Muhammad Hammad Latif FA18-BCS-134

Rana Muhammad Sobaan FA18-BCS-038

Importing Libraries

In [1]:
```
import pandas as pd
import re
import numpy as np
import nltk

from nltk.tokenize import TweetTokenizer
from nltk import PorterStemmer
from nltk import ngrams

from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

Reading Training Dataset

We are using _ as a seperator/ delimeter

In [2]:
```
trainData = pd.read_csv("TrainData.csv","_")

trainData
```

Out[2]:

	Index	Comment	Polarity
0	0	time to eat with my best buddy! #lunch	Happy
1	1	@user @user if they want reflection money. #ksleg	Happy
2	2	---Good job but I' will expect a lot more in f...	Happy
3	3	totally dissatisfied with the service###%%@@ n...	Sad
4	4	loved my work!!!!!	Happy
5	5	Worst customer care service.....@@$$$angry	Sad
6	6	Brilliant effort guys!!!	Happy
7	7	@user @user you point one finger @user million...	Sad
8	8	words r free, it's how u use that can cost you...	Happy
9	9	you might be a libtard if... #libtard #sjw #li...	Sad

Reading Test Dataset

In [3]:
```
testData = pd.read_csv("TestData.csv","_")

testData
```

Out[3]:

	Index	Comment	Polarity
0	0	@use the pic says otherwise for young girls co...	Sad
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	Happy
2	2	@user when you're blocked by a troll because y...	Sad
3	3	dinner with sister!!	Happy
4	4	who else is planning on watching @user tomorrow?	happy

PreProcessing Begins

In [4]:
```
#Function to remove @user in the data

def remove_pattern(text,pattern):

    # re.findall() finds the pattern i.e @user and creating list
    r = re.findall(pattern,text)

    # re.sub() removes @user from the sentences in the dataset
    for i in r:
        text = re.sub(i,"",text)

    return text
```

In [5]:
```
#Applying function to both datasets for removing @user

trainData['Processed_Comment'] = np.vectorize(remove_pattern)(trainData['Comm

testData['Processed_Comment'] = np.vectorize(remove_pattern)(testData['Commen
```

In [6]:
```
# Removing everything except text i.e letters/words

trainData['Processed_Comment'] = trainData['Processed_Comment'].str.replace("

trainData
```

Out[6]:

	Index	Comment	Polarity	Processed_Comment
0	0	time to eat with my best buddy! #lunch	Happy	time to eat with my best buddy lunch
1	1	@user @user if they want reflection money. #ksleg	Happy	if they want reflection money ksleg
2	2	---Good job but I' will expect a lot more in f...	Happy	Good job but I will expect a lot more in f...
3	3	totally dissatisfied with the service###%%@@ n...	Sad	totally dissatisfied with the service nev...
4	4	loved my work!!!!!	Happy	loved my work
5	5	Worst customer care service.....@@$$$angry	Sad	Worst customer care service angry
6	6	Brilliant effort guys!!!	Happy	Brilliant effort guys
7	7	@user @user you point one finger @user million...	Sad	you point one finger millions are pointed r...
8	8	words r free, it's how u use that can cost you...	Happy	words r free it s how u use that can cost you...
9	9	you might be a libtard if... #libtard #sjw #li...	Sad	you might be a libtard if libtard sjw li...

In [7]:
```
# Removing everything except text i.e letters/words

testData['Processed_Comment'] = testData['Processed_Comment'].str.replace("[^

testData
```

Out[7]:

	Index	Comment	Polarity	Processed_Comment
0	0	@use the pic says otherwise for young girls co...	Sad	the pic says otherwise for young girls confin...
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	Happy	good night faith ever vaitacacommafiasdv
2	2	@user when you're blocked by a troll because y...	Sad	when you re blocked by a troll because you pr...
3	3	dinner with sister!!	Happy	dinner with sister
4	4	who else is planning on watching @user tomorrow?	happy	who else is planning on watching tomorrow

In [8]:
```
#Removing Short Words
trainData['Processed_Comment'] = trainData['Processed_Comment'].apply(lambda

trainData
```

Out[8]:

	Index	Comment	Polarity	Processed_Comment
0	0	time to eat with my best buddy! #lunch	Happy	time with best buddy lunch
1	1	@user @user if they want reflection money. #ksleg	Happy	they want reflection money ksleg
2	2	---Good job but I' will expect a lot more in f...	Happy	Good will expect more future
3	3	totally dissatisfied with the service###%%@@ n...	Sad	totally dissatisfied with service never used t...
4	4	loved my work!!!!!	Happy	loved work
5	5	Worst customer care service.....@@$$$angry	Sad	Worst customer care service angry
6	6	Brilliant effort guys!!!	Happy	Brilliant effort guys
7	7	@user @user you point one finger @user million...	Sad	point finger millions pointed right back jewis...
8	8	words r free, it's how u use that can cost you...	Happy	words free that cost verbal abuse love adult teen
9	9	you might be a libtard if... #libtard #sjw #li...	Sad	might libtard libtard liberal politics

In [9]:
```
#Removing Short Words
testData['Processed_Comment'] = testData['Processed_Comment'].apply(lambda x:

testData
```

Out[9]:

	Index	Comment	Polarity	Processed_Comment
0	0	@use the pic says otherwise for young girls co...	Sad	says otherwise young girls confined that kitch...
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	Happy	good night faith ever vaitacacommafiasdv
2	2	@user when you're blocked by a troll because y...	Sad	when blocked troll because promise blacklivesm...
3	3	dinner with sister!!	Happy	dinner with sister
4	4	who else is planning on watching @user tomorrow?	happy	else planning watching tomorrow

```
Making sure that all Happy and Sad are same
```

In [10]:
```
trainData['Polarity'] = trainData['Polarity'].apply(lambda x: x.capitalize())

trainData
```

Out[10]:

	Index	Comment	Polarity	Processed_Comment
0	0	time to eat with my best buddy! #lunch	Happy	time with best buddy lunch
1	1	@user @user if they want reflection money. #ksleg	Happy	they want reflection money ksleg
2	2	---Good job but I' will expect a lot more in f...	Happy	Good will expect more future
3	3	totally dissatisfied with the service###%%@@ n...	Sad	totally dissatisfied with service never used t...
4	4	loved my work!!!!!	Happy	loved work
5	5	Worst customer care service.....@@$$$angry	Sad	Worst customer care service angry
6	6	Brilliant effort guys!!!	Happy	Brilliant effort guys
7	7	@user @user you point one finger @user million...	Sad	point finger millions pointed right back jewis...
8	8	words r free, it's how u use that can cost you...	Happy	words free that cost verbal abuse love adult teen
9	9	you might be a libtard if... #libtard #sjw #li...	Sad	might libtard libtard liberal politics

In [11]:
```
testData['Polarity'] = testData['Polarity'].apply(lambda x: x.capitalize())

testData
```

Out[11]:

	Index	Comment	Polarity	Processed_Comment
0	0	@use the pic says otherwise for young girls co...	Sad	says otherwise young girls confined that kitch...
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	Happy	good night faith ever vaitacacommafiasdv
2	2	@user when you're blocked by a troll because y...	Sad	when blocked troll because promise blacklivesm...
3	3	dinner with sister!!	Happy	dinner with sister
4	4	who else is planning on watching @user tomorrow?	Happy	else planning watching tomorrow

Label Encoding for Train/Test Data

In [12]:
```
def labelEncoder(polarity):
    if(polarity == 'Happy'):
        return 1
    return 0
```

In [13]:
```
trainData['Polarity'] = trainData['Polarity'].apply(lambda x: labelEncoder(x))

trainData
```

Out[13]:

	Index	Comment	Polarity	Processed_Comment
0	0	time to eat with my best buddy! #lunch	1	time with best buddy lunch
1	1	@user @user if they want reflection money. #ksleg	1	they want reflection money ksleg
2	2	---Good job but I' will expect a lot more in f...	1	Good will expect more future
3	3	totally dissatisfied with the service###%%@@ n...	0	totally dissatisfied with service never used t...
4	4	loved my work!!!!!	1	loved work
5	5	Worst customer care service.....@@$$$angry	0	Worst customer care service angry
6	6	Brilliant effort guys!!!	1	Brilliant effort guys
7	7	@user @user you point one finger @user million...	0	point finger millions pointed right back jewis...
8	8	words r free, it's how u use that can cost you...	1	words free that cost verbal abuse love adult teen
9	9	you might be a libtard if... #libtard #sjw #li...	0	might libtard libtard liberal politics

In [14]:
```
testData['Polarity'] = testData['Polarity'].apply(lambda x: labelEncoder(x))

