jupyter Australia Election Sentiment Analysis Last Checkpoint: 09/05/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Python 3 (ipykernel)

Run | Code

In [5]:
```
import nltk
import numba
import matplotlib
from nltk.stem import WordNetLemmatizer
from textblob import TextBlob
from nltk.util import ngrams
from nltk.metrics.distance import edit_distance
import nltk.classify.util
from nltk.probability import FreqDist
from nltk.classify import NaiveBayesClassifier
from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
import pandas as pd
nltk.download('punkt')
nltk.download('words')
from nltk.corpus import words
correct_spelling = words.words()
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Nachiket\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package words to
[nltk_data]     C:\Users\Nachiket\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!
```

In [ ]:
```
# DATA COLUMNS EXPLANATIONS

# created_at = Date on which the tweet was posted

# id = ID of the tweet

# full_text = Full tweet text

# retweet_count = Number of retweets

# favorite_count = Number of likes

# user_id = User ID of user who posted the tweet

# user_name = Username of user who posted the tweet

# user_screen_name = Screen name of user who posted the tweet

# user_description = Free text description on profile of user who posted the tweet

# user_location = Free text location on profile of user who posted the tweet
```

In [6]:
```
df = pd.read_csv("auspol2019.csv")
```

In [7]:
```
df.head()
```

Out[7]:

	created_at	id	full_text	retweet_count	favorite_count	user_id	user_name	user_screen_name	user_description	user_loca
0	2019-05-20 09:13:44	1130401208756187136	After the climate election: shellshocked green...	0.0	0.0	9.248486e+07	PIPELINEPETE	jocksjig	Retired Tradesman and Progressive Anti Conserv...	Brisb Queens
1	2019-05-20 09:13:43	1130401205367140357	@narendramodi @smritiirani Coverage of indian ...	0.0	0.0	7.756474e+08	Narinder Parmar	nparmar1957	Life coach & trainer, Motivational speaker, Ma...	Wollong N AUSTR
2	2019-05-20 09:13:33	1130401162782371841	@workmanalice Do you know if Facebook is relea...	0.0	0.0	5.687300e+04	Peter Wells	peterwells	Writes for @theage and @smh on technology and ...	Melbc
3	2019-05-20 09:13:29	1130401143551434753	@vanbadham We all understand we have a compuls...	0.0	0.0	9.081660e+17	The Realist	therealist822	Calls it as I see it. Anti PC, SJW and VS. If ...	
4	2019-05-20 09:13:23	1130401118666809345	Shares were mixed in Asia, with India and Aust...	0.0	0.0	5.260074e+08	Inquirer Business	InquirerBiz	The official Twitter account of the Inquirer G...	Philipp

In [8]:
```
df.describe()
```

Out[8]:

	retweet_count	favorite_count	user_id
count	183370.000000	183370.000000	1.833700e+05
mean	3.814310	11.159006	1.846078e+17
std	37.727466	118.324495	3.789751e+17
min	0.000000	0.000000	2.200000e+01
25%	0.000000	0.000000	1.002494e+08
50%	0.000000	1.000000	4.854974e+08
75%	1.000000	3.000000	2.981410e+09
max	6622.000000	15559.000000	1.130347e+18

In [11]:
```
#NUMBER OF WORDS IN EACH MESSAGE
for item in df['full_text']:
    print("Number of Words in each message: ", len(item))
```

```
Number of Words in each message:  93
Number of Words in each message:  121
Number of Words in each message:  159
Number of Words in each message:  214
Number of Words in each message:  170
Number of Words in each message:  84
Number of Words in each message:  162
Number of Words in each message:  234
Number of Words in each message:  102
Number of Words in each message:  82
Number of Words in each message:  197
Number of Words in each message:  288
Number of Words in each message:  92
Number of Words in each message:  84
Number of Words in each message:  290
Number of Words in each message:  204
Number of Words in each message:  106
Number of Words in each message:  253
Number of Words in each message:  103
```

In [14]:
```
#NUMBER OF TOKENS IN EACH MESSAGE
for item in df['full_text']:
    print("Number of Words in each message: ", len(set(item)))
```

```
Number of Words in each message:  28
Number of Words in each message:  37
Number of Words in each message:  30
Number of Words in each message:  42
Number of Words in each message:  33
Number of Words in each message:  30
Number of Words in each message:  38
Number of Words in each message:  38
Number of Words in each message:  30
Number of Words in each message:  30
Number of Words in each message:  36
Number of Words in each message:  51
```

```
Number of Words in each message:  35
Number of Words in each message:  29
Number of Words in each message:  36
Number of Words in each message:  37
Number of Words in each message:  34
Number of Words in each message:  57
Number of Words in each message:  39
```

In [15]:
```
nltk.download('wordnet')
nltk.download('stopwords')
from nltk.stem import SnowballStemmer
snowball_stemmer = SnowballStemmer('english')
import ast


###Converting all to lowercase emma

twitter_lower=[item.lower() for item in df['full_text']]
import string

###removing punctuations
table = str.maketrans('','', string.punctuation)
stripped = [item.translate(table) for item in twitter_lower]


###removing digits
words = [word for word in stripped if word.isalpha()]
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Nachiket\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Nachiket\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

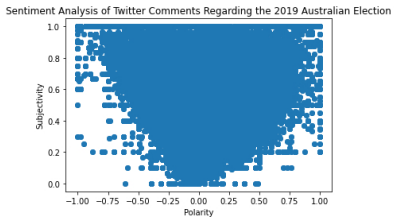
In [21]:
```
# ####removing stopwords
# from nltk.corpus import stopwords
# stop_words = set(stopwords.words('english'))
# ##removing unwanted words like author name, chapter and volume
# stop_words.update(["@"])
# words_ = [w for w in words if not w in stop_words]

