Homework4

April 18, 2019

In [50]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer,ENGLISH_STOP_WORDS
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score
from sklearn.pipeline import Pipeline
from scipy.sparse import coo_matrix, hstack
from sklearn.feature_extraction.text import TfidfVectorizer, TfidfTransformer
import re
from imblearn.under_sampling import RandomUnderSampler
from sklearn.ensemble import RandomForestClassifier
import xgboost
import os
os.environ['KMP_DUPLICATE_LIB_OK']='True'
from xgboost import XGBClassifier
from sklearn.feature_extraction.text import TfidfVectorizer, TfidfTransformer
import nltk
words = set(nltk.corpus.words.words())
from gensim.corpora import Dictionary
from gensim.models.word2vec import Word2Vec
import string
```

```
/anaconda3/lib/python3.7/site-packages/smart_open/ssh.py:34: UserWarning: paramiko missing, ope
  warnings.warn('paramiko missing, opening SSH/SCP/SFTP paths will be disabled.  `pip install p
```

In [48]:
```
# Load the datasets
df = pd.read_csv("./Data/reddit_200k_train.csv",encoding='iso-8859-1')
test = pd.read_csv("./Data/reddit_200k_test.csv",encoding='iso-8859-1')
```

In [3]:
```
# Strip unnecessary columns
df = df.drop([df.columns[i] for i in range(len(df.columns)) if i != 1 and i != 7],axis
test = test.drop([test.columns[i] for i in range(len(test.columns)) if i != 1 and i !=
```

In [4]:
```
# Separate the predictive feature and the targeted feature
X_list = df["body"].tolist()
X = df["body"]
y = df["REMOVED"]
X_n = df['body'].str.replace('\d+','')
```

1 Task 1 Bag of Words and simple Features

1.1 1.1: Baseline Model (Logistic Regression)

In [5]:
```
# Tokenization
vect = CountVectorizer()
cv_X = vect.fit_transform(X)

# Split the dataset into the training set and test set
X_train, X_test, y_train, y_test = train_test_split(cv_X,y, random_state = 0)
```

In [6]:
```
# Build a baseline model (logistic regression)
lg = LogisticRegression()

# Fit the model to the data
lg.fit(X_train, y_train)

# Compute the accurecy
lg.score(X_test,y_test)
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
/anaconda3/lib/python3.7/site-packages/sklearn/svm/base.py:931: ConvergenceWarning: Liblinear
  "the number of iterations.", ConvergenceWarning)
```

Out[6]:
```
0.6886087434042452
```

In [7]:
```
# Compute the test score
y_pred = lg.predict(X_test)
accuracy_score(y_test, y_pred)
```

Out[7]:
```
0.6886087434042452
```

In [8]:
```
# Build a confusion matrix
confusion_matrix(y_test, y_pred)
```

Out[8]:
```
array([[19777,  5859],
       [ 7183,  9064]])
```

In [9]:
```
# The table of each test score
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

       False       0.73      0.77      0.75     25636
        True       0.61      0.56      0.58     16247

   micro avg       0.69      0.69      0.69     41883
   macro avg       0.67      0.66      0.67     41883
weighted avg       0.68      0.69      0.69     41883
```

2 1.2: Rescale and Use Other Techniques (n-grams, tf-idf)

Strip punctuations and stopwords without n_gram

In [10]:
```
# Create a countvectorizer with arguments of removing punctuations and stopwords
vect = CountVectorizer(token_pattern=r"\b\w+\b", stop_words=ENGLISH_STOP_WORDS, min_d
X_n_cv = vect.fit_transform(X_n)
```

In [11]:
```
#Split the dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X_n_cv,y, random_state = 0)

# Build a logistic model and fit it to the data
lg = LogisticRegression()
lg.fit(X_train, y_train)

# Compute the test score
lg.score(X_test,y_test)
```

Out[11]: 0.6833082634959291

In [13]:
```
# Gridsearch
pipe = Pipeline([
    ("tf",TfidfTransformer()),
    ("lg", LogisticRegression())])



param_grid = {
    "lg__penalty":["l1"],
    "lg__C":np.logspace(-3,3,7)

}


grid = GridSearchCV(pipe, param_grid=param_grid, cv=3 , scoring="roc_auc", verbose=10


X_train, X_test, y_train, y_test = train_test_split(X_n_cv,y, random_state = 0)

grid.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 7 candidates, totalling 21 fits
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
```