testData
```

Out[14]:

	Index	Comment	Polarity	Processed_Comment
0	0	@use the pic says otherwise for young girls co...	0	says otherwise young girls confined that kitch...
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	1	good night faith ever vaitacacommafiasdv
2	2	@user when you're blocked by a troll because y...	0	when blocked troll because promise blacklivesm...
3	3	dinner with sister!!	1	dinner with sister
4	4	who else is planning on watching @user tomorrow?	1	else planning watching tomorrow

Tokanizing Comments of train data

In [15]:
```
tokenized_trainComment = trainData['Processed_Comment'].apply(lambda x: x.spl

tokenized_trainComment
```

Out[15]:
```
0                     [time, with, best, buddy, lunch]
1                [they, want, reflection, money, ksleg]
2                    [Good, will, expect, more, future]
3    [totally, dissatisfied, with, service, never, ...
4                                           [loved, work]
5               [Worst, customer, care, service, angry]
6                               [Brilliant, effort, guys]
7    [point, finger, millions, pointed, right, back...
8    [words, free, that, cost, verbal, abuse, love,...
9         [might, libtard, libtard, liberal, politics]
Name: Processed_Comment, dtype: object
```

POS Tagging

In [16]:

```
#nltk.download('punkt')
#nltk.download('averaged_perceptron_tagger')
trainDataList = trainData['Processed_Comment'].tolist()
taggedList = list()
posList = list()

for sentence in trainDataList:
        tokenized = nltk.word_tokenize(sentence)
        taggedList.append(nltk.pos_tag(tokenized))

#removing repititions
for tList in taggedList:
    for word_tuple in tList:
        if word_tuple not in posList:
            posList.append(word_tuple)

posList
```

Out[16]:

```
[('time', 'NN'),
 ('with', 'IN'),
 ('best', 'JJS'),
 ('buddy', 'NN'),
 ('lunch', 'NN'),
 ('they', 'PRP'),
 ('want', 'VBP'),
 ('reflection', 'NN'),
 ('money', 'NN'),
 ('ksleg', 'NN'),
 ('Good', 'NNP'),
 ('will', 'MD'),
 ('expect', 'VB'),
 ('more', 'JJR'),
 ('future', 'JJ'),
 ('totally', 'RB'),
 ('dissatisfied', 'JJ'),
 ('service', 'NN'),
 ('never', 'RB'),
 ('used', 'VBD'),
 ('this', 'DT'),
 ('again', 'RB'),
 ('loved', 'VBN'),
 ('work', 'NN'),
 ('Worst', 'NNP'),
 ('customer', 'NN'),
 ('care', 'NN'),
 ('angry', 'JJ'),
 ('Brilliant', 'JJ'),
 ('effort', 'NN'),
 ('guys', 'NNS'),
 ('point', 'NN'),
 ('finger', 'NN'),
 ('millions', 'NNS'),
 ('pointed', 'VBD'),
 ('right', 'JJ'),
 ('back', 'RB'),
 ('jewishsupremacist', 'NN'),
 ('words', 'NNS'),
```

```
('free', 'VBP'),
('that', 'IN'),
('cost', 'NN'),
('verbal', 'JJ'),
('abuse', 'NN'),
('love', 'NN'),
('adult', 'NN'),
('teen', 'NN'),
('might', 'MD'),
('libtard', 'RB'),
('libtard', 'VB'),
('liberal', 'JJ'),
('politics', 'NNS')]
```

Removing additional letters such as ed, 's etc.

In [17]:
```
ps = PorterStemmer()

tokenized_trainComment = tokenized_trainComment.apply(lambda x: [ps.stem(i) f

tokenized_trainComment
```

Out[17]:
```
0                    [time, with, best, buddi, lunch]
1                 [they, want, reflect, money, ksleg]
2                   [good, will, expect, more, futur]
3    [total, dissatisfi, with, servic, never, use, ...
4                                          [love, work]
5                [worst, custom, care, servic, angri]
6                             [brilliant, effort, guy]
7    [point, finger, million, point, right, back, j...
8    [word, free, that, cost, verbal, abus, love, a...
9              [might, libtard, libtard, liber, polit]
Name: Processed_Comment, dtype: object
```

Replacing old Processed comments

In [18]:
```
for i in range(len(tokenized_trainComment)):
    tokenized_trainComment[i] = ' '.join(tokenized_trainComment[i])

