Apply+SVM+to+Amazon+reviews+data+set+bow+tfidf

June 1, 2018

```
In [1]: !pip install imblearn
        !pip install gensim

Collecting imblearn
  Downloading https://files.pythonhosted.org/packages/81/a7/4179e6ebfd654bd0eac0b9c06125b8b4c96a
Collecting imbalanced-learn (from imblearn)
  Downloading https://files.pythonhosted.org/packages/80/a4/900463a3c0af082aed9c5a43f4ec317a9469
    100% || 153kB 3.2MB/s ta 0:00:01
Requirement already satisfied: numpy in /usr/local/lib/python3.6/site-packages (from imbalanced-
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.6/site-packages (from imba
Requirement already satisfied: scipy in /usr/local/lib/python3.6/site-packages (from imbalanced-
Installing collected packages: imbalanced-learn, imblearn
Successfully installed imbalanced-learn-0.3.3 imblearn-0.0
You are using pip version 9.0.1, however version 10.0.1 is available.You should consider upgradi
Collecting gensim
  Downloading https://files.pythonhosted.org/packages/33/33/df6cb7acdcec5677ed130f4800f67509d24d
    100% || 22.6MB 66kB/s  eta 0:00:01    58% |             | 13.2MB 42.6MB/s eta 0:00:01    97%
Requirement already satisfied: six>=1.5.0 in /usr/local/lib/python3.6/site-packages (from gensim
Requirement already satisfied: numpy>=1.11.3 in /usr/local/lib/python3.6/site-packages (from gen
Collecting smart-open>=1.2.1 (from gensim)
  Downloading https://files.pythonhosted.org/packages/4b/69/c92661a333f733510628f28b8282698b62cd
Requirement already satisfied: scipy>=0.18.1 in /usr/local/lib/python3.6/site-packages (from gen
Collecting boto>=2.32 (from smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/bd/b7/a88a67002b1185ed9a8e8a6ef15266728c23
    100% || 1.4MB 1.1MB/s eta 0:00:01
Collecting bz2file (from smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/61/39/122222b5e85cd41c391b68a99ee296584b2a
Requirement already satisfied: requests in /usr/local/lib/python3.6/site-packages (from smart-op
Collecting boto3 (from smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/25/54/288f3d87d055440e20eb7a9dce58fdd005c9
    100% || 133kB 9.3MB/s eta 0:00:01
Collecting s3transfer<0.2.0,>=0.1.10 (from boto3->smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/d7/14/2a0004d487464d120c9fb85313a75cd3d71a
    100% || 61kB 10.6MB/s ta 0:00:01
Collecting botocore<1.11.0,>=1.10.30 (from boto3->smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/da/aa/b227500e26dbbd95bd6cda78cf784f769bba
    100% || 4.3MB 371kB/s eta 0:00:01
```

```
Collecting jmespath<1.0.0,>=0.7.1 (from boto3->smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/b7/31/05c8d001f7f87f0f07289a5fc0fc3832e9a5
Collecting docutils>=0.10 (from botocore<1.11.0,>=1.10.30->boto3->smart-open>=1.2.1->gensim)
  Downloading https://files.pythonhosted.org/packages/36/fa/08e9e6e0e3cbd1d362c3bbee8d01d0aedb21
    100% || 552kB 2.6MB/s eta 0:00:01
Requirement already satisfied: python-dateutil<3.0.0,>=2.1; python_version >= "2.7" in /usr/loca
Building wheels for collected packages: smart-open, bz2file
  Running setup.py bdist_wheel for smart-open ... done
  Stored in directory: /root/.cache/pip/wheels/b1/9e/7d/bb3d3b55c597e72617140a0638c06382a5f17283
  Running setup.py bdist_wheel for bz2file ... done
  Stored in directory: /root/.cache/pip/wheels/81/75/d6/e1317bf09bf1af5a30befc2a007869fa6e1f516b
Successfully built smart-open bz2file
Installing collected packages: boto, bz2file, jmespath, docutils, botocore, s3transfer, boto3, s
Successfully installed boto-2.48.0 boto3-1.7.30 botocore-1.10.30 bz2file-0.98 docutils-0.14 gens
You are using pip version 9.0.1, however version 10.0.1 is available.You should consider upgradi
```

```
In [1]: from sklearn.model_selection import train_test_split

        from sklearn.grid_search import GridSearchCV
        from sklearn.grid_search import RandomizedSearchCV
        from scipy.stats import uniform
        from scipy.stats import norm

        from imblearn.over_sampling import SMOTE

        import sqlite3

        import pandas as pd
        import numpy as np

        from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.feature_extraction.text import CountVectorizer
        import gensim

        from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

        from sklearn.preprocessing import StandardScaler
```

```
/usr/local/lib/python3.6/site-packages/sklearn/cross_validation.py:41: DeprecationWarning: This
  "This module will be removed in 0.20.", DeprecationWarning)
/usr/local/lib/python3.6/site-packages/sklearn/grid_search.py:42: DeprecationWarning: This modul
  DeprecationWarning)
```

```
In [2]: con = sqlite3.connect('final.sqlite') # this is cleaned dataset
        final = pd.read_sql_query("""
        SELECT Score, Text_not_included
```

```
        FROM reviews
        """, con)[:2000]
```

```
In [3]: for i, seq in enumerate(final['Text_not_included']):
          final['Text_not_included'][i]=final['Text_not_included'][i].decode('UTF-8')
        X_train, X_test, y_train , y_test = train_test_split(final['Text_not_included'], final['
```