###Word Lemmatization
wordnet_lemmatizer = WordNetLemmatizer()
lemmatized_word = [wordnet_lemmatizer.lemmatize(word) for word in df['full_text']]
print (lemmatized_word)
# ###Word Stemming
# stemmed_word = [snowball_stemmer.stem(word) for word in lemmatized_word]
# ####Tokenization
# tokens_tweets = [nltk.word_tokenize(sent) for sent in stemmed_word]
# print(tokens_tweets)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

In [22]:
```
#Sentiment Analysis
samples = df['full_text']#stemmed_word

x = []
y = []

polarity = 0
subject = 0



for item in samples:

    blob = TextBlob(item)
    current = blob.sentiment

    print(current)


    for value in current:
        polarity = blob.sentiment.polarity
        x.append(polarity)

        subject = blob.sentiment.subjectivity
        y.append(subject)
```

```
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=-0.7, subjectivity=0.2)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=0.35714285714285715, subjectivity=0.5)
Sentiment(polarity=0.0, subjectivity=0.3)
Sentiment(polarity=0.3025, subjectivity=0.51)
Sentiment(polarity=0.075, subjectivity=0.5)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=0.0, subjectivity=0.3333333333333333)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=-0.05, subjectivity=0.55)
Sentiment(polarity=-0.3, subjectivity=0.525)
Sentiment(polarity=0.15, subjectivity=0.55)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=0.25, subjectivity=0.6166666666666667)
Sentiment(polarity=0.16666666666666666, subjectivity=0.16666666666666666)
Sentiment(polarity=0.625, subjectivity=0.5)
Sentiment(polarity=0.0, subjectivity=0.0)
Sentiment(polarity=0.0, subjectivity=0.0)
```

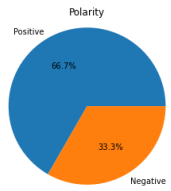
In [23]:
```
#Sentiment Scatterplot
import matplotlib.pyplot as plt
%matplotlib inline

plt.scatter(x,y)
plt.xlabel("Polarity")
plt.ylabel("Subjectivity")
plt.title("Sentiment Analysis of Twitter Comments Regarding the 2019 Australian Election")
plt.show()
```



In [24]:
```
#Sentiment (continued)
label_one = 'Positive', 'Negative'

size_one = 0
size_two = 0


for x_values in x:

    #Positive Polarity
    if (x_values > 0 and x_values <= 1):
```

```
        size_one += 1

    #Negative Polarity
    elif (x_values < 0 and x_values >= -1):
        size_two += 1



sizes_polarity = [size_one, size_two]

figure1, ax1 = plt.subplots()
ax1.pie(sizes_polarity, labels = label_one, autopct='%1.1f%%')

ax1.axis("equal")

plt.title("Polarity")
```

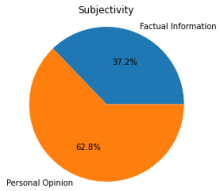
Out[24]: Text(0.5, 1.0, 'Polarity')



In [25]:
```
#positive, negative, and neutral sentiment
#!pip install vaderSentiment
import numpy as np
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyser = SentimentIntensityAnalyzer()
```

In [26]:
```
#Sentiment (continued)

label_two = 'Factual Information', 'Personal Opinion'

size_one = 0
size_two = 0

for y_values in y:

    #Tending Towards Personal Opinion
    if (y_values >= 0.5 and y_values <= 1):
        size_one += 1

    #Tending Towards Factual Information
    elif (y_values >= 0 and y_values <= 0.5):
        size_two += 1



sizes_subject = [size_one, size_two]

figure1, ax1 = plt.subplots()
ax1.pie(sizes_subject, labels = label_two, autopct='%1.1f%%')

ax1.axis("equal")

plt.title("Subjectivity")
```

Out[26]: Text(0.5, 1.0, 'Subjectivity')



In [36]:
```
###Wordcloud
from wordcloud import WordCloud, STOPWORDS,ImageColorGenerator
import matplotlib.pyplot as plt
import pandas as pd
from PIL import Image
import numpy as np
from io import BytesIO
import requests
comment_=''

for val in df['full_text'][0:1000]:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()
    tokens_=[w.replace("'", "") for w in tokens]

    # Converts each token into lowercase
    for i in range(len(tokens_)):
        comment_ += " ".join(tokens_)+" "



response = requests.get("https://live.staticflickr.com/5062/5613967601_62d4b0573a_b_d.jpg")
custom_mask = np.array(Image.open(BytesIO(response.content)))
wc = WordCloud(background_color="white", mask=custom_mask)
wc.generate(comment_)
plt.figure(figsize = (10, 10), facecolor = None)
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
print("############################Wordcloud for The First 1000 2019 Australian Election Tweets################################")
plt.show()
```

```
############################Wordcloud for The First 1000 2019 Australian Election Tweets################################
```


win https
indian election Politics ABC contest Labor conservative incumbents happened https

In [ ]: