

pos_tokens_sentic = [token for line in pos_tweet_sentic['text'] for token in line.split()]
```

```
neg_tokens_sentic = [token for line in neg_tweet_sentic['text'] for token in line.split()]
neut_tokens_sentic = [token for line in neu_tweet_sentic['text'] for token in line.split()]

# Get Most Commonest Keywords
from collections import Counter
def get_tokens(docx,num=30):
    word_tokens = Counter(docx)
    most_common = word_tokens.most_common(num)
    result = dict(most_common)
    return result
get_tokens(pos_tokens_sentic)

most_common_pos_words = get_tokens(pos_tokens_sentic)
most_common_neg_words = get_tokens(neg_tokens_sentic)
most_common_neut_words = get_tokens(neut_tokens_sentic)
```

```
neg_df = pd.DataFrame(most_common_neg_words.items(),columns=['words','scores'])
pos_df = pd.DataFrame(most_common_pos_words.items(),columns=['words','scores'])
neut_df = pd.DataFrame(most_common_neut_words.items(),columns=['words','scores'])
```

Majorly contributing Wordcounts for negative polarity

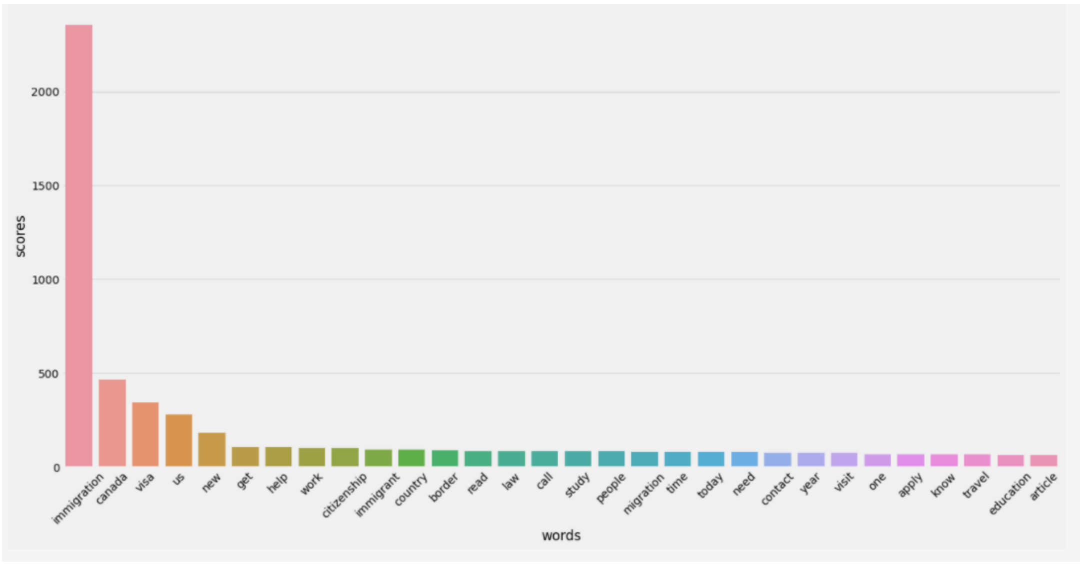
```
plt.figure(figsize=(20,10))
sns.barplot(x='words',y='scores',data=neg_df)
plt.xticks(rotation=45)
plt.show()
```