trainData['Processed_Comment'] = tokenized_trainComment

trainData
```

Out[18]:

	Index	Comment	Polarity	Processed_Comment
0	0	time to eat with my best buddy! #lunch	1	time with best buddi lunch
1	1	@user @user if they want reflection money. #ksleg	1	they want reflect money ksleg
2	2	---Good job but I' will expect a lot more in f...	1	good will expect more futur
3	3	totally dissatisfied with the service###%%@@ n...	0	total dissatisfi with servic never use thi ser...
4	4	loved my work!!!!!	1	love work
5	5	Worst customer care service.....@@$$$angry	0	worst custom care servic angri
6	6	Brilliant effort guys!!!	1	brilliant effort guy
7	7	@user @user you point one finger @user million...	0	point finger million point right back jewishsu...
8	8	words r free, it's how u use that can cost you...	1	word free that cost verbal abus love adult teen
9	9	you might be a libtard if... #libtard #sjw #li...	0	might libtard libtard liber polit

Tokanizing Comments of Test Data

In [19]:
```
tokenized_testComment = testData["Processed_Comment"].apply(lambda x: x.split

tokenized_testComment
```

Out[19]:
```
0    [says, otherwise, young, girls, confined, that...
1       [good, night, faith, ever, vaitacacommafiasdv]
2    [when, blocked, troll, because, promise, black...
3                                [dinner, with, sister]
4                  [else, planning, watching, tomorrow]
Name: Processed_Comment, dtype: object
```

POS Tagging

In [20]:
```
#nltk.download('punkt')
#nltk.download('averaged_perceptron_tagger')
testDataList = testData['Processed_Comment'].tolist()
taggedList = list()
posList = list()

for sentence in testDataList:
        tokenized = nltk.word_tokenize(sentence)
        taggedList.append(nltk.pos_tag(tokenized))

#removing repititions
for tList in taggedList:
    for word_tuple in tList:
        if word_tuple not in posList:
            posList.append(word_tuple)

posList
```

Out[20]:
```
[('says', 'VBZ'),
 ('otherwise', 'RB'),
 ('young', 'JJ'),
 ('girls', 'NNS'),
 ('confined', 'VBD'),
 ('that', 'IN'),
 ('kitchen', 'NN'),
 ('void', 'NN'),
 ('meaning', 'VBG'),
 ('beyond', 'IN'),
 ('cheap', 'JJ'),
 ('publicity', 'NN'),
 ('topoli', 'NN'),
 ('good', 'JJ'),
 ('night', 'NN'),
 ('faith', 'NN'),
 ('ever', 'RB'),
 ('vaitacacommafiasdv', 'VBD'),
 ('when', 'WRB'),
 ('blocked', 'VBN'),
 ('troll', 'NN'),
 ('because', 'IN'),
 ('promise', 'NN'),
 ('blacklivesmatter', 'NN'),
 ('nonsensical', 'JJ'),
 ('rants', 'NNS'),
 ('dinner', 'NN'),
 ('with', 'IN'),
 ('sister', 'NN'),
 ('else', 'RB'),
 ('planning', 'VBG'),
 ('watching', 'VBG'),
 ('tomorrow', 'NN')]
```

Removing additional letters such as ed, 's etc.

In [21]:
```
ps = PorterStemmer()

tokenized_testComment = tokenized_testComment.apply(lambda x : [ps.stem(i) fo


tokenized_testComment
```

Out[21]:
```
0    [say, otherwis, young, girl, confin, that, kit...
1       [good, night, faith, ever, vaitacacommafiasdv]
2    [when, block, troll, becaus, promis, blacklive...
3                               [dinner, with, sister]
4                         [els, plan, watch, tomorrow]
Name: Processed_Comment, dtype: object
```

Replacing old Processed comments

In [22]:
```
for i in range(len(tokenized_testComment)):
    tokenized_testComment[i] = ' '.join(tokenized_testComment[i])

testData['Processed_Comment'] = tokenized_testComment

testData
```

Out[22]:

	Index	Comment	Polarity	Processed_Comment
0	0	@use the pic says otherwise for young girls co...	0	say otherwis young girl confin that kitchen vo...
1	1	#good night! ?? #faith ever #vaitacacommafiasdv	1	good night faith ever vaitacacommafiasdv
2	2	@user when you're blocked by a troll because y...	0	when block troll becaus promis blacklivesmatt ...
3	3	dinner with sister!!	1	dinner with sister
4	4	who else is planning on watching @user tomorrow?	1	els plan watch tomorrow

Feature Extraction from Train Data

Bag of Words

In [23]:

```
cv= CountVectorizer()

bag_of_words_train = cv.fit_transform(trainData['Processed_Comment']).toarray

print(cv.vocabulary_)

print(cv.get_feature_names())
print('\n')
print(bag_of_words_train)
```

```
{'time': 39, 'with': 45, 'best': 5, 'buddi': 7, 'lunch': 24, 'they': 37, 'w
ant': 43, 'reflect': 32, 'money': 27, 'ksleg': 20, 'good': 17, 'will': 44,
'expect': 13, 'more': 28, 'futur': 16, 'total': 40, 'dissatisfi': 11, 'serv
ic': 34, 'never': 29, 'use': 41, 'thi': 38, 'again': 2, 'love': 23, 'work':
47, 'worst': 48, 'custom': 10, 'care': 8, 'angri': 3, 'brilliant': 6, 'effo
rt': 12, 'guy': 18, 'point': 30, 'finger': 14, 'million': 26, 'right': 33,
'back': 4, 'jewishsupremacist': 19, 'word': 46, 'free': 15, 'that': 36, 'co
st': 9, 'verbal': 42, 'abus': 0, 'adult': 1, 'teen': 35, 'might': 25, 'libt
ard': 22, 'liber': 21, 'polit': 31}
['abus', 'adult', 'again', 'angri', 'back', 'best', 'brilliant', 'buddi',
'care', 'cost', 'custom', 'dissatisfi', 'effort', 'expect', 'finger', 'fre
e', 'futur', 'good', 'guy', 'jewishsupremacist', 'ksleg', 'liber', 'libtar
d', 'love', 'lunch', 'might', 'million', 'money', 'more', 'never', 'point',
'polit', 'reflect', 'right', 'servic', 'teen', 'that', 'they', 'thi', 'tim
e', 'total', 'use', 'verbal', 'want', 'will', 'with', 'word', 'work', 'wors
t']


[[0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 1 0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0
  0 1 0 0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 1 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 2 0 0
  0 0 1 0 1 1 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 1 0]
 [0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
  0 0 0 0 0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 2 0 0 1 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0]
 [1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1
  1 0 0 0 0 0 1 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 1 0 0 0 0 0 1 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0]]
```

TF-IDF

In [24]:

```
tfidf = TfidfVectorizer()

tfidf_matrix = tfidf.fit_transform(trainData['Processed_Comment'])

trainData_tfidf = pd.DataFrame(tfidf_matrix.todense())

display(trainData_tfidf)
```

	0	1	2	3	4	5	6	7	8	
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.460158	0.00000	0.460158	0.000000	0
1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0
3	0.000000	0.000000	0.322526	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0
5	0.000000	0.000000	0.000000	0.460158	0.000000	0.000000	0.00000	0.000000	0.460158	0
6	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.57735	0.000000	0.000000	0
7	0.000000	0.000000	0.000000	0.000000	0.333333	0.000000	0.00000	0.000000	0.000000	0
8	0.338591	0.338591	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0
9	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0

10 rows × 49 columns

N-Gram

In [25]:
```
trainDataList = trainData['Processed_Comment'].tolist()
grams = list()
n = 3

for sentence in trainDataList:
    threeGrams = ngrams(sentence.split(), n)
    for gram in threeGrams:
      grams.append(gram)

grams
```

Out[25]:
```
[('time', 'with', 'best'),
 ('with', 'best', 'buddi'),
 ('best', 'buddi', 'lunch'),
 ('they', 'want', 'reflect'),
 ('want', 'reflect', 'money'),
 ('reflect', 'money', 'ksleg'),
 ('good', 'will', 'expect'),
 ('will', 'expect', 'more'),
 ('expect', 'more', 'futur'),
 ('total', 'dissatisfi', 'with'),
 ('dissatisfi', 'with', 'servic'),
 ('with', 'servic', 'never'),
 ('servic', 'never', 'use'),
 ('never', 'use', 'thi'),
 ('use', 'thi', 'servic'),
 ('thi', 'servic', 'again'),
 ('worst', 'custom', 'care'),
 ('custom', 'care', 'servic'),
 ('care', 'servic', 'angri'),
 ('brilliant', 'effort', 'guy'),
 ('point', 'finger', 'million'),
 ('finger', 'million', 'point'),
 ('million', 'point', 'right'),
 ('point', 'right', 'back'),
 ('right', 'back', 'jewishsupremacist'),
 ('word', 'free', 'that'),
 ('free', 'that', 'cost'),
 ('that', 'cost', 'verbal'),
 ('cost', 'verbal', 'abus'),
 ('verbal', 'abus', 'love'),
 ('abus', 'love', 'adult'),
 ('love', 'adult', 'teen'),
 ('might', 'libtard', 'libtard'),
 ('libtard', 'libtard', 'liber'),
 ('libtard', 'liber', 'polit')]
```

Feature Extraction for Test Data

Bag of Words

In [26]:
```
cv= CountVectorizer()