0.1 Generate Count BoW vectors

```
In [4]: count_vect = CountVectorizer(ngram_range=(1,2) )
        count_vect.fit(X_train)
        bow_train=count_vect.transform(X_train)
        bow_test=count_vect.transform(X_test)
```

0.2 Generate TF IDF vectors

```
In [5]: tf_idf_vect=TfidfVectorizer(ngram_range=(1,2), min_df=10, dtype=float)
        tf_idf_vect.fit(X_train)
        tf_idf_train=tf_idf_vect.transform(X_train)
        tf_idf_test=tf_idf_vect.transform(X_test)
```

```
/usr/local/lib/python3.6/site-packages/sklearn/feature_extraction/text.py:1089: FutureWarning: C
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

0.3 Upsampling followed by standardization

```
In [6]: # Upsampling minority class
        over_sampler = SMOTE(ratio='minority')
        bow_train_resampled, y_train_resampled = over_sampler.fit_sample(bow_train, y_train)
        tf_idf_train_resampled, y_train_resampled = over_sampler.fit_sample(tf_idf_train, y_trai

        scaler_bow=StandardScaler(with_mean=False)
        scaler_tf_idf=StandardScaler(with_mean=False)

        scaler_bow.fit(bow_train_resampled)
        scaler_tf_idf.fit(tf_idf_train_resampled)

        bow_train_scaled=scaler_bow.transform(bow_train_resampled)
        tf_idf_train_scaled=scaler_tf_idf.transform(tf_idf_train_resampled)

        bow_test_scaled=scaler_bow.transform(bow_test)
        tf_idf_test_scaled=scaler_tf_idf.transform(tf_idf_test)
```

```
/usr/local/lib/python3.6/site-packages/sklearn/utils/validation.py:444: DataConversionWarning: D
  warnings.warn(msg, DataConversionWarning)
```

```
In [7]: from sklearn.svm import SVC
```

0.4 Classification using count Bow

In [16]:
```
tuned_parameters = {'C': np.linspace(10.0, 20, 10, dtype=float), 'gamma' : np.linspace(

        #Using GridSearchCV
        gscv = GridSearchCV(SVC(), tuned_parameters, scoring = 'accuracy', cv=5)

        print(gscv.fit(bow_train_scaled, y_train_resampled))

        tuned_parameters = {'C' : uniform(10,20), 'gamma' : uniform(0,1)}

        #Using RandomizedSearchCV
        rscv = RandomizedSearchCV(SVC(), tuned_parameters, scoring = 'accuracy', cv=5, n_iter=2

        print(rscv.fit(bow_train_scaled, y_train_resampled))
```

```
GridSearchCV(cv=5, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params={}, iid=True, n_jobs=1,
       param_grid={'C': array([10.     , 11.11111, 12.22222, 13.33333, 14.44444, 15.55556,
       16.66667, 17.77778, 18.88889, 20.     ]), 'gamma': array([0.001, 0.112, 0.223, 0.334, 0.4
       1.   ])},
       pre_dispatch='2*n_jobs', refit=True, scoring='accuracy', verbose=0)
RandomizedSearchCV(cv=5, error_score='raise',
          estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
          fit_params={}, iid=True, n_iter=20, n_jobs=1,
          param_distributions={'C': <scipy.stats._distn_infrastructure.rv_frozen object at 0x7f5
          pre_dispatch='2*n_jobs', random_state=None, refit=True,
          scoring='accuracy', verbose=0)
```

In [17]:
```
predictions = gscv.best_estimator_.predict(bow_test_scaled)
        print(classification_report(y_test, predictions))
        print(confusion_matrix(y_test, predictions).T)
        tn, fp, fn, tp = confusion_matrix(y_test, predictions).ravel()

        print("TPR = {}\n TNR = {}\n FPR = {}\n FNR = {}".format(tp/(fn+tp), tn/(tn+fp), fp/(tn

        predictions = rscv.best_estimator_.predict(bow_test_scaled)
        print(classification_report(y_test, predictions))
        print(confusion_matrix(y_test, predictions).T)
        tn, fp, fn, tp = confusion_matrix(y_test, predictions).ravel()
```

```
        print("TPR = {}\n TNR = {}\n FPR = {}\n FNR = {}".format(tp/(fn+tp), tn/(tn+fp), fp/(tn
```

