

3. Post_Clean_TF-IDF

June 1, 2018

1 AAI Course: TSNE Visualization (Part III)

1.1 TF-IDF & t-SNE

Data Source: The preprocessing step has produced `final.sqlite` file after doing the data preparation & cleaning. The review text is now devoid of punctuations, HTML markups and stop words.

Objective: To plot t-SNE plot after doing TF-IDF & Truncated SVD for dimensionality reduction. The aim is to check whether there is a separation between data points.

1.2 Preprocessed Data Loading

```
In [1]: import sqlite3
import pdb
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

# using the SQLite Table to read data.
con = sqlite3.connect('./final.sqlite')

#filtering only positive and negative reviews i.e.
# not taking into consideration those reviews with Score=3
final = pd.read_sql_query("""
```

```
SELECT *
FROM Reviews
""", con)
```

```
print(final.head(3))
```

	index	Id	ProductId	UserId	ProfileName	\
0	138706	150524	0006641040	ACITT7DI6IDDL	shari zychinski	
1	138688	150506	0006641040	A2IW4PEEK02ROU	Tracy	
2	138689	150507	0006641040	A1S4A3IQ2MU7V4	sally sue "sally sue"	

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	\
0		0	0 positive	939340800	
1		1	1 positive	1194739200	
2		1	1 positive	1191456000	

	Summary	\
0	EVERY book is educational	
1	Love the book, miss the hard cover version	
2	chicken soup with rice months	

	Text	\
0	this witty little book makes my son laugh at l...	
1	I grew up reading these Sendak books, and watc...	
2	This is a fun way for children to learn their ...	

	CleanedText
0	b'witti littl book make son laugh loud recit c...
1	b'grew read sendak book watch realli rosi movi...
2	b'fun way children learn month year learn poem...

1.3 TF-IDF & Truncated SVD

TF-IDF will produce a high dimensional vector. If the number of features is very high, it is highly recommended to use another dimensionality reduction method.

For instance, **PCA for dense data or TruncatedSVD for sparse data**. Here, as the TF-IDF data is sparse, TruncatedSVD is used.

```
In [2]: #TF-IDF
```

```
tf_idf_vect = TfidfVectorizer()
final_tf_idf = tf_idf_vect.fit_transform(final['CleanedText'].values)
final_tf_idf.get_shape()
```

```
Out[2]: (364171, 71624)
```

```
In [3]: # TruncatedSVD
```

```

from sklearn.manifold import TSNE
import pdb

num_points = 10000
# Picking the top 1000 points as TSNE takes a lot of time for 15K points
data_1000 = final_tf_idf[0:num_points,:]
print(type(final_tf_idf))
labels_1000 = final['Score'].head(num_points)

from sklearn.decomposition import TruncatedSVD
from sklearn.random_projection import sparse_random_matrix
svd = TruncatedSVD(n_components=50, n_iter=10, random_state=42)
data_1000 = svd.fit_transform(data_1000)

print(data_1000.shape)

<class 'scipy.sparse.csr.csr_matrix'>
(10000, 50)

```

2 t-SNE Visualization

The output of Truncated SVD is fed into t-SNE for visualization.

In [4]: *# TSNE Plot after dimensionality reduction*

```

#from MulticoreTSNE import MulticoreTSNE as TSNE

#tsne = TSNE(n_jobs=1)#, n_components=2,
#             random_state=0, perplexity = 100, n_iter = 1000)
#tsne_data = tsne.fit_transform(data_1000)

model = TSNE(n_components=2, random_state=0,
             perplexity = 100, n_iter = 1000)
# configuring the parameteres
# the number of components = 2
# default perplexity = 30
# default learning rate = 200
# default Maximum number of iterations for the optimization = 1000

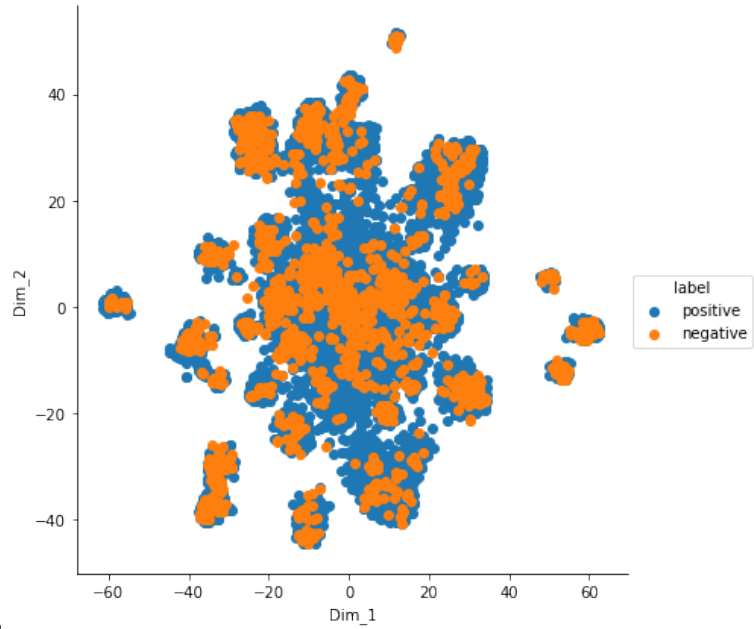
tsne_data = model.fit_transform(data_1000)

# creating a new data frame which help us in plotting the result data
tsne_data = np.vstack((tsne_data.T, labels_1000)).T
tsne_df = pd.DataFrame(data=tsne_data, columns=("Dim_1", "Dim_2", "label"))

# Ploting the result of tsne

```

```
sns.FacetGrid(tsne_df, hue="label", size=6).map(
    plt.scatter, 'Dim_1', 'Dim_2').add_legend()
plt.show()
```



3 Observations

1. The **overlap is still significant using TF-IDF method**. Not much improvement is seen, in terms of separation, compared to BoW-tSNE visualization.
2. Hence, **we will use Word2Vec dense vector representation** for reviews and bypass Truncated SVD before feeding in to t-SNE.