```
# Plot with seaborn
pos_df = pd.DataFrame(most_common_pos_words.items(),columns=['words','scores'])
plt.figure(figsize=(20,10))
sns.barplot(x='words',y='scores',data=pos_df)
plt.xticks(rotation=45)
plt.show()
```

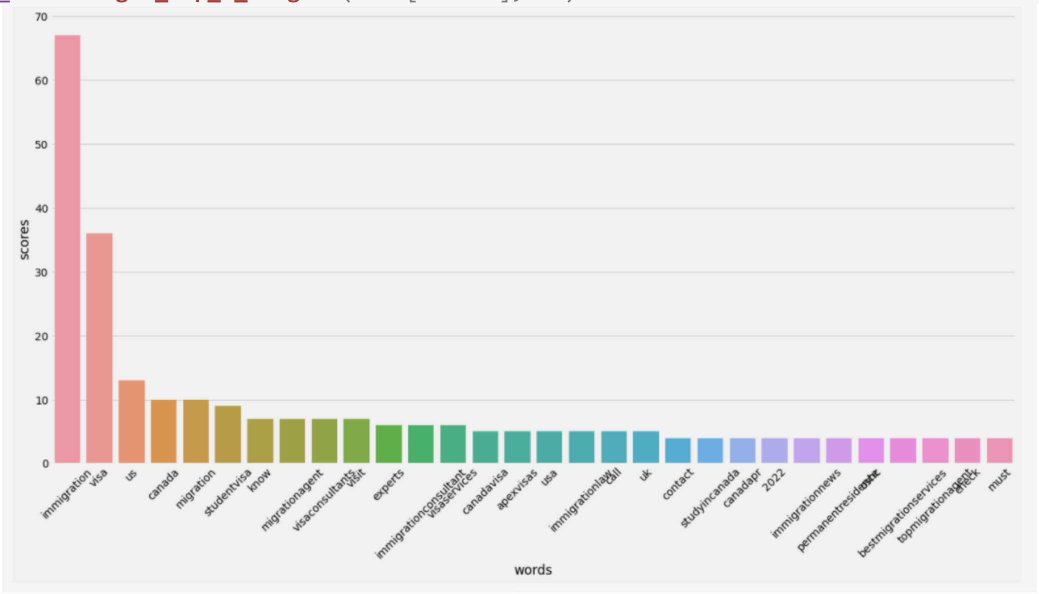
Majorly contributing Wordcounts for positive polarity

```python
# Plot with seaborn
neut_df = pd.DataFrame(most_common_neut_words.items(),columns=['words','scores'])
plt.figure(figsize=(20,10))
sns.barplot(x='words',y='scores',data=neut_df)
plt.xticks(rotation=45)
plt.show()
```

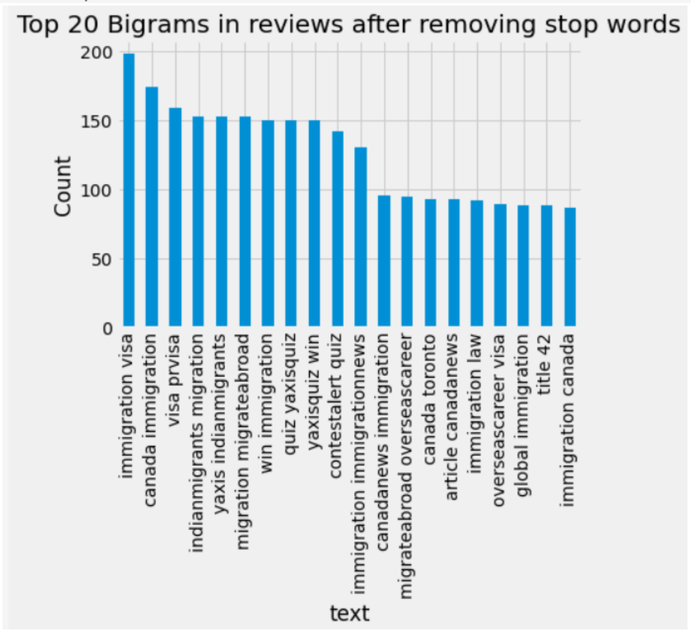


Majorly contributing Wordcounts for neutral polarity

```python
def get_top_n_trigram(corpus, n=None):
    vec = CountVectorizer(ngram_range=(2, 2), stop_words='english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]
common_words = get_top_n_trigram(data['text'], 20)
```



```
for word, freq in common_words:
    print(word, freq)
data1 = pd.DataFrame(common_words, columns = ['text' , 'count'])
data1.groupby('text').sum()['co
unt'].sort_values(ascending=False).plot(
    kind='bar', ylabel='Count', title='Top 20 Bigrams in reviews after removing stop
words')
```

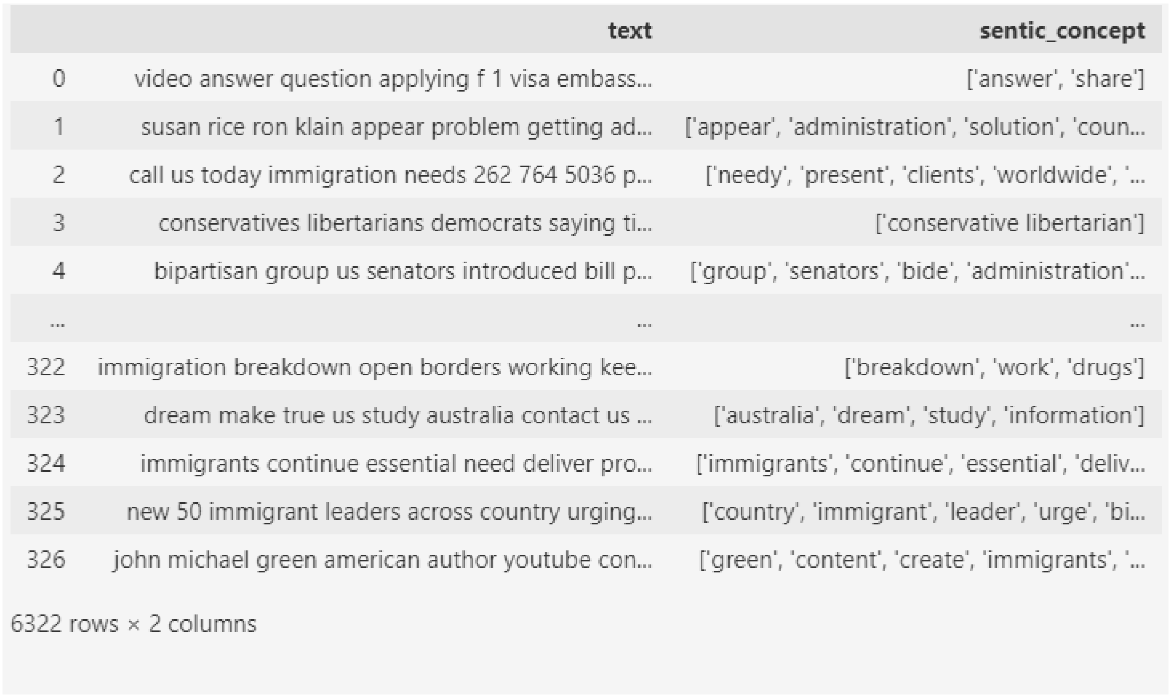


```
immigration visa 198
canada immigration 174
visa prvisa 159
yaxis indianmigrants 152
indianmigrants migration 152
migration migrateabroad 152
quiz yaxisquiz 150
yaxisquiz win 150
win immigration 150
contestalert quiz 142
immigration immigrationnews 130
canadanews immigration 95
migrateabroad overseascareer 94
article canadanews 93
canada toronto 93
immigration law 92
overseascareer visa 89
title 42 88
global immigration 88
immigration canada 86
```

Usage of Concept Parsing API to understand the labelling

Parsing in NLP is the process of determining the syntactic structure of a text by analysing its constituent words based on an underlying grammar. The purpose of this phase is to draw exact meaning, or we can say dictionary meaning from the text. Syntax analysis checks the text for meaningfulness comparing to the rules of formal grammar. It is the process of analysing a text which contains a sequence of tokens, to determine its grammatical structure with respect to a given grammar. The words are placed into distinct grammatical categories, and then the grammatical relationships between the words are identified, allowing us to interpret the sentence.