```
             precision    recall  f1-score   support

   negative       0.11      0.01      0.02        87
   positive       0.78      0.97      0.87       313

avg / total       0.63      0.77      0.68       400

[[  1   8]
 [ 86 305]]
TPR = 0.9744408945686901
 TNR = 0.011494252873563218
 FPR = 0.9885057471264368
 FNR = 0.025559105431309903
             precision    recall  f1-score   support

   negative       0.00      0.00      0.00        87
   positive       0.78      1.00      0.88       313

avg / total       0.61      0.78      0.69       400

[[  0   0]
 [ 87 313]]
TPR = 1.0
 TNR = 0.0
 FPR = 1.0
 FNR = 0.0


/usr/local/lib/python3.6/site-packages/sklearn/metrics/classification.py:1135: UndefinedMetricWa
  'precision', 'predicted', average, warn_for)
```

In [18]:
```
print(gscv.best_estimator_)
```

```
SVC(C=10.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [19]:
```
print(rscv.best_estimator_)
```

```
SVC(C=11.996771023901712, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.14509256003855375,
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
```

0.5 Classification using TF IDF

In [20]:
```
tuned_parameters = {'C': np.linspace(10.0, 20, 20, dtype=float), 'gamma' : np.linspace(
#Using GridSearchCV
gscv = GridSearchCV(SVC(), tuned_parameters, scoring = 'accuracy', cv=5)

print(gscv.fit(tf_idf_train_scaled, y_train_resampled))

tuned_parameters = {'C' : uniform(10,20), 'gamma' : uniform(0,1)}

#Using RandomizedSearchCV
rscv = RandomizedSearchCV(SVC(), tuned_parameters, scoring = 'accuracy', cv=5, n_iter=1

print(rscv.fit(tf_idf_train_scaled, y_train_resampled))
```

```
GridSearchCV(cv=5, error_score='raise',
       estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
       fit_params={}, iid=True, n_jobs=1,
       param_grid={'C': array([10.     , 10.52632, 11.05263, 11.57895, 12.10526, 12.63158,
       13.15789, 13.68421, 14.21053, 14.73684, 15.26316, 15.78947,
       16.31579, 16.84211, 17.36842, 17.89474, 18.42105, 18.94737,
       19.47368, 20.     ]), 'gamma': array([0.001, 0.112, 0.223, 0.334, 0.445, 0.556, 0.667, 0.
       1.   ])},
       pre_dispatch='2*n_jobs', refit=True, scoring='accuracy', verbose=0)
RandomizedSearchCV(cv=5, error_score='raise',
          estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False),
          fit_params={}, iid=True, n_iter=15, n_jobs=1,
          param_distributions={'C': <scipy.stats._distn_infrastructure.rv_frozen object at 0x7f5
          pre_dispatch='2*n_jobs', random_state=None, refit=True,
          scoring='accuracy', verbose=0)
```

In [21]:
```
predictions = gscv.best_estimator_.predict(tf_idf_test_scaled)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions).T)
tn, fp, fn, tp = confusion_matrix(y_test, predictions).ravel()

print("TPR = {}\n TNR = {}\n FPR = {}\n FNR = {}".format(tp/(fn+tp), tn/(tn+fp), fp/(tn

predictions = rscv.best_estimator_.predict(tf_idf_test_scaled)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions).T)
tn, fp, fn, tp = confusion_matrix(y_test, predictions).ravel()
```

```
        print("TPR = {}\n TNR = {}\n FPR = {}\n FNR = {}".format(tp/(fn+tp), tn/(tn+fp), fp/(tn
```

```
             precision    recall  f1-score   support

   negative       0.60      0.03      0.07        87
   positive       0.79      0.99      0.88       313

avg / total       0.75      0.79      0.70       400

[[  3   2]
 [ 84 311]]
TPR = 0.9936102236421726
 TNR = 0.034482758620689655
 FPR = 0.9655172413793104
 FNR = 0.006389776357827476
             precision    recall  f1-score   support

   negative       0.00      0.00      0.00        87
   positive       0.78      1.00      0.88       313

avg / total       0.61      0.78      0.69       400

[[  0   0]
 [ 87 313]]
TPR = 1.0
 TNR = 0.0
 FPR = 1.0
 FNR = 0.0


/usr/local/lib/python3.6/site-packages/sklearn/metrics/classification.py:1135: UndefinedMetricWa
  'precision', 'predicted', average, warn_for)
```

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

```
Out[22]: SVC(C=10.0, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
           max_iter=-1, probability=False, random_state=None, shrinking=True,
           tol=0.001, verbose=False)
```

```
In [23]: rscv.best_estimator_
```

```
Out[23]: SVC(C=29.836444326925047, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma=0.03469353355992577,
           kernel='rbf', max_iter=-1, probability=False, random_state=None,
           shrinking=True, tol=0.001, verbose=False)
```

0.6 Conclusions

Both BoW and TFIDF although provide excellent TPR, are failing at TNR. Also given large amount of time taken to train, SVMs combined with such high dimensional representations are not a good choice for text classification. Somewhat decent results are given by gamma : 0.001 $10 < C < 20$