In [10]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import CountVectorizer, TfidfVector
izer
from sklearn import model_selection, naive_bayes, svm
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score
from tqdm import tqdm
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MaxAbsScaler
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import roc_auc_score
import os.path
import pickle
```

In [20]:
```
X_train = pd.read_pickle('../../../Preprocessing/Data/X_train.pkl')
X_test = pd.read_pickle('../../../Preprocessing/Data/X_test.pkl')
y_train = pd.read_pickle('../../../Preprocessing/Data/y_train.pkl')
y_test = pd.read_pickle('../../../Preprocessing/Data/y_test.pkl')
```

Before we do anything, we need to get the vectors. We can download one of the pre-trained models. We downloaded the pretrained model from http://nlp.stanford.edu/data/glove.6B.zip (http://nlp.stanford.edu/data/glove.6B.zip)

In [21]:
```
import numpy as np
w2v = {}

f = open("../Word2Vec_Data/glove.6B.50d.txt", "rb")

for line in f:
    w2v[line.split()[0]] = np.array(line.split()[1:]).astype(np.float)
```

In [22]:
```
words_not_found = 0
train_doc_vectors = pd.DataFrame() # creating empty final dataframe
if os.path.isfile('../Word2Vec_Data/train_doc_vectors.pkl'):
    train_doc_vectors = pd.read_pickle('../Word2Vec_Data/train_doc_vecto
rs.pkl')
else:
    for doc in tqdm(X_train.values): # looping through each document and
cleaning it
        temp = pd.DataFrame()  # creating a temporary dataframe(store va
lue for 1st doc & for 2nd doc remove the details of 1st & proced through
2nd and so on..)
        word_vec = np.zeros(50)
        temp = temp.append(pd.Series(word_vec), ignore_index = True) # i
f word is present then append it to temporary dataframe

        for word in doc.split(" "): # looping through each word of a sin
gle document and spliting through space
            word = word.encode("utf-8")
            try:
                word_vec = w2v[word] # if word is present in embeddings
(goole provides weights associate with words(300)) then proceed
                temp = temp.append(pd.Series(word_vec), ignore_index = T
rue) # if word is present then append it to temporary dataframe
            except:
                word_vec = np.zeros(50)
                words_not_found += 1
                temp = temp.append(pd.Series(word_vec), ignore_index = T
rue) # if word is present then append it to temporary dataframe
                pass
        doc_vector = temp.mean() # take the average of each column(w0, w
1, w2,.....w300)
        train_doc_vectors = train_doc_vectors.append(doc_vector, ignore_
index = True) # append each document value to the final dataframe
    train_doc_vectors.to_pickle("../Word2Vec_Data/train_doc_vectors.pkl"
)

print(train_doc_vectors.shape)
```

```
(39912, 50)
```

In [23]:
```
words_not_found_test = 0
test_doc_vectors = pd.DataFrame() # creating empty final dataframe
if os.path.isfile('../Word2Vec_Data/test_doc_vectors.pkl'):
    test_doc_vectors = pd.read_pickle('../Word2Vec_Data/test_doc_vector
s.pkl')
else:
    for doc in tqdm(X_test.values): # looping through each document and
 cleaning it
        temp = pd.DataFrame()  # creating a temporary dataframe(store va
lue for 1st doc & for 2nd doc remove the details of 1st & proced through
2nd and so on..)
        word_vec = np.zeros(50)
        temp = temp.append(pd.Series(word_vec), ignore_index = True) # i
f word is present then append it to temporary dataframe

        for word in doc.split(" "): # looping through each word of a sin
gle document and spliting through space
            word = word.encode("utf-8")
            try:
                word_vec = w2v[word] # if word is present in embeddings
(goole provides weights associate with words(300)) then proceed
                temp = temp.append(pd.Series(word_vec), ignore_index = T
rue) # if word is present then append it to temporary dataframe
            except:
                word_vec = np.zeros(50)
                words_not_found_test += 1
                temp = temp.append(pd.Series(word_vec), ignore_index = T
rue) # if word is present then append it to temporary dataframe
                pass
        doc_vector = temp.mean() # take the average of each column(w0, w
1, w2,.....w300)
        test_doc_vectors = test_doc_vectors.append(doc_vector, ignore_in
dex = True) # append each document value to the final dataframe
    test_doc_vectors.to_pickle("../Word2Vec_Data/test_doc_vectors.pkl")