bag_of_words_test = cv.fit_transform(testData['Processed_Comment']).toarray()

print(cv.vocabulary_)

print(cv.get_feature_names())
print('\n')
print(bag_of_words_test)
```

```
{'say': 21, 'otherwis': 16, 'young': 32, 'girl': 10, 'confin': 5, 'that': 2
3, 'kitchen': 12, 'void': 28, 'mean': 13, 'beyond': 1, 'cheap': 4, 'publi
c': 19, 'topoli': 25, 'good': 11, 'night': 14, 'faith': 9, 'ever': 8, 'vait
acacommafiasdv': 27, 'when': 30, 'block': 3, 'troll': 26, 'becaus': 0, 'pro
mis': 18, 'blacklivesmatt': 2, 'nonsens': 15, 'rant': 20, 'dinner': 6, 'wit
h': 31, 'sister': 22, 'els': 7, 'plan': 17, 'watch': 29, 'tomorrow': 24}
['becaus', 'beyond', 'blacklivesmatt', 'block', 'cheap', 'confin', 'dinne
r', 'els', 'ever', 'faith', 'girl', 'good', 'kitchen', 'mean', 'night', 'no
nsens', 'otherwis', 'plan', 'promis', 'public', 'rant', 'say', 'sister', 't
hat', 'tomorrow', 'topoli', 'troll', 'vaitacacommafiasdv', 'void', 'watch',
'when', 'with', 'young']


[[0 1 0 0 1 1 0 0 0 0 1 0 1 1 0 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 0 0 1]
 [0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
 [1 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 1 0 0 0 1 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0]]
```

TF-IDF

In [27]:
```
tfidf = TfidfVectorizer()

tfidf_matrix = tfidf.fit_transform(testData['Processed_Comment'])

testData_tfidf = pd.DataFrame(tfidf_matrix.todense())

display(testData_tfidf)
```

	0	1	2	3	4	5	6	7	8	9
0	0.000000	0.27735	0.000000	0.000000	0.27735	0.27735	0.00000	0.0	0.000000	0.000000
1	0.000000	0.00000	0.000000	0.000000	0.00000	0.00000	0.00000	0.0	0.447214	0.447214
2	0.353553	0.00000	0.353553	0.353553	0.00000	0.00000	0.00000	0.0	0.000000	0.000000
3	0.000000	0.00000	0.000000	0.000000	0.00000	0.00000	0.57735	0.0	0.000000	0.000000
4	0.000000	0.00000	0.000000	0.000000	0.00000	0.00000	0.00000	0.5	0.000000	0.000000

5 rows × 33 columns

N-Gram

In [28]:

```
testDataList = testData['Processed_Comment'].tolist()
grams = list()
n = 3

for sentence in testDataList:
    threeGrams = ngrams(sentence.split(), n)
    for gram in threeGrams:
      grams.append(gram)

grams
```

Out[28]:

```
[('say', 'otherwis', 'young'),
 ('otherwis', 'young', 'girl'),
 ('young', 'girl', 'confin'),
 ('girl', 'confin', 'that'),
 ('confin', 'that', 'kitchen'),
 ('that', 'kitchen', 'void'),
 ('kitchen', 'void', 'mean'),
 ('void', 'mean', 'beyond'),
 ('mean', 'beyond', 'cheap'),
 ('beyond', 'cheap', 'public'),
 ('cheap', 'public', 'topoli'),
 ('good', 'night', 'faith'),
 ('night', 'faith', 'ever'),
 ('faith', 'ever', 'vaitacacommafiasdv'),
 ('when', 'block', 'troll'),
 ('block', 'troll', 'becaus'),
 ('troll', 'becaus', 'promis'),
 ('becaus', 'promis', 'blacklivesmatt'),
 ('promis', 'blacklivesmatt', 'nonsens'),
 ('blacklivesmatt', 'nonsens', 'rant'),
 ('dinner', 'with', 'sister'),
 ('els', 'plan', 'watch'),
 ('plan', 'watch', 'tomorrow')]
```