In [4]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfVector
izer
from sklearn import model_selection, naive_bayes, svm
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score
from tqdm import tqdm
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MaxAbsScaler
import os.path
import pickle
```

In [5]:
```
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')
```

In [6]:
```
vec = TfidfVectorizer(ngram_range=(1, 2), min_df=3,
                      max_df=0.9, strip_accents='unicode', use_idf=1, sm
ooth_idf=1, sublinear_tf=1)
df = pd.read_pickle('../../../Preprocessing/Data/preprocess_pickle.pkl')
df['comment_text_final'] = [" ".join(text) for text in df['comment_text_
final'].values]
vec = vec.fit(df['comment_text_final'])
```

In [7]:
```
train_term_doc = vec.transform(X_train)
test_term_doc = vec.transform(X_test)
```

In [8]:
```
scaler = MaxAbsScaler()
train_term_doc = scaler.fit_transform(train_term_doc)
test_term_doc = scaler.fit_transform(test_term_doc)
```

In [6]:
```
if os.path.isfile('Models/svm_poly.sav'):
    clf = pickle.load(open('Models/svm_poly.sav', 'rb'))
else:
    clf = svm.SVC(kernel = 'poly', gamma='scale', degree=1, verbose=1)
    clf.fit(train_term_doc, y_train['toxic'])
    pickle.dump(clf, open('Models/svm_poly.sav', 'wb'))
```

In [7]:
```
svm_pred = clf.predict(test_term_doc)
```

In [8]:
```
print('Accuracy SVM Linear Kernel:', accuracy_score(y_test['toxic'], svm
_pred))
print('F1 Score SVM Linear Kernel:', f1_score(y_test['toxic'], svm_pred
))
print('ROC-AUC Score SVM Linear Kernel:', roc_auc_score(y_test['toxic'],
svm_pred))
```

```
Accuracy SVM Linear Kernel: 0.9064041914644692
F1 Score SVM Linear Kernel: 0.7991704867932765
ROC-AUC Score SVM Linear Kernel: 0.8492133295324785
```

In [9]:
```
if os.path.isfile('Models/svm_rbf.sav'):
    clf_rbf = pickle.load(open('Models/svm_rbf.sav', 'rb'))
else:
    clf_rbf = svm.SVC(kernel = 'rbf', gamma='scale', verbose=1)
    clf_rbf.fit(train_term_doc, y_train['toxic'])
    pickle.dump(clf, open('Models/svm_rbf.sav', 'wb'))
```

```
[LibSVM]
```

In [10]:
```
svm_rbf_pred = clf_rbf.predict(test_term_doc)
```

In [11]:
```
print('Accuracy SVM RBF Kernel:', accuracy_score(y_test['toxic'], svm_rb
f_pred))
print('F1 Score SVM RBG Kernel:', f1_score(y_test['toxic'], svm_rbf_pred
))
print('ROC-AUC Score SVM RBF Kernel:', roc_auc_score(y_test['toxic'], sv
m_rbf_pred))
```

```
Accuracy SVM RBF Kernel: 0.8759346864031741
F1 Score SVM RBG Kernel: 0.6976571216065452
ROC-AUC Score SVM RBF Kernel: 0.7728846276718616
```

In [3]:
```
n_folds = 5
c_vals = np.power(float(10), range(-7, 7 + 1))
degree_of_poly_kernel = [1, 2, 3]
param_grid = {'C': c_vals, 'degree' : degree_of_poly_kernel}
grid_search = GridSearchCV(svm.SVC(kernel='poly', gamma='scale'), param_
grid, cv=n_folds, iid=False, n_jobs = -1, verbose=10)
```

In [17]:
```
if os.path.isfile('Models/svm_poly_cv.sav'):
    grid_search = pickle.load(open('Models/svm_poly_cv.sav', 'rb'))
else:
    grid_search.fit(train_term_doc, y_train['toxic'])
    pickle.dump(grid_search, open('Models/svm_poly_cv.sav', 'wb'))
```

```
Fitting 5 folds for each of 33 candidates, totalling 165 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent wor
kers.
[Parallel(n_jobs=-1)]: Done   1 tasks      | elapsed:  3.5min
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:  3.6min
[Parallel(n_jobs=-1)]: Done  17 tasks      | elapsed:  6.9min
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed: 10.6min
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed: 18.4min
[Parallel(n_jobs=-1)]: Done  48 tasks      | elapsed: 26.9min
[Parallel(n_jobs=-1)]: Done  61 tasks      | elapsed: 41.9min
[Parallel(n_jobs=-1)]: Done  74 tasks      | elapsed: 65.1min
[Parallel(n_jobs=-1)]: Done  89 tasks      | elapsed: 97.1min
[Parallel(n_jobs=-1)]: Done 104 tasks      | elapsed: 134.7min
[Parallel(n_jobs=-1)]: Done 121 tasks      | elapsed: 182.5min
[Parallel(n_jobs=-1)]: Done 138 tasks      | elapsed: 234.5min
[Parallel(n_jobs=-1)]: Done 159 out of 165 | elapsed: 291.8min remainin
g: 11.0min
[Parallel(n_jobs=-1)]: Done 165 out of 165 | elapsed: 304.3min finished
```

Out[17]:
```
GridSearchCV(cv=5, error_score='raise-deprecating',
             estimator=SVC(C=1.0, cache_size=200, class_weight=None, co
ef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='scale', kernel='poly', max_iter=-1,
                           probability=False, random_state=None, shrink
ing=True,
                           tol=0.001, verbose=False),
             iid=False, n_jobs=-1,
             param_grid={'C': array([1.e-05, 1.e-04, 1.e-03, 1.e-02, 1.
e-01, 1.e+00, 1.e+01, 1.e+02,
       1.e+03, 1.e+04, 1.e+05]),
                         'degree': [1, 2, 3]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=Fa
lse,
             scoring=None, verbose=10)
```

In [18]:
```
svm_poly_grid_search_predict = grid_search.predict(test_term_doc)
```

In [20]:
```
f1_score(y_test['toxic'], svm_poly_grid_search_predict)
```

Out[20]:
```
0.7991704867932765
```

In [22]:
```
grid_search.best_estimator_
```

Out[22]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=1, gamma='scale', kernel='pol
y',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```