print(test_doc_vectors.shape)
```

```
(19659, 50)
```

In [24]:
```
train_doc_vectors.fillna(0)
test_doc_vectors.fillna(0)
scaler = MaxAbsScaler()
# using averaged word embeddings
train_term_doc = scaler.fit_transform(train_doc_vectors)
test_term_doc = scaler.fit_transform(test_doc_vectors)
```

In [25]:
```
def logistic_regression_with_CV(label):
    if os.path.isfile('Models/ridge_lr_' + label + '_w2v.sav') and os.pa
th.isfile('Models/lasso_lr_' + label + '_w2v.sav'):
        ridge_logistic_regressor_grid_cv = pickle.load(open('Models/ridg
e_lr_' + label + '_w2v.sav', 'rb'))
        lasso_logistic_regressor_grid_cv = pickle.load(open('Models/lass
o_lr_' + label + '_w2v.sav', 'rb'))
    else:
        ridge_logistic_regressor = LogisticRegression(penalty="l2", solv
er="liblinear", max_iter = 2000)
        lasso_logistic_regressor = LogisticRegression(penalty="l1", solv
er="liblinear", max_iter = 2000)

        ridge_logistic_regressor_grid_cv = GridSearchCV(estimator=ridge_
logistic_regressor,
                                                        param_grid={'C':np.
logspace(-4, 4, 20)}, cv= 5, iid=False)
        lasso_logistic_regressor_grid_cv = GridSearchCV(estimator=lasso_
logistic_regressor,
                                                        param_grid={'C':np.
logspace(-4, 4, 20)}, cv= 5, iid=False)

        ridge_logistic_regressor_grid_cv.fit(train_term_doc, y_train[lab
el])
        lasso_logistic_regressor_grid_cv.fit(train_term_doc, y_train[lab
el])

        pickle.dump(ridge_logistic_regressor_grid_cv, open('Models/ridge
_lr_' + label + '_w2v.sav', 'wb'))
        pickle.dump(lasso_logistic_regressor_grid_cv, open('Models/lasso
_lr_' + label + '_w2v.sav', 'wb'))

    ridge_train_pred = ridge_logistic_regressor_grid_cv.predict(train_te
rm_doc)
    lasso_train_pred = lasso_logistic_regressor_grid_cv.predict(train_te
rm_doc)

    ridge_test_pred = ridge_logistic_regressor_grid_cv.predict(test_term
_doc)
    lasso_test_pred = lasso_logistic_regressor_grid_cv.predict(test_term
_doc)

    print(label + " Ridge Train Accuracy - " + str(ridge_logistic_regres
sor_grid_cv.score(train_term_doc, y_train[label])))
    print(label + " Lasso Train Accuracy - " + str(lasso_logistic_regres
sor_grid_cv.score(train_term_doc, y_train[label])) + '\n')

    print(label + " Ridge Train F1 Score - " + str(f1_score(y_train[labe
l], ridge_train_pred)))
    print(label + " Lasso Train F1 Score - " + str(f1_score(y_train[labe
l], lasso_train_pred)) + '\n')

    print(label + " Ridge Train ROC-AUC Score - " + str(roc_auc_score(y_
train[label], ridge_train_pred)))
    print(label + " Lasso Train ROC-AUC Score - " + str(roc_auc_score(y_
train[label], lasso_train_pred)) + '\n')
```

```
    print(label + " Ridge Test Accuracy - " + str(ridge_logistic_regress
or_grid_cv.score(test_term_doc, y_test[label])))
    print(label + " Lasso Test Accuracy - " + str(lasso_logistic_regress
or_grid_cv.score(test_term_doc, y_test[label])) + '\n')

    print(label + " Ridge Test F1 Score - " + str(f1_score(y_test[label
], ridge_test_pred)))
    print(label + " Lasso Test F1 Score - " + str(f1_score(y_test[label
], lasso_test_pred)) + '\n')

    print(label + " Ridge Test ROC-AUC Score - " + str(roc_auc_score(y_t
est[label], ridge_test_pred)))
    print(label + " Lasso Test ROC-AUC Score - " + str(roc_auc_score(y_t
est[label], lasso_test_pred)) + '\n\n')
```

In [26]:
```
label_cols = ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult', 'i
dentity_hate']
for label in label_cols:
    logistic_regression_with_CV(label)
```

```
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
```

```
toxic Ridge Train Accuracy - 0.8567097614752456
toxic Lasso Train Accuracy - 0.8567348165965123

toxic Ridge Train F1 Score - 0.6864755221753194
toxic Lasso Train F1 Score - 0.687199124726477

toxic Ridge Train ROC-AUC Score - 0.7755787448157945
toxic Lasso Train ROC-AUC Score - 0.7762289078191099

toxic Ridge Test Accuracy - 0.8598097563456941
toxic Lasso Test Accuracy - 0.8585889414517524

toxic Ridge Test F1 Score - 0.6866757617098681
toxic Lasso Test F1 Score - 0.6821404070432198

toxic Ridge Test ROC-AUC Score - 0.7756724605128862
toxic Lasso Test ROC-AUC Score - 0.7724120192205299


severe_toxic Ridge Train Accuracy - 0.9737422329124072
severe_toxic Lasso Train Accuracy - 0.9739426738825416

severe_toxic Ridge Train F1 Score - 0.2305433186490455
severe_toxic Lasso Train F1 Score - 0.25501432664756446

severe_toxic Ridge Train ROC-AUC Score - 0.5719634302498421
severe_toxic Lasso Train ROC-AUC Score - 0.581664532521584

severe_toxic Ridge Test Accuracy - 0.9719721247265883
severe_toxic Lasso Test Accuracy - 0.9722264611628262

severe_toxic Ridge Test F1 Score - 0.26238286479250333
severe_toxic Lasso Test F1 Score - 0.2680965147453083

severe_toxic Ridge Test ROC-AUC Score - 0.5891942359176479
severe_toxic Lasso Test ROC-AUC Score - 0.5911558941605408


obscene Ridge Train Accuracy - 0.9050160352776108
obscene Lasso Train Accuracy - 0.9050160352776108

obscene Ridge Train F1 Score - 0.5899405083829097
obscene Lasso Train F1 Score - 0.5898517797251974

obscene Ridge Train ROC-AUC Score - 0.7285465099794247
obscene Lasso Train ROC-AUC Score - 0.7284726721426786

obscene Ridge Test Accuracy - 0.8960272648659647
obscene Lasso Test Accuracy - 0.8957729284297269

obscene Ridge Test F1 Score - 0.5537117903930131
obscene Lasso Test F1 Score - 0.5521311475409836

obscene Ridge Test ROC-AUC Score - 0.7115051210483617
obscene Lasso Test ROC-AUC Score - 0.7106106666655351
```

```
threat Ridge Train Accuracy - 0.9914812587692925
threat Lasso Train Accuracy - 0.9914060934054921

threat Ridge Train F1 Score - 0.0
threat Lasso Train F1 Score - 0.017191977077363897

threat Ridge Train ROC-AUC Score - 0.5
threat Lasso Train ROC-AUC Score - 0.5043359535262604

threat Ridge Test Accuracy - 0.9929803143598351
threat Lasso Test Accuracy - 0.99287857978534

threat Ridge Test F1 Score - 0.0
threat Lasso Test F1 Score - 0.0410958904109589

threat Ridge Test ROC-AUC Score - 0.5
threat Lasso Test ROC-AUC Score - 0.5107414980077197


insult Ridge Train Accuracy - 0.8977500501102426
insult Lasso Train Accuracy - 0.8978252154740429

insult Ridge Train F1 Score - 0.5065892878732923
insult Lasso Train F1 Score - 0.5070116054158608

insult Ridge Train ROC-AUC Score - 0.6860466086747585
insult Lasso Train ROC-AUC Score - 0.6862513307318064

insult Ridge Test Accuracy - 0.8924156874713871
insult Lasso Test Accuracy - 0.8925174220458822

insult Ridge Test F1 Score - 0.4729628706703215
insult Lasso Test F1 Score - 0.4731987035651958

insult Ridge Test ROC-AUC Score - 0.6679969588333052
insult Lasso Test ROC-AUC Score - 0.6680556546130787


identity_hate Ridge Train Accuracy - 0.9760222489476849
identity_hate Lasso Train Accuracy - 0.9760222489476849

identity_hate Ridge Train F1 Score - 0.0
identity_hate Lasso Train F1 Score - 0.0

identity_hate Ridge Train ROC-AUC Score - 0.5
identity_hate Lasso Train ROC-AUC Score - 0.5

identity_hate Ridge Test Accuracy - 0.9772114553130882
identity_hate Lasso Test Accuracy - 0.9772114553130882

identity_hate Ridge Test F1 Score - 0.0
identity_hate Lasso Test F1 Score - 0.0

identity_hate Ridge Test ROC-AUC Score - 0.5
identity_hate Lasso Test ROC-AUC Score - 0.5
```

```
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/metrics/clas
sification.py:1437: UndefinedMetricWarning: F-score is ill-defined and
being set to 0.0 due to no predicted samples.
  'precision', 'predicted', average, warn_for)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/metrics/clas
sification.py:1437: UndefinedMetricWarning: F-score is ill-defined and
being set to 0.0 due to no predicted samples.
  'precision', 'predicted', average, warn_for)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator LogisticRegression from versi
on 0.21.3 when using version 0.21.2. This might lead to breaking code o
r invalid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/base.py:306:
UserWarning: Trying to unpickle estimator GridSearchCV from version 0.2
1.3 when using version 0.21.2. This might lead to breaking code or inva
lid results. Use at your own risk.
  UserWarning)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/metrics/clas
sification.py:1437: UndefinedMetricWarning: F-score is ill-defined and
being set to 0.0 due to no predicted samples.
  'precision', 'predicted', average, warn_for)
/Users/abhay/anaconda3/lib/python3.7/site-packages/sklearn/metrics/clas
sification.py:1437: UndefinedMetricWarning: F-score is ill-defined and
being set to 0.0 due to no predicted samples.
  'precision', 'predicted', average, warn_for)
```

In [ ]: