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 scipy.sparse import hstack
# 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("train.csv")
df=df[:5000]
# 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]:

```
#prepro_features_train.csv (Simple Preprocessing Feartures)
#nlp_features_train.csv (NLP Features)
if os.path.isfile('nlp_features_train.csv'):
    dfnlp = pd.read_csv("nlp_features_train.csv",encoding='latin-1')
    dfnlp=dfnlp[:5000]
else:
```

```
    print("download nlp_features_train.csv from drive or run previous notebook")

if os.path.isfile('df_fe_without_preprocessing_train.csv'):
    dfppro = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-1')
    dfppro=dfppro[:5000]
else:
    print("download df_fe_without_preprocessing_train.csv from drive or run previous notebook")
```

In [5]:

```
dfnlp.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1, inplace=True)
dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1, inplace=True)
data=pd.concat([df,dfnlp,dfppro],axis=1)
```

In [6]:

```
y_true = data['is_duplicate']
data.drop(['is_duplicate'], axis=1, inplace=True)
```

In [7]:

```
from sklearn.model_selection import train_test_split
X_tr,X_test, y_tr, y_test = train_test_split(data, y_true, stratify=y_true, test_size=0.3)
```

In [8]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
# merge texts
#questions_train = list(X_train['question1']) + list(X_train['question2'])
#questions_test = list(X_test['question1']) + list(X_test['question2'])

tfidf = TfidfVectorizer()
tfidf_ques1_tr= tfidf.fit_transform(X_tr['question1'].values)
tfidf_ques1_test= tfidf.transform(X_test['question1'].values)
tfidf = TfidfVectorizer()
tfidf_ques2_tr= tfidf.fit_transform(X_tr['question2'].values)
tfidf_ques2_test= tfidf.transform(X_test['question2'].values)

#tfidf_tr=hstack((tfidf_ques1_tr,tfidf_ques2_tr))
#tfidf_test=hstack((tfidf_ques1_test,tfidf_ques2_test))

X_tr['q1_feats_m'] = list(tfidf_ques1_tr.toarray())
X_test['q1_feats_m'] = list(tfidf_ques1_test.toarray())
X_tr['q2_feats_m'] = list(tfidf_ques2_tr.toarray())
X_test['q2_feats_m'] = list(tfidf_ques2_test.toarray())

# dict key:word and value:tf-idf score
#word2tfidf = dict(zip(tfidf.get_feature_names(), tfidf.idf_))
```

In [9]:

```
print("Number of data points in train data :",X_tr.shape)
print("Number of data points in test data :",X_test.shape)
```

```
Number of data points in train data : (3500, 35)
Number of data points in test data : (1500, 35)
```

- After we find TF-IDF scores, we convert each question to a weighted average of word2vec vectors by these scores.
- here we use a pre-trained GLOVE model which comes free with "Spacy". https://spacy.io/usage/vectors-similarity
- It is trained on Wikipedia and therefore, it is stronger in terms of word semantics.

In [10]:

```
X_tr.drop(['id','qid1','qid2','question1','question2'],axis=1,inplace=True)
X_test.drop(['id','qid1','qid2','question1','question2'],axis=1,inplace=True)
```

In [11]:

```
print("Number of data points in train data :",X_tr.shape)
print("Number of data points in test data :",X_test.shape)
```

```
Number of data points in train data : (3500, 28)
Number of data points in test data : (1500, 28)
```

In [12]:

```
X_q1_tr = pd.DataFrame(X_tr.q1_feats_m.values.tolist(), index= X_tr.index)
X_q1_test = pd.DataFrame(X_test.q1_feats_m.values.tolist(), index= X_test.index)
X_q2_tr = pd.DataFrame(X_tr.q2_feats_m.values.tolist(), index= X_tr.index)
X_q2_test = pd.DataFrame(X_test.q2_feats_m.values.tolist(), index= X_test.index)
```

In [13]:

```
print("Number of data points in train data q1 :",X_q1_tr.shape)
print("Number of data points in test data q1:",X_q1_test.shape)
print("Number of data points in train data q2:",X_q2_tr.shape)
print("Number of data points in test data q2:",X_q2_test.shape)
```

```
Number of data points in train data q1 : (3500, 6557)
Number of data points in test data q1: (1500, 6557)
Number of data points in train data q2: (3500, 6327)
Number of data points in test data q2: (1500, 6327)
```

In [14]:

```
X_tr.drop(['q1_feats_m','q2_feats_m'],axis=1,inplace=True)
X_test.drop(['q1_feats_m','q2_feats_m'],axis=1,inplace=True)
```

In [15]:

```
X_tr=pd.concat([X_tr,X_q1_tr,X_q2_tr],axis=1)
X_test=pd.concat([X_test,X_q1_test,X_q2_test],axis=1)
```

In [16]:

```
print("Number of features in nlp dataframe :", dfnlp.shape[1])
print("Number of features in preprocessed dataframe :", dfppro.shape[1])
print("Number of features in question1 w2v train dataframe :", X_q1_tr.shape[1])
print("Number of features in question2 w2v train dataframe :", X_q2_tr.shape[1])
print("Number of features in question1 w2v test dataframe :", X_q1_test.shape[1])
print("Number of features in question2 w2v test dataframe :", X_q2_test.shape[1])
print("Number of features in final dataframe  :", dfnlp.shape[1]+dfppro.shape[1]+X_q1_tr.shape[1]+X
_q2_tr.shape[1]+X_q1_test.shape[1]+X_q2_tr.shape[1])
```

```
Number of features in nlp dataframe : 16
Number of features in preprocessed dataframe : 12
Number of features in question1 w2v train dataframe : 6557
Number of features in question2 w2v train dataframe : 6327
Number of features in question1 w2v test dataframe : 6557
Number of features in question2 w2v test dataframe : 6327
Number of features in final dataframe  : 25796
```

In [17]:

```
X_tr_columns=X_tr.columns
X_test_columns=X_test.columns
y_tr_columns=y_tr.name
y_test_columns=y_test.name
```

In [18]:

```
# storing the final features to csv file
#https://stackoverflow.com/questions/37756991/best-way-to-join-two-large-datasets-in-pandas

X_tr.to_csv('X_tr.csv',index=False,header=X_tr_columns)
X_test.to_csv('X_test.csv',index=False,header=X_test_columns)
y_tr.to_csv('y_tr.csv',index=False,header=y_tr_columns)
```

```
y_cv.to_csv('y_cv.csv',index=False,header=y_cv_columns)
y_test.to_csv('y_test.csv',index=False,header=y_test_columns)
```

In [19]:

```
#def multiple_dfs(df_list, sheets, file_name, spaces):
#    writer = pd.ExcelWriter(file_name,engine='xlsxwriter')
#    row = 0
#    for dataframe in df_list:
#        dataframe.to_excel(writer,sheet_name=sheets,startrow=0, startcol=row)
#        row = row + len(dataframe.columns) + spaces + 1
#    writer.save()
#df_list=[df1,df2]
## run function
#multiple_dfs(df_list, 'Validation', 'test2.xlsx', 0)
```

In [20]:

```
#>>> from sqlalchemy import create_engine
#>>> engine = create_engine('sqlite:///data.db')
#X_tr.to_sql('X_tr', engine, if_exists='replace')
#X_test.to_sql('X_test', engine, if_exists='replace')
#y_tr.to_sql('y_tr', engine, if_exists='replace')
#y_test.to_sql('y_test', engine, if_exists='replace')
```