Twitter sentiment analysis using Python

In this case study, it is required to build a sentiment analysis tool for the English language

Dataset - https://www.kaggle.com/kazanova/sentiment140

Content of the data set

1. **target: the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)**
2. **ids: The id of the tweet (2087)**
3. **date: the date of the tweet (Sat May 16 23:58:44 UTC 2009)**
4. **flag: The query (lyx). If there is no query, then this value is NO_QUERY.**
5. **user: the user that tweeted (robotickilldozr)**
6. **text: the text of the tweet (Lyx is cool)**

Step 1: Import the nescessary libraries for the sentiment analysis

In [2]:

```
#Libraries needed

#data manupilation libraries
import pandas as pd
import numpy as np
import re
import string

#text processing libraries and feature extraction using sklearn
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer

#machine learning libraries(sklearn)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
```

Step 2: Load the dataset and perform EDA on it

In [34]:

```
def load_dataset(filename, cols):
    df = pd.read_csv(filename, encoding='latin-1')
    df.columns = cols
    return df

#load dataset
dataset = load_dataset("dataset.csv", ['target', 't_id', 'created_at', 'query', 'user',
'text'])
```

In [35]:

```
dataset.head()
```

Out[35]:

	target	t_id	created_at	query	user	text
0	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
1	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
2	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
3	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....
4	0	1467811372	Mon Apr 06 22:20:00 PDT 2009	NO_QUERY	joy_wolf	@Kwesidei not the whole crew

In [36]:

```
dataset.tail()
```

Out[36]:

	target	t_id	created_at	query	user	text
1599994	4	2193601966	Tue Jun 16 08:40:49 PDT 2009	NO_QUERY	AmandaMarie1028	Just woke up. Having no school is the best fee...
1599995	4	2193601969	Tue Jun 16 08:40:49 PDT 2009	NO_QUERY	TheWDBoards	TheWDB.com - Very cool to hear old Walt interv...
1599996	4	2193601991	Tue Jun 16 08:40:49 PDT 2009	NO_QUERY	bpbabe	Are you ready for your MoJo Makeover? Ask me f...
1599997	4	2193602064	Tue Jun 16 08:40:49 PDT 2009	NO_QUERY	tinydiamondz	Happy 38th Birthday to my boo of alll time!!! ...
1599998	4	2193602129	Tue Jun 16 08:40:50 PDT 2009	NO_QUERY	RyanTrevMorris	happy #charitytuesday @theNSPCC @SparksCharity...

In [37]:

```
#names of the coloumns
dataset.columns

#Length of the dataset
print("length of the datase is", len(dataset))

#shape of the dataset
print("shape of the dataset is:", dataset.shape )

#information about the dataset
print("The information of the dataset is")
dataset.info()
```

```
length of the datase is 1599999
shape of the dataset is: (1599999, 6)
The information of the dataset is
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599999 entries, 0 to 1599998
Data columns (total 6 columns):
 #   Column      Non-Null Count    Dtype 
---  ------      --------------    ----- 
 0   target      1599999 non-null  int64 
 1   t_id        1599999 non-null  int64 
 2   created_at  1599999 non-null  object
 3   query       1599999 non-null  object
 4   user        1599999 non-null  object
 5   text        1599999 non-null  object
dtypes: int64(2), object(4)
memory usage: 73.2+ MB
```

Step 3: Getting rid of unwanted columns

In [38]:

```
def remove_unwanted_cols(df, cols):
    for col in cols:
        del df[col]
    return df

# Remove unwanted columns from dataset
dataset = remove_unwanted_cols(dataset, ['t_id', 'created_at', 'query', 'user'])
```

In [39]:

```
dataset.head()
```

Out[39]:

	target	text
0	**0**	**is upset that he can't update his Facebook by ...**
1	**0**	**@Kenichan I dived many times for the ball. Man...**
2	**0**	**my whole body feels itchy and like its on fire**
3	**0**	**@nationwideclass no, it's not behaving at all....**
4	**0**	**@Kwesidei not the whole crew**

In [40]:

```
dataset.tail()
```

Out[40]:

	target	text
1599994	**4**	**Just woke up. Having no school is the best fee...**
1599995	**4**	**TheWDB.com - Very cool to hear old Walt interv...**
1599996	**4**	**Are you ready for your MoJo Makeover? Ask me f...**
1599997	**4**	**Happy 38th Birthday to my boo of alll time!!! ...**
1599998	**4**	**happy #charitytuesday @theNSPCC @SparksCharity...**

In [41]:

```
#names of the coloumns
dataset.columns

#Length of the dataset
print("length of the datase is", len(dataset))

#shape of the dataset
print("shape of the dataset is:", dataset.shape )

#information about the dataset
print("The information of the dataset is")
dataset.info()
```

```
length of the datase is 1599999
shape of the dataset is: (1599999, 2)
The information of the dataset is
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599999 entries, 0 to 1599998
Data columns (total 2 columns):
 #   Column  Non-Null Count    Dtype
---  ------  --------------    -----
 0   target  1599999 non-null  int64
```

```
 1   text    1599999 non-null  object
dtypes: int64(1), object(1)
memory usage: 24.4+ MB
```

Step 4: Import the english stopwords

In [13]:

```
#download the english stop words
nltk.download("stopwords")
spw = set(stopwords.words('english'))
spw
```

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

Out[13]:

```
{'a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
 'been',
 'before',
 'being',
 'below',
 'between',
 'both',
 'but',
 'by',
 'can',
 'couldn',
 "couldn't",
 'd',
 'did',
 'didn',
 "didn't",
 'do',
 'does',
 'doesn',
 "doesn't",
 'doing',
 'don',
 "don't",
 'down',
 'during',
 'each',
 'few',
 'for',
 'from',
 'further',
 'had',
 'hadn',
 "hadn't",
 'has',
```

```
'hasn',
"hasn't",
'have',
'haven',
"haven't",
'having',
'he',
'her',
'here',
'hers',
'herself',
'him',
'himself',
'his',
'how',
'i',
'if',
'in',
'into',
'is',
'isn',
"isn't",
'it',
"it's",
'its',
'itself',
'just',
'll',
'm',
'ma',
'me',
'mightn',
"mightn't",
'more',
'most',
'mustn',
"mustn't",
'my',
'myself',
'needn',
"needn't",
'no',
'nor',
'not',
'now',
'o',
'of',
'off',
'on',
'once',
'only',
'or',
'other',
'our',
'ours',
'ourselves',
'out',
'over',
'own',
're',
's',
'same',
'shan',
"shan't",
'she',
"she's",
'should',
"should've",
'shouldn',
"shouldn't",
'so',
'some',
```

```
 'such',
 't',
 'than',
 'that',
 "that'll",
 'the',
 'their',
 'theirs',
 'them',
 'themselves',
 'then',
 'there',
 'these',
 'they',
 'this',
 'those',
 'through',
 'to',
 'too',
 'under',
 'until',
 'up',
 've',
 'very',
 'was',
 'wasn',
 "wasn't",
 'we',
 'were',
 'weren',
 "weren't",
 'what',
 'when',
 'where',
 'which',
 'while',
 'who',
 'whom',
 'why',
 'will',
 'with',
 'won',
 "won't",
 'wouldn',
 "wouldn't",
 'y',
 'you',
 "you'd",
 "you'll",
 "you're",
 "you've",
 'your',
 'yours',
 'yourself',
 'yourselves'}
```

Step 5: Preprocess the the data

In [42]:

```
#download the punkt to tokanize the words
import nltk
nltk.download('punkt')
```

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

Out[42]:

```
True
```

In [43]:

```
def preprocess_tweet_text(tweet):

    # covert all the text in lowercase
    tweet = tweet.lower()

    # Remove urls
    tweet = re.sub(r"http\S+|www\S+|https\S+", '', tweet, flags=re.MULTILINE)

    # Remove user @ references and '#' from tweet
    tweet = re.sub(r'\@\w+|\#','', tweet)

    # Remove punctuations
    tweet = tweet.translate(str.maketrans('', '', string.punctuation))

    # Remove stopwords
    tweet_tokens = word_tokenize(tweet)
    filtered_words = [word for word in tweet_tokens if word not in spw]

    return " ".join(filtered_words)
```

Step 6: Vectorize the Data

In [44]:

```
def get_feature_vector(train_fit):
    vector = TfidfVectorizer(sublinear_tf=True)
    vector.fit(train_fit)
    return vector
```

Step 7: Convert interger results to String

In [18]:

```
def int_to_string(sentiment):
    if sentiment == 0:
        return "Negative"
    elif sentiment == 2:
        return "Neutral"
    else:
        return "Positive"
```

Step 8: Output of the processed dataset

In [47]:

```
#Preprocess data
dataset.text = dataset['text'].apply(preprocess_tweet_text)
dataset.text.head()
```

Out[47]:

```
0    upset cant update facebook texting might cry r...
1    dived many times ball managed save 50 rest go ...
2                     whole body feels itchy like fire
3                             behaving im mad cant see
4                                          whole crew
Name: text, dtype: object
```

Step 9: Split the dataset into train and test

In [48]:

```
# Split dataset into Train, Test
```

```
# Same tf vector will be used for Testing sentiments on unseen trending data
tf_vector = get_feature_vector(np.array(dataset.iloc[:, 1]).ravel())
X = tf_vector.transform(np.array(dataset.iloc[:, 1]).ravel())
y = np.array(dataset.iloc[:, 0]).ravel()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=30
)
```

Step 10] a): Train the dataset with Naive Bayes Model and find it's accuracy score

In [49]:

```
# Training Naive Bayes model
NB_model = MultinomialNB()
NB_model.fit(X_train, y_train)
y_predict_nb = NB_model.predict(X_test)
print(accuracy_score(y_test, y_predict_nb))
```

```
0.7664125
```

Step 10] b): Train the dataset with Logistic Regression model and find it's accuracy score

In [50]:

```
# Training Logistics Regression model
LR_model = LogisticRegression(solver='lbfgs')
LR_model.fit(X_train, y_train)
y_predict_lr = LR_model.predict(X_test)
print(accuracy_score(y_test, y_predict_lr))
```

```
0.784925
```

```
C:\Users\rymbhavsar\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:762: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```