```
import requests
LANG = 'en'
APIKEY = '*********'
APIURL = 'https://sentic.net/api/' + LANG + '/' + APIKEY + '.py?text='

def process_data(tweets_df):
    tweets_df = tweets_df.dropna()
    num_tweets = tweets_df.shape[0]
    print("Total tweets: " + str(num_tweets))
    print("Beginning processing of tweets ")
    print(tweets_df.iloc[0][0])
    label_tweets=[]
    for i in range(num_tweets):
        label = concept_api(tweets_df.iloc[i][0])
        label_tweets.append(label)
        if(i % 100 == 0):
            print(str(i) + " tweets processed")
    print("Finished processing of tweets ")
    return label_tweets

def concept_api(raw_text):
    text=raw_text
    for c in [';', '&', '#', '{', '}']: text = text.replace(c, ':')
    label = str(requests.get(APIURL + text).content)[2:-3]
    return label

labels = process_data(df)
df['sentic_concept']=labels
df.head()
df.to_csv("Concept_Parsed_SenticNet.csv")
```



	text	sentic_concept
0	video answer question applying f 1 visa embass...	['answer', 'share']
1	susan rice ron klain appear problem getting ad...	['appear', 'administration', 'solution', 'coun...
2	call us today immigration needs 262 764 5036 p...	['needy', 'present', 'clients', 'worldwide', '...
3	conservatives libertarians democrats saying ti...	['conservative libertarian']
4	bipartisan group us senators introduced bill p...	['group', 'senators', 'bide', 'administration'...
...	...	...
322	immigration breakdown open borders working kee...	['breakdown', 'work', 'drugs']
323	dream make true us study australia contact us ...	['australia', 'dream', 'study', 'information']
324	immigrants continue essential need deliver pro...	['immigrants', 'continue', 'essential', 'deliv...
325	new 50 immigrant leaders across country urging...	['country', 'immigrant', 'leader', 'urge', 'bi...
326	john michael green american author youtube con...	['green', 'content', 'create', 'immigrants', '...

6322 rows × 2 columns

Mislabelled Tweets Analysis

➢ Unsupervised learning Approach

The unsupervised approach to sentiment analysis calling Semantic Similarity Analysis (SSA) consist firstly training a word embedding model using all the text tweets. The characteristics of this embedding space is the similarity between words in this space (Cosine similarity here) which is a measure of their semantic relevance. Next choosing two sets of words that carry positive and negative sentiments in the context space, in order to predict the sentiment of a review, we will calculate its similarity in the word embedding space to these positive and negative sets and see which sentiment the text is closest to.

The w2v_utils module contains all general utility functions

- Tokenizer is a class to clean, de-noise and tokenize the text.
- evaluate_model calculates F1, Accuracy, Recall, and Precision Score.
- w2v_trainer trains a Word2Vec model with tokenized documents and returns keyed vectors and keyed vocabs.
- overall_semantic_sentiment_analysis calculates the semantic sentiment of the text. It first computes a vector for the text (average of the wordvectors building the text document vector) and two vectors representing our given positive and negative lists of words and then calculates Positive and Negative Sentiment Scores as cosine similarity between the text vector and the positive and negative vectors respectively.
- topn_semantic_sentiment_analysis calculates the semantic sentiment of an list of tokenized text documents by using calculate_topn_similarity_score function.

```
# Importing functions and classes from utility module
from w2v_utils import (Tokenizer,
                       evaluate_model,
                       w2v_trainer,
                       overall_semantic_sentiment_analysis,
                       topn_semantic_sentiment_analysis,
)

# Instancing the Tokenizer class
tokenizer = Tokenizer(clean= True,
                      lower= True,
                      de_noise= True,
                      remove_stop_words= True,
                      keep_negation=True)
```

The Tokenizer class will handle all different tokenization options. This class has the following attributes: clean, lower, de_noise, remove_stop_words, and keep_negation. A nuance here is negation stopwords such as "not" and "no". Negation words are considered as *sentiment shifter* as they often change sentiment of the sentence in the opposite directions[7]. keep_neagation is True, the tokenizer will attach the negation tokens to the next token and treat them as a single word before removing the stopwords.

```
df = pd.read_csv("Polarity_Labelled_SenticNet.csv")
df['sentic_polarity'].replace(['POSITIVE','NEGATIVE','NEUTRAL'],[1,0,-1],inplace=True)
df.drop(df.loc[df['sentic_polarity']==-1].index, inplace=True)
df['sentic_polarity'].value_counts()
```

```
1    4921
0    1319
Name: sentic_polarity, dtype: int64
```

Gathering all the tweets concepts which were labelled positive by SenticNet APIs

```
pos_concepts_tweets=[]
for ind in df.index:
     if df['sentic_polarity'][ind]==1:
        pos_concepts_tweets.append(df['text'][ind])

df_concepts = pd.read_csv("Concept_Parsed_SenticNet.csv")

pos_concepts=[]
for i in pos_concepts_tweets:
    data_ex =df_concepts.loc[df_concepts['text'] == i, 'sentic_concept']
    data_ex.reset_index(drop=True, inplace=True)
    pos_concepts.append(data_ex[0].strip('[]').strip(' ').replace("'", "").replace("
","").split(','))



list_of_pos_words = []
for l in pos_concepts:
    for i in l:
        list_of_pos_words.append(i)
list_of_pos_words
```

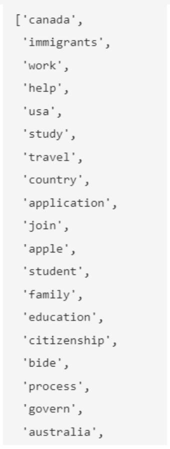
Similarly gathering all the tweets concepts which were labelled negative by SenticNet APIs.

```
neg_concepts_tweets=[]
for ind in df.index:
     if df['sentic_polarity'][ind]==0:
        neg_concepts_tweets.append(df['text'][ind])
```

```
neg_concepts=[]
for i in neg_concepts_tweets:
    data_ex =df_concepts.loc[df_concepts['text'] == i, 'sentic_concept']
    data_ex.reset_index(drop=True, inplace=True)
    neg_concepts.append(data_ex[0].strip('[]').strip(' ').replace("'", "").replace("
","").split(','))
neg_concepts
list_of_neg_words = []
for l in neg_concepts:
    for i in l:
        list_of_neg_words.append(i)
list_of_neg_words
```

Taking 50 most common words from the positive concepts

```
from collections import Counter
c_n = Counter(list_of_pos_words)
required = (([(l,k) for k,l in sorted([(j,i) for i,j in c_n.items()], reverse=True)][:50]))

req_pos_concepts = []
for i in required:
    req_pos_concepts.append(i[0])
req_pos_concepts
```



```
['canada',
 'immigrants',
 'work',
 'help',
 'usa',
 'study',
 'travel',
 'country',
 'application',
 'join',
 'apple',
 'student',
 'family',
 'education',
 'citizenship',
 'bide',
 'process',
 'govern',
 'australia',
```

Similarly taking 50 most common words from the negative concepts

```
c_n = Counter(list_of_neg_words)
required = (([(l,k) for k,l in sorted([(j,i) for i,j in c_n.items()], reverse=True)][:50]))

req_neg_concepts = []
for i in required:
    req_neg_concepts.append(i[0])
req_neg_concepts
```

```
['immigrants',
 'canada',
 'policy',
 'covid',
 'illegal',
 'country',
 'usa',
 'bide',
 'ukraine',
 'asylum',
 'dacer',
 'texas',
 'win',
 'mexico',
 'crime',
 'qui',
 'job',
 'policies',
 'work',
 'pandemic',
 'status',
 'govern',
 'free',
 'restrictions',
 'help',
```

Calculate cosine similarity in the word embedding space to these positive and negative concept sets.

```
pos_concepts = [concept for concept in req_pos_concepts if concept in keyed_vocab]
len(req_pos_concepts)
50
neg_concepts = [concept for concept in req_neg_concepts if concept in keyed_vocab]
len(req_neg_concepts)
50
df['tokenized_text'] = df['text'].apply(tokenizer.tokenize)
df['tokenized_text_len'] = df['tokenized_text'].apply(len)
df['tokenized_text_len'].apply(np.log).describe()
```

```
count    6240.000000
mean        2.875184
std         0.421326
min         0.693147
25%         2.639057
50%         2.995732
75%         3.178054
max         3.951244
Name: tokenized_text_len, dtype: float64
```

Word embedding is capable of capturing the meaning of a word in a document, semantic and syntactic similarity, relation with other words. Its input is a text corpus and its output is a set of vectors. Usage of gensim package to train the word2vec model also setting the vector_size to 300.

```
# Training a Word2Vec model
keyed_vectors, keyed_vocab = w2v_trainer(df['tokenized_text'],
                                         epochs=10,
                                         workers=3,
                                         vector_size=300,
                                         window=5,
                                         min_count=2)
```

Calculating the semantic sentiment of the tweets

As mentioned, in order to predict the sentiment of a tweet, we need to calculate its similarity to our negative and positive concept sets. Let these similarities be negative semantic score (NSS) and positive semantic scores (PSS) respectively. Building the document vector by averaging over the wordvectors, we will have a vector for every tweet and two vectors representing our positive and negative concept sets. The PSS and NSS are then calculated by a simple cosine similarity between the review vector and the positive and negative vectors respectively. Let's call this approach *Overall Semantic Sentiment Analysis* (**OSSA**).

```
# Calculating Semantic Sentiment Scores by OSSA model
overall_df_scores = overall_semantic_sentiment_analysis (keyed_vectors = keyed_vectors,
                                                         positive_target_tokens = pos_concepts,
                                                         negative_target_tokens = neg_concepts,
                                                         doc_tokens = df['tokenized_text'])
```

After calculating the positive and negative scores, we define

semantic_sentiment_score (S3) = positive_sentiment_score (PSS) - negative_sentiment_score (NSS)

If the S3 is positive, we can classify the review as positive, and if it is negative, we can classify it as negative.

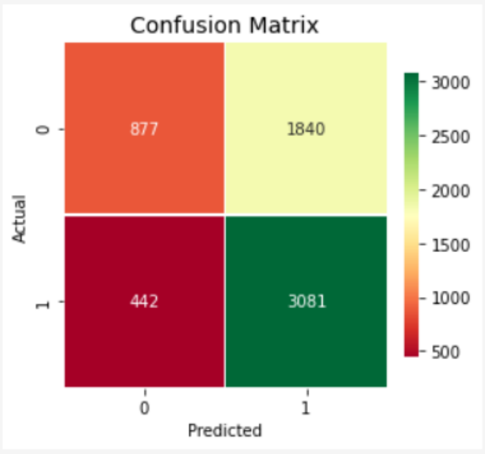
```
# To store semantic sentiment store computed by OSSA model in df
df['overall_PSS'] = overall_df_scores[0]
df['overall_NSS'] = overall_df_scores[1]
df['overall_semantic_sentiment_score'] = overall_df_scores[2]
df['overall_semantic_sentiment_polarity'] = overall_df_scores[3]
df
```

	text	sentic_polarity	tokenized_text	tokenized_text_len	overall_PSS	overall_NSS	overall_semantic_sentiment_score	overall_semantic_sentiment_polarity
0	video answer question applying f 1 visa embass...	1	[video, answer, question, applying, f, 1, visa...	21	0.954638	0.866853	0.087785	1
1	susan rice ron klain appear problem getting ad...	1	[susan, rice, ron, klain, appear, problem, get...	17	0.847782	0.882623	-0.034841	0
2	call us today immigration needs 262 764 5036 p...	1	[call, us, today, immigration, needs, 262, 764...	27	0.965740	0.836796	0.128944	1
3	conservatives libertarians democrats	1	[conservatives, libertarians, democrats,	7	0.809841	0.954525	-0.144685	0

Performance Metrics using y_test as OSSA model and y_pred as SenticNet polarity

```
y_test = df['overall_semantic_sentiment_polarity']
y_pred_SenticNet = df['sentic_polarity']
evaluate_model(y_true = y_test,
                         y_pred = y_pred_SenticNet,
                         report=True,
                         plot=True)
```

```
              precision    recall  f1-score   support

           0     0.6649    0.3228    0.4346      2717
           1     0.6261    0.8745    0.7297      3523

    accuracy                         0.6343      6240
   macro avg     0.6455    0.5987    0.5822      6240
weighted avg     0.6430    0.6343    0.6012      6240
```



```
df['sentic_polarity'].value_counts()
```

```
1    4921
0    1319
Name: sentic_polarity, dtype: int64
```

```
df['overall_semantic_sentiment_polarity'].value_counts()
```

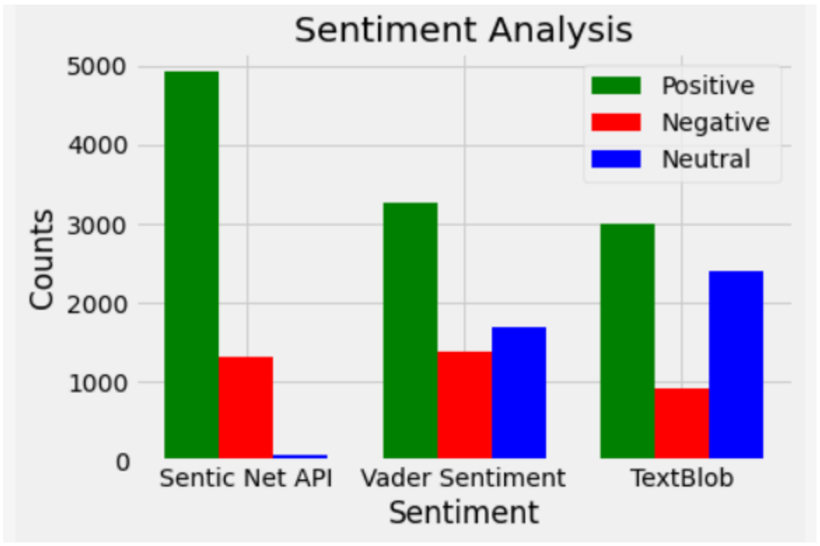
```
1    3478
0    2762
Name: overall_semantic_sentiment_polarity, dtype: int64
```

```
# OSSA Model Evaluation
print("OSSA Model Evaluation: ")
evaluate_model(df['sentic_polarity'],
               df['overall_semantic_sentiment_polarity'])