```
[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.5s
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.6s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.5s
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    1.4s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.6s
[CV] lg__C=0.01, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    2.3s remaining:    0.0s


[CV]  lg__C=0.01, lg__penalty=l1, score=0.543320952797687, total=   0.7s
[CV] lg__C=0.01, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    3.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.545242171679164, total=   0.9s
[CV] lg__C=0.01, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    4.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.5436764386427085, total=   0.9s
[CV] lg__C=0.1, lg__penalty=l1 ...
```

```
[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    5.4s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7141468589305742, total=   0.9s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    6.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7214084651395443, total=   0.9s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   8 out of   8 | elapsed:    7.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7169638571093843, total=   0.8s
[CV] lg__C=1.0, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:    8.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7571239869393849, total=   1.1s
[CV] lg__C=1.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7588402902349063, total=   1.0s
[CV] lg__C=1.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7554309139610973, total=   1.5s
[CV] lg__C=10.0, lg__penalty=l1 ...
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7232123744950699, total=   2.4s
[CV] lg__C=10.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7216165450123482, total=   1.5s
[CV] lg__C=10.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7200405552795927, total=   3.1s
[CV] lg__C=100.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.68735210005142, total=   8.2s
[CV] lg__C=100.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6861621522619682, total=   7.2s
[CV] lg__C=100.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6841141845653332, total=   8.6s
[CV] lg__C=1000.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
```

```
[CV]  lg__C=1000.0, lg__penalty=l1, score=0.675826573453972, total=  21.7s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6748955641104528, total=  12.6s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6727147857190562, total=  17.1s


[Parallel(n_jobs=1)]: Done  21 out of  21 | elapsed:  1.6min finished
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


Out[13]: GridSearchCV(cv=3, error_score='raise-deprecating',
               estimator=Pipeline(memory=None,
              steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
                  intercept_scaling=1, max_iter=100, multi_class='warn',
                  n_jobs=None, penalty='l2', random_state=None, solver='warn',
                  tol=0.0001, verbose=0, warm_start=False))]),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'lg__penalty': ['l1'], 'lg__C': array([1.e-03, 1.e-02, 1.e-01, 1.e-
               pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
               scoring='roc_auc', verbose=10)
```

```
In [14]: # The best parameters and accuracy score
         final_model = grid.best_estimator_
         y_pred = grid.predict(X_test)
         print(grid.best_params_ )
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))
```

```
{'lg__C': 1.0, 'lg__penalty': 'l1'}
Accurancy: 0.6949836449155982
```

Strip punctuations and stopwords with bi_gram

```
In [15]: # Create a countVectorizer with arguments of removing punctuations and stopwords and
         vect = CountVectorizer(token_pattern=r"\b\w+\b", stop_words=ENGLISH_STOP_WORDS, min_d
         X_n_cv = vect.fit_transform(X_n)
```

In [16]:
```
#Split the dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X_n_cv,y, random_state = 0)

# Build a logistic model and fit it to the data
lg = LogisticRegression()
lg.fit(X_train, y_train)

# Compute the test score
print("Accuracy: {}".format(lg.score(X_test,y_test))
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/svm/base.py:931: ConvergenceWarning: Liblinear i
  "the number of iterations.", ConvergenceWarning)
```

Out[16]:
```
0.6847885777045579
```

In [17]:
```
# Gridsearch
pipe = Pipeline([
    ("tf",TfidfTransformer()),
    ("lg", LogisticRegression())])



param_grid = {
#     "cv__token_pattern":[r"\b\w+\b"],
#     "cv__min_df":np.arange(4,8,2),
#     "cv__stop_words":[ENGLISH_STOP_WORDS]
    "lg__penalty":["l1"],
    "lg__C":np.logspace(-3,3,7)

}
grid = GridSearchCV(pipe, param_grid=param_grid, cv=3 , scoring="roc_auc", verbose=10)


X_train, X_test, y_train, y_test = train_test_split(X_n_cv,y, random_state = 0)

grid.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 7 candidates, totalling 21 fits
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   1.0s
[CV] lg__C=0.001, lg__penalty=l1 ...
```

```
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    1.3s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   1.0s
[CV] lg__C=0.001, lg__penalty=l1 ...

[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    2.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   1.0s
[CV] lg__C=0.01, lg__penalty=l1 ...

[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    3.8s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.5200629884521171, total=   1.1s
[CV] lg__C=0.01, lg__penalty=l1 ...

[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    5.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.5198304689322654, total=   1.4s
[CV] lg__C=0.01, lg__penalty=l1 ...

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    6.8s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.5192083365736061, total=   1.5s
[CV] lg__C=0.1, lg__penalty=l1 ...

[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    8.6s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
```

```
[CV]  lg__C=0.1, lg__penalty=l1, score=0.7049510086449231, total=   1.5s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:   10.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.712579769644371, total=   2.3s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   8 out of   8 | elapsed:   13.0s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7086567501579834, total=   1.9s
[CV] lg__C=1.0, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:   15.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7571157638944396, total=   3.0s
[CV] lg__C=1.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7595311984040095, total=   1.9s
[CV] lg__C=1.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.756901679814624, total=   1.7s
[CV] lg__C=10.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
```

```
[CV]  lg__C=10.0, lg__penalty=l1, score=0.711344983553862, total=   3.9s
[CV] lg__C=10.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7115647019000113, total=   4.9s
[CV] lg__C=10.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7076470468629725, total=   6.1s
[CV] lg__C=100.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6762409862943093, total=  10.8s
[CV] lg__C=100.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6768835219292323, total=   9.0s
[CV] lg__C=100.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6703638537038803, total=  10.8s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6613833368361557, total=  17.1s
[CV] lg__C=1000.0, lg__penalty=l1 ...
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6618883625646825, total=  15.3s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.655401593091671, total=  15.3s


[Parallel(n_jobs=1)]: Done  21 out of  21 | elapsed:  2.0min finished
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


Out[17]: GridSearchCV(cv=3, error_score='raise-deprecating',
               estimator=Pipeline(memory=None,
              steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
                   intercept_scaling=1, max_iter=100, multi_class='warn',
                   n_jobs=None, penalty='l2', random_state=None, solver='warn',
                   tol=0.0001, verbose=0, warm_start=False))]),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'lg__penalty': ['l1'], 'lg__C': array([1.e-03, 1.e-02, 1.e-01, 1.e
               pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
               scoring='roc_auc', verbose=10)
```

```
In [18]: # The best parameters and accuracy score
         final_model = grid.best_estimator_
         y_pred = grid.predict(X_test)
         print(grid.best_params_ )
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))
```

```
{'lg__C': 1.0, 'lg__penalty': 'l1'}
Accurancy: 0.69593868634052
```

3 1.3: Adding New Features (URL and Numbers)

```
In [19]: # A function that checks if a string contains an url
         def findurl(string):
             url = re.findall('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+] |[!*\(\), ]|(?:%[0-9a-fA
             return url
```

```
        # A function that checks if a number(s) are  in a string
        def findnum(string):
            num = [str(i) for i in range(10) if str(i) in string]
            return num
```

In [22]:
```
        #Adding extra features
        df['url'] = -1
        df["num"] = -1
        for i in range(df.shape[0]):
            st = df.iloc[i,0]
            result = findurl(st)
            result1 = findnum(st)
            if result != []:
                df.iloc[i,-2] = 1
            else:
                df.iloc[i,-2] = 0

            if result1 != []:
                df.iloc[i,-1] = 1
            else:
                df.iloc[i,-1] = 0
```

In [23]:
```
        num = df["num"]
        url = df["url"]

        r = [[num[i],url[i]] for i in range(len(num))]
```

In [24]:
```
        vect = CountVectorizer(token_pattern=r"\b\w+\b", stop_words=ENGLISH_STOP_WORDS, min_d
        X_n_cv = vect.fit_transform(X_n)

        X_n_cv = hstack([X_n_cv,r])
```

In [25]:
```
        pipe = Pipeline([
            ("tf",TfidfTransformer()),
            ("lg", LogisticRegression())])



        param_grid = {
        #     "cv__token_pattern":[r"\b\w+\b"],
        #     "cv__min_df":np.arange(4,8,2),
        #     "cv__stop_words":[ENGLISH_STOP_WORDS]
            "lg__penalty":["l1"],
            "lg__C":np.logspace(-3,3,7)

        }
        grid = GridSearchCV(pipe, param_grid=param_grid, cv=3 , scoring="roc_auc", verbose=10
```

```
        X_train, X_test, y_train, y_test = train_test_split(X_n_cv,y, random_state = 42)

        grid.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 7 candidates, totalling 21 fits
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.4s
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.3s
[CV] lg__C=0.001, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    1.0s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV] ... lg__C=0.001, lg__penalty=l1, score=0.5, total=   0.4s
[CV] lg__C=0.01, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    1.4s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.5531012670985127, total=   0.4s
[CV] lg__C=0.01, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    1.9s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)
```

```
[CV]  lg__C=0.01, lg__penalty=l1, score=0.5520064177220975, total=   0.5s
[CV] lg__C=0.01, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    2.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.01, lg__penalty=l1, score=0.55245930848009, total=   0.5s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    3.1s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.718794809410702, total=   0.6s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    3.8s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7109608045513142, total=   0.6s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   8 out of   8 | elapsed:    4.6s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.7195023952145924, total=   0.6s
[CV] lg__C=1.0, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:    5.3s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7573855941397839, total=   1.6s
[CV] lg__C=1.0, lg__penalty=l1 ...
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7528435090333897, total=   1.5s
[CV] lg__C=1.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.7578018654843406, total=   1.2s
[CV] lg__C=10.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7226647315623086, total=   2.1s
[CV] lg__C=10.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7179227553408662, total=   2.6s
[CV] lg__C=10.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=10.0, lg__penalty=l1, score=0.7237819191192159, total=   2.5s
[CV] lg__C=100.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6835210004996994, total=   3.9s
[CV] lg__C=100.0, lg__penalty=l1 ...

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)
```

```
[CV]  lg__C=100.0, lg__penalty=l1, score=0.6830174899568582, total=   4.5s
[CV] lg__C=100.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=100.0, lg__penalty=l1, score=0.6840735473302715, total=   4.9s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6719418174848119, total=   7.2s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6719357116277758, total=   8.0s
[CV] lg__C=1000.0, lg__penalty=l1 ...


/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1000.0, lg__penalty=l1, score=0.6711785331687943, total=   5.8s


[Parallel(n_jobs=1)]: Done  21 out of  21 | elapsed:   52.9s finished
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


Out[25]: GridSearchCV(cv=3, error_score='raise-deprecating',
               estimator=Pipeline(memory=None,
              steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
                   intercept_scaling=1, max_iter=100, multi_class='warn',
                   n_jobs=None, penalty='l2', random_state=None, solver='warn',
                   tol=0.0001, verbose=0, warm_start=False))]),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'lg__penalty': ['l1'], 'lg__C': array([1.e-03, 1.e-02, 1.e-01, 1.e
               pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
               scoring='roc_auc', verbose=10)
```

```
In [26]: # The best parameters and accuracy score
         final_model = grid.best_estimator_
         y_pred = grid.predict(X_test)
         print(grid.best_params_ )
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))
```

```
{'lg__C': 1.0, 'lg__penalty': 'l1'}
Accurancy: 0.6952462813074517
```

3.0.1 Remove non-English words

```
In [30]: words = set(nltk.corpus.words.words())
         X_n_n = [] #string without numbers and non english words
         for sentence in X_n:
             sent = sentence


             a = " ".join(w for w in nltk.wordpunct_tokenize(sent) \
                     if w.lower() in words or not w.isalpha())
             X_n_n.append(a)
```

```
In [31]: vect = CountVectorizer(token_pattern=r"\b\w+\b", stop_words=ENGLISH_STOP_WORDS, min_d
         X_n_n_cv = vect.fit_transform(X_n_n)

         X_n_n_cv = hstack([X_n_n_cv,r])
```

```
In [32]: pipe = Pipeline([
             ("tf",TfidfTransformer()),
             ("lg", LogisticRegression())])



         param_grid = {
             "lg__penalty":["l1"],
         }
         grid = GridSearchCV(pipe, param_grid=param_grid, cv=3 , scoring="roc_auc", verbose =


         X_train, X_test, y_train, y_test = train_test_split(X_n_n_cv, y, stratify = y, random_

         grid.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 1 candidates, totalling 3 fits
[CV] lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
```

```
  FutureWarning)


[CV] ... lg__penalty=l1, score=0.7300862231110424, total=   0.5s
[CV] lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__penalty=l1, score=0.7271632904992076, total=   0.5s
[CV] lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    1.1s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__penalty=l1, score=0.7221026183479625, total=   0.5s


[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    1.7s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    1.7s finished
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


Out[32]: GridSearchCV(cv=3, error_score='raise-deprecating',
               estimator=Pipeline(memory=None,
              steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
                  intercept_scaling=1, max_iter=100, multi_class='warn',
                  n_jobs=None, penalty='l2', random_state=None, solver='warn',
                  tol=0.0001, verbose=0, warm_start=False))]),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'lg__penalty': ['l1']}, pre_dispatch='2*n_jobs',
               refit=True, return_train_score='warn', scoring='roc_auc',
               verbose=10)

In [33]: # The best parameters and accuracy score
         final_model = grid.best_estimator_
         y_pred = final_model.predict(X_test)
         print(grid.best_params_)
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))

{'lg__penalty': 'l1'}
Accurancy: 0.6716806341475061
```

3.0.2 Undersampling

```
In [34]: rus = RandomUnderSampler(replacement=False)
         X_train_subsample, y_train_subsample = rus.fit_sample(
             X.to_frame(), y)

         print(X_train_subsample.shape)
         print(np.bincount(y_train_subsample))
```

```
(129476, 1)
[64738 64738]
```

```
In [35]: url = []
         num = []
         for i in range(X_train_subsample.shape[0]):
             st = X_train_subsample[i][0]
            #print("st",st)
             result = findurl(st)
             result1 = findnum(st)
             if result != []:
                 url.append(1)
             else:
                 url.append(0)

             if result1 != []:
                 num.append(1)
             else:
                 num.append(0)

         r = [[num[i],url[i]] for i in range(len(num))]


         X_train_subsample = pd.Series(X_train_subsample.flatten()).str.replace('\d+','')
```

```
In [36]: vect = CountVectorizer(token_pattern=r"\b\w+\b", stop_words=ENGLISH_STOP_WORDS, min_d
         X_SS_cv = vect.fit_transform(X_train_subsample)
         X_SS_cv = hstack([X_SS_cv,r])
```

```
In [37]: undersample_pipe = Pipeline([
         #     ("cv", CountVectorizer(token_pattern= r"\b\w+\b",ngram_range=(1,3))),
             ("tf",TfidfTransformer()),
             ("lg", LogisticRegression())])

         X_train, X_test, y_train, y_test = train_test_split(X_SS_cv, y_train_subsample, strat
```

```
In [38]: undersample_pipe.fit(X_train, y_train)
         y_pred = undersample_pipe.predict(X_test)
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))
```

```
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


Accurancy: 0.7003614569495505


In [39]: param_grid = {
             #"cv__max_features":np.arange(2200,2500,100),
         #    "cv__token_pattern":[r"\b\w+\b"],
         #    "cv__min_df":np.arange(2,8,2),
         #    "cv__stop_words":[ENGLISH_STOP_WORDS],
         #    "cv__ngram_range": [(1,3)]
             "lg__C":np.logspace(-3,3,7)
         }
         grid = GridSearchCV(undersample_pipe, param_grid=param_grid, cv=3 , scoring="accuracy

         grid.fit(X_train, y_train)

Fitting 3 folds for each of 7 candidates, totalling 21 fits
[CV] lg__C=0.001 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV] ... lg__C=0.001, score=0.6774482545566883, total=   0.5s
[CV] lg__C=0.001 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.6s remaining:    0.0s


[CV] ... lg__C=0.001, score=0.6732676326114492, total=   0.4s
[CV] lg__C=0.001 ...


[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    1.1s remaining:    0.0s


[CV] ... lg__C=0.001, score=0.6777063766683143, total=   0.4s
[CV] lg__C=0.01 ...


[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    1.6s remaining:    0.0s
```

```
[CV] ... lg__C=0.01, score=0.6810318195860364, total=   0.4s
[CV] lg__C=0.01 ...

[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    2.1s remaining:    0.0s

[CV] ... lg__C=0.01, score=0.6766659458123513, total=   0.4s
[CV] lg__C=0.01 ...

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    2.6s remaining:    0.0s

[CV] ... lg__C=0.01, score=0.6795600593178448, total=   0.4s
[CV] lg__C=0.1 ...

[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    3.2s remaining:    0.0s

[CV] ... lg__C=0.1, score=0.6905468025949953, total=   0.6s
[CV] lg__C=0.1 ...

[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    3.9s remaining:    0.0s

[CV] ... lg__C=0.1, score=0.6861812227748771, total=   0.6s
[CV] lg__C=0.1 ...

[Parallel(n_jobs=1)]: Done   8 out of   8 | elapsed:    4.6s remaining:    0.0s

[CV] ... lg__C=0.1, score=0.6924740484429066, total=   0.6s
[CV] lg__C=1.0 ...

[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:    5.3s remaining:    0.0s

[CV] ... lg__C=1.0, score=0.6941921532282979, total=   0.9s
[CV] lg__C=1.0 ...
[CV] ... lg__C=1.0, score=0.6894250671939202, total=   0.9s
[CV] lg__C=1.0 ...
[CV] ... lg__C=1.0, score=0.6951309935739002, total=   1.0s
[CV] lg__C=10.0 ...
[CV] ... lg__C=10.0, score=0.6685511275872722, total=   1.9s
[CV] lg__C=10.0 ...
```

```
[CV] ... lg__C=10.0, score=0.6667799437733634, total=   1.9s
[CV] lg__C=10.0 ...
[CV] ... lg__C=10.0, score=0.6725469599604548, total=   2.0s
[CV] lg__C=100.0 ...
[CV] ... lg__C=100.0, score=0.6407784986098239, total=   4.1s
[CV] lg__C=100.0 ...
[CV] ... lg__C=100.0, score=0.6375544502456053, total=   4.0s
[CV] lg__C=100.0 ...
[CV] ... lg__C=100.0, score=0.6432896688087, total=   5.5s
[CV] lg__C=1000.0 ...
[CV] ... lg__C=1000.0, score=0.6251158480074143, total=   9.6s
[CV] lg__C=1000.0 ...
[CV] ... lg__C=1000.0, score=0.6227254471871235, total=   9.6s
[CV] lg__C=1000.0 ...
[CV] ... lg__C=1000.0, score=0.625370736529906, total=   8.8s


[Parallel(n_jobs=1)]: Done  21 out of  21 | elapsed:   56.8s finished


Out[39]: GridSearchCV(cv=3, error_score='raise-deprecating',
               estimator=Pipeline(memory=None,
              steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
                  intercept_scaling=1, max_iter=100, multi_class='warn',
                  n_jobs=None, penalty='l2', random_state=None, solver='warn',
                  tol=0.0001, verbose=0, warm_start=False))]),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'lg__C': array([1.e-03, 1.e-02, 1.e-01, 1.e+00, 1.e+01, 1.e+02, 1.e
               pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
               scoring='accuracy', verbose=10)

In [40]: # The best parameters and accuracy score
         final_model = grid.best_estimator_
         y_pred = final_model.predict(X_test)
         print(grid.best_params_ )
         print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))

{'lg__C': 1.0}
Accurancy: 0.7003614569495505
```

3.1 Different Models

3.1.1 Random Forest

```
In [43]: pipe = Pipeline([
         #     ("cv", CountVectorizer(max_features=2000,min_df=5,stop_words=ENGLISH_STOP_WORDS
             ("tf",TfidfTransformer()),
             ("rf", RandomForestClassifier(max_depth= 10, n_estimators=1000))])
```

```
param_grid = {
    "rf__n_estimators":np.arange(1000,1500,100),
    "rf__max_depth":np.arange(10,20,5),
    "rf__max_features": np.arange(1000,1500,100)
}
```

```
X_train, X_test, y_train, y_test = train_test_split(X_SS_cv, y_train_subsample, strati

pipe.fit(X_train, y_train)

# grid = GridSearchCV(pipe, param_grid=param_grid, cv=3, scoring="roc_auc", verbose =
# grid.fit(X_train, y_train)
```

Out[43]:
```
Pipeline(memory=None,
     steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
            max_depth=10, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_sp...obs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False))])
```

In [44]:
```
y_pred = pipe.predict(X_test)
accuracy_score(y_test, y_pred)
```

Out[44]:
```
0.6369674688745405
```

3.1.2 XGBoost

In [46]:
```
pipe = Pipeline([
    #("cv", CountVectorizer(min_df=5,ngram_range=(1,2),stop_words=ENGLISH_STOP_WORDS,
    ("tf",TfidfTransformer()),
    ("xgb", XGBClassifier())])

# params = {
#          'xgb__max_depth': np.arange(4,10,2),
#          'xgb__eta': [0.1,0.15,0.2],
#          'xgb__subsample': [0.5,1]
#         }

X_train, X_test, y_train, y_test = train_test_split(X_SS_cv, y_train_subsample, strati

pipe.fit(X_train, y_train)
# grid = GridSearchCV(pipe, param_grid=params, cv=3)
# grid.fit(X_train, y_train)
```

Out[46]:
```
Pipeline(memory=None,
     steps=[('tf', TfidfTransformer(norm='l2', smooth_idf=True, sublinear_tf=False, us
```

```
                colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,
                max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
                n_jobs=1, nthread=None, objective='binary:logistic', random_state=0,
                reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
                silent=True, subsample=1))])
```

In [47]:
```
y_pred = pipe.predict(X_test)
accuracy_score(y_test, y_pred)
```

Out[47]:
```
0.6404893571009299
```

4 Task 2 Word Vectors

In [53]:
```
#X_n is body data without numbers
WE = X_n.tolist()
sw = set(nltk.corpus.stopwords.words("english"))
lemmatizer = nltk.stem.WordNetLemmatizer()
stemmer = nltk.stem.PorterStemmer()

#preprocessing words lemmization and stemming
#returing X_we, which is the stemmed and lemmazied data
X_we = []
count = 0
for sentence in WE:
#     if count%100 == 0:
#         print(count)
    X_we.append([])
    s = nltk.word_tokenize(sentence)
    for word in s:
        if word.lower() not in sw and word.lower() not in string.punctuation:
            current = word.lower()
            current = lemmatizer.lemmatize(current)
            current = stemmer.stem(current)
            X_we[-1].append(current)
    #count += 1
```

In [54]:
```
# preparing the Word2Vec model
wmodel = Word2Vec(sentences=X_we,
                  sg=1,
                  hs=0,
                  workers=4,
                  size=200,
                  min_count=3,
                  window=6,
                  sample=1e-3,
                  negative=5
                 )
```

```
In [55]: #Coverting each sentence into vector points for classifier
         all_words = set(wmodel.wv.vocab.keys())
         X_we_vec = []
         for words in WE:
             X_we_vec.append([])
             vec = np.zeros(wmodel.vector_size)
             nwords = 0
             for word in words:
                 if word in all_words:
                     vec = np.add(vec, wmodel[word])
                 nwords += 1
             vec = np.divide(vec, nwords)
             X_we_vec[-1].append(vec.flatten())


/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:9: DeprecationWarning: Call to dep
  if __name__ == '__main__':


In [56]: #reformatting into dataframe
         temp = []
         for s in X_we_vec:
             temp.append([])
             for w in s[0]:
                 temp[-1].append(w)
         X_we_vec_df = pd.DataFrame(temp)

In [58]: #splitting data
         X_train, X_test, y_train, y_test = train_test_split(X_we_vec_df,y, random_state = 0)

In [59]: lg_we = LogisticRegression()
         lg_we.fit(X_train, y_train)
         lg_we.score(X_test,y_test)

/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Def
  FutureWarning)


Out[59]: 0.6368216221378602

In [ ]: pipe = Pipeline([
             ("lg", LogisticRegression())
        ])

        param_grid = {
                "lg__C":np.logspace(-1,1,3),
                "lg__penalty":["l1","l2"]
        }
```

```
        grid = GridSearchCV(pipe, param_grid=param_grid, scoring='roc_auc', cv=3, verbose=10)

        X_train, X_test, y_train, y_test = train_test_split(X_we_vec_df,y, random_state = 0)

        grid.fit(X_train, y_train)

        final_model = best.best_estimator_
        y_pred = final_model.predict(X_test)
        print(best.best_params_)
        print("Accurancy: {}".format(accuracy_score(y_test, y_pred)))
```

```
Fitting 3 folds for each of 6 candidates, totalling 18 fits
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.6161397177819223, total=  13.8s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   13.9s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.6175085388282047, total=  16.1s
[CV] lg__C=0.1, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:   30.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l1, score=0.6172458646402055, total=  14.3s
[CV] lg__C=0.1, lg__penalty=l2 ...


[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:   44.5s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l2, score=0.6161420799378785, total=   2.4s
[CV] lg__C=0.1, lg__penalty=l2 ...
```

```
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:   47.0s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l2, score=0.6178731476874055, total=   2.6s
[CV] lg__C=0.1, lg__penalty=l2 ...


[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:   49.6s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=0.1, lg__penalty=l2, score=0.6171897613922924, total=   2.5s
[CV] lg__C=1.0, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:   52.2s remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)


[CV]  lg__C=1.0, lg__penalty=l1, score=0.665838704544025, total= 2.2min
[CV] lg__C=1.0, lg__penalty=l1 ...


[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:  3.1min remaining:    0.0s
/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: De
  FutureWarning)
```