3.6 Featurizing text data with tfidf weighted word-vectors

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import re
import time
import warnings
import numpy as np
from nltk.corpus import stopwords
from sklearn.preprocessing import normalize
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
warnings.filterwarnings("ignore")
import sys
import os
import pandas as pd
import numpy as np
from tqdm import tqdm
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from scipy.sparse import coo_matrix, vstack

# exctract word2vec vectors
# https://github.com/explosion/spaCy/issues/1721
# http://landinghub.visualstudio.com/visual-cpp-build-tools
import spacy
```

In [2]:

```
# avoid decoding problems
df = pd.read_csv(r"D:\AppliedAI\Homework-n-Assignments\# 20 Quora\train.csv")
df = df.head(69999)
# encode questions to unicode
# https://stackoverflow.com/a/6812069
# ---------------- python 2 ---------------------
# df['question1'] = df['question1'].apply(lambda x: unicode(str(x),"utf-8"))
# df['question2'] = df['question2'].apply(lambda x: unicode(str(x),"utf-8"))
# ---------------- python 3 ---------------------
df['question1'] = df['question1'].apply(lambda x: str(x))
df['question2'] = df['question2'].apply(lambda x: str(x))
```

In [3]:

```
df.head()
```

Out[3]:

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

In [4]:

```
q1 = df['question1'].values
```

```
q2 = df ['question2'].values
Y = df['is_duplicate'].values

q1_train, q1_test, y_train, y_test = train_test_split(q1, Y, test_size=0.33, shuffle=False)
q1_train, q1_CV, y_train, y_CV = train_test_split(q1_train, y_train, test_size=0.33, shuffle=False)

q2_train, q2_test, = train_test_split(q2,test_size=0.33, shuffle=False)
q2_train, q2_CV = train_test_split(q2_train, test_size=0.33, shuffle=False)
```

In [5]:

```
tf_idf_vect = TfidfVectorizer(ngram_range=(1,3),max_features=5000)
tfidf_q1_train = tf_idf_vect.fit_transform(q1_train)
tfidf_q1_test = tf_idf_vect.transform(q1_test)
tfidf_q1_CV = tf_idf_vect.transform(q1_CV)

tfidf_q2_train = tf_idf_vect.fit_transform(q2_train)
tfidf_q2_test = tf_idf_vect.transform(q2_test)
tfidf_q2_CV = tf_idf_vect.transform(q2_CV)

q1_train_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q1_train)
q1_CV_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q1_CV)
q1_test_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q1_test)

q2_train_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q2_train)
q2_CV_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q2_CV)
q2_test_std =StandardScaler(with_mean=False,with_std=False).fit_transform(tfidf_q2_test)
```

In [6]:

```
print (q1_train_std.shape)
print (q1_CV_std.shape)
print (q1_test_std.shape)


newq1=vstack([q1_train_std, q1_CV_std,q1_test_std])
newq2=vstack([q2_train_std, q2_CV_std,q2_test_std])
```

```
(31422, 5000)
(15477, 5000)
(23100, 5000)
```

In [7]:

```
q1_arr = newq1.todense()
q2_arr = newq2.toarray()
```

In [8]:

```
if os.path.isfile(r'D:\AppliedAI\Homework-n-Assignments\# 20 Quora\nlp_features_train.csv'):
    dfnlp = pd.read_csv("nlp_features_train.csv",encoding='latin-1')
    dfnlp = dfnlp.head(69999)
else:
    print("download nlp_features_train.csv from drive or run previous notebook")

if os.path.isfile(r'D:\AppliedAI\Homework-n-Assignments\# 20
Quora\df_fe_without_preprocessing_train.csv'):
    dfppro = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-1')
    dfppro = dfppro.head(69999)

else:
    print("download df_fe_without_preprocessing_train.csv from drive or run previous notebook")
```

In [9]:

```
df1 = dfnlp.drop(['qid1','qid2','question1','question2'],axis=1)
df2 = dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
```

In [10]:

```
df3 = df.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
#df3_q1 = pd.DataFrame(q1_arr, index= df3.index)
#df3_q2 = pd.DataFrame(q2_arr, index= df3.index)

df3_q1 = pd.DataFrame(q1_arr)
df3_q2 = pd.DataFrame(q2_arr)
```

In [11]:

```
# dataframe of nlp features
df1.head()
```

Out[11]:

	id	is_duplicate	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_eq	abs_len_diff	
0	0	0	0.999980	0.833319	0.999983	0.999983	0.916659	0.785709	0.0	1.0	2.0	
1	1	0	0.799984	0.399996	0.749981	0.599988	0.699993	0.466664	0.0	1.0	5.0	
2	2	0	0.399992	0.333328	0.399992	0.249997	0.399996	0.285712	0.0	1.0	4.0	
3	3	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0	2.0	
4	4	0	0.399992	0.199998	0.999950	0.666644	0.571420	0.307690	0.0	1.0	6.0	

In [12]:

```
# data before preprocessing
df2.head()
```

Out[12]:

	id	freq_qid1	freq_qid2	q1len	q2len	q1_n_words	q2_n_words	word_Common	word_Total	word_share	freq_q1+q2	f
0	0	1	1	66	57	14	12	10.0	23.0	0.434783	2	0
1	1	4	1	51	88	8	13	4.0	20.0	0.200000	5	3
2	2	1	1	73	59	14	10	4.0	24.0	0.166667	2	0
3	3	1	1	50	65	11	9	0.0	19.0	0.000000	2	0
4	4	3	1	76	39	13	7	2.0	20.0	0.100000	4	2

In [13]:

```
df3_q1.head()
```

Out[13]:

	0	1	2	3	4	5	6	7	8	9	...	4990	4991	4992	4993	4994	4995	4996	4997	4998	4999
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 5000 columns

In [14]:

```
df3_q2.head()
```

Out[14]:

	0	1	2	3	4	5	6	7	8	9	...	4990	4991	4992	4993	4994	4995	4996	4997	4998	4999
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 5000 columns

In [15]:

```
print("Number of features in nlp dataframe :", df1.shape[1])
print("Number of features in preprocessed dataframe :", df2.shape[1])
print("Number of features in question1 w2v  dataframe :", df3_q1.shape[1])
print("Number of features in question2 w2v  dataframe :", df3_q2.shape[1])
print("Number of features in final dataframe  :", df1.shape[1]+df2.shape[1]+df3_q1.shape[1]+df3_q2.
shape[1])
```

```
Number of features in nlp dataframe : 17
Number of features in preprocessed dataframe : 12
Number of features in question1 w2v  dataframe : 5000
Number of features in question2 w2v  dataframe : 5000
Number of features in final dataframe  : 10029
```

In [16]:

```
# storing the final features to csv file
df3_q1['id']=df1['id']
df3_q2['id']=df1['id']
df1  = df1.merge(df2, on='id',how='left')
df2  = df3_q1.merge(df3_q2, on='id',how='left')
result  = df1.merge(df2, on='id',how='left')
result.to_csv('final_features.csv')
```