print("=======================")
```

```
OSSA Model Evaluation:
* Accuracy Score:  63.4295%
* F1 Score:  72.9749%
* Recall Score:  62.6092%
* Precision Score:  87.4539%
=======================
```

➢ Error Analysis:

Overall distribution of the three major Sentiment analysis tools



As from the above graphs we can notice quite a large variation among popular sentiment analysis tools vs SenticNet labelling.

- Vader Sentiment and Textblob have labelling frequencies similar to each other whereas for SenticNet most of the text data are predicted positive. This can be seen from the most_common_pos_words extracted for positive polarity text as: *{'immigration', 'visa', 'canada', 'us', 'amp', 'immigrants', 'new', 'call', 'get', 'study', 'uk', 'work', 'usa', 'immigrationlaw', 'visit', 'law', 'migration', 'today', 'help', 'studentvisa', 'read', 'studyabroad', 'travel', 'contact', 'join', 'apply', 'border', 'australia'}*

  However, words like visa, Canada, US/USA, Australia, contact, today and several others have neutral effect.

- *"Supportimmigrants", "jobsincanada", "workincanada", "settleincanada", "justiceforrefugees"* such hashtags convey a lot of meaning and can directly affect the sentiment polarity, however, identifying the individual words in these hashtags during concept parsing is a challenge. This is usually a preprocessing step to be done on the text, but if there is a model that could get the words separated before the concept parsing, it would highly impact the sentiment analysis.

- Tweets which are neutral sentiment from human understanding, for example *"Do the #immigration officers check #facebook at the #marriage #greencard interview?"* have been predicted positive. The concept parsed for this example is only ['marriage']. Such errors can be subjective to tweets, in this case it has been predicted neutral by both Vader and Textblob but as positive by SenticNet API.

- *"Such a bad #immigration experience at #Mumbai #airport, no sense of urgency; not enough officers, lines getting longer"*, is clearly a negative comment however, SenticNet has labelled it as POSITIVE using the concept parsed as ['mumbai', 'bad', 'airport', 'sense', 'urgency', 'longer', 'work', 'needy'].

  The word 'bad' is clearly negative.

- On a positive note, *"Canada relaxes temporary foreign worker program rules to address labour shortages"* tweets like these with mixed emotion words like "relaxes" and "shortage" in the concepts gave out a neutral polarity for Textblob and Vader, however, it is a positive sentiment in human understanding as well as SenticNet labelling.

- *"#GOP Blocks Senate COVID Bill, Demands Votes on #Immigration | loving this new GOP, thank you #Trump!"* is a completely sarcastic statement. Concept parsed: ['blocky', 'senate', 'covid', 'demand', 'loving', 'thank'], this tweet, even though there are words like thankyou and loving (positive) SenticNet predicted it as negative, whereas Vader and TextBlob gave it a positive polarity. Such examples of sarcastic statements are also a challenge, and depends on various factors such as emoticons (☹ : (, 😐 : |), flow of the sentence etc.

- Some of the tweets lost the semantic meaning in the preprocessing stage. The word order effects the meaning of the sentence.

- Removal of some of the stopwords is also a kind of data loss for statements which can be have Sarcasm, irony or euphism.

I have only shown the different scenarios that had an effect on the polarity taking only certain example tweets from the polarity.

Conclusion

Immigration sentiment analysis favourably revealed a positive polarity score for all of the different tools used, indicating that the tweets are biased or inclined towards acceptance or support towards immigration. The polarities may help derive an overall quality score, however, to get a clear understanding of the particular topics that are related to such a positive view on immigration we may have to categorize opinions by aspect and identify the sentiment related to each of them.

To summarize on the procedures followed in the entire course, **I have clearly visualized in terms of wordclouds, graphs and metrics wherever necessary to understand the differences in the opinion**. Further using the Gensim word2vec model applied an unsupervised learning method which is based on cosine similarity of positive and negative concepts parsed from SenticNet and the vectors of tweets. There has been quite a difference identified between both the models, that is the classification from SenticNet polarity and the word2vec model.

The variation in the polarity classified through various of these tools like TextBlob, Vader and Gensim word2vec model compared with SenticNet polarity classifications could be based on sarcasm detection, negation detection, word ambiguity or even multipolarity. Knowing about each of these will help us avoid possible problems and significantly increase sentiment analysis accuracy in a classification model.

References

[1] Monitoring Users' Behavior: Anti-Immigration Speech Detection on Twitter, Nikolaos Pitropakis, Kamil Kokot, Dimitra Gkatzia , Robert Ludwiniak, Alexios Mylonas and Miltiadis Kandias, 2020.

[2] Legewie, J. Terrorist events and attitudes toward immigrants: A natural experiment. Am. J. Sociol. 2013, 118, 1199–1245.

[3] Evolvi, G. # Islamexit: Inter-group antagonism on Twitter. Inf. Commun. Soc. 2019, 22, 386–401.

[4] i Orts, Ò.G. Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter at SemEval-2019 Task 5: Frequency Analysis Interpolation for Hate in Speech Detection. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 460–463.

[5] Understanding anti-immigrant sentiment, Peter Dizikes, MIT News Office, 2010.

[6] HAINMUELLER, J., & HISCOX, M. (2010). Attitudes toward Highly Skilled and Low-skilled Immigration: Evidence from a Survey Experiment. *American Political Science Review, 104*(1), 61-84. doi:10.1017/S0003055409990372.

[7] Bing Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*, Cambridge University Press 2015, pp. 116-122).