3.6 Featurizing text data with tfidf

In [5]:

```
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

import spacy
from scipy import sparse

from scipy.sparse import csr_matrix
import pickle
from sklearn.model_selection import train_test_split
```

In [134]:

```
# avoid decoding problems
df = pd.read_csv("train.csv")
df=df[:100000]
df['question1'] = df['question1'].apply(lambda x: str(x))
df['question2'] = df['question2'].apply(lambda x: str(x))
```

In [135]:

```
def do_pickling(filename,data):
    with open(filename, "wb") as f:
        pickle.dump(data,f)
```

In [136]:

```
y = df['is_duplicate']
```

In [137]:

```
y.shape
```

Out[137]:

```
(100000,)
```

In [181]:

```
df.head()
```

Out[181]:

	id	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_
0	0	0.999980	0.833319	0.999983	0.999983	0.916659	0.785709	0.0	
1	1	0.799984	0.399996	0.749981	0.599988	0.699993	0.466664	0.0	
2	2	0.399992	0.333328	0.399992	0.249997	0.399996	0.285712	0.0	
3	3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	
4	4	0.399992	0.199998	0.999950	0.666644	0.571420	0.307690	0.0	

5 rows × 29 columns

In [153]:

```
#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')
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')
else:
    print("download df_fe_without_preprocessing_train.csv from drive or run prev
ious notebook")
```

In [154]:

```
df1 = dfnlp.drop(['qid1','qid2','question1','question2'],axis=1)
df2 = dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
df3 = df.drop(['qid1','qid2','is_duplicate'],axis=1)
```

In [155]:

```
df1 = df1[:100000]
df2 = df2[:100000]
```

In [156]:

```
print(df1.shape)
print(df2.shape)
print(df3.shape)
```

```
(100000, 17)
(100000, 12)
(100000, 3)
```

In [157]:

```
df3.head()
```

Out[157]:

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

In [158]:

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

Out[158]:

	id	is_duplicate	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word
0	0	0	0.999980	0.833319	0.999983	0.999983	0.916659	0.785709	
1	1	0	0.799984	0.399996	0.749981	0.599988	0.699993	0.466664	
2	2	0	0.399992	0.333328	0.399992	0.249997	0.399996	0.285712	
3	3	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
4	4	0	0.399992	0.199998	0.999950	0.666644	0.571420	0.307690	

In [159]:

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

Out[159]:

	id	freq_qid1	freq_qid2	q1len	q2len	q1_n_words	q2_n_words	word_Common	word_T
0	0	1	1	66	57	14	12	10.0	2
1	1	4	1	51	88	8	13	4.0	2
2	2	1	1	73	59	14	10	4.0	2
3	3	1	1	50	65	11	9	0.0	1
4	4	3	1	76	39	13	7	2.0	2

In [160]:

```
df1  = df1.merge(df2, on='id',how='left')
df1 = df1.merge(df3,on ='id',how='left')
```

In [161]:

```
df1.shape
df=df1
y = df['is_duplicate']
df=df1.drop(['is_duplicate'],axis=1)
```

In [162]:

```
df.head()
```

Out[162]:

	id	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_
0	0	0.999980	0.833319	0.999983	0.999983	0.916659	0.785709	0.0	
1	1	0.799984	0.399996	0.749981	0.599988	0.699993	0.466664	0.0	
2	2	0.399992	0.333328	0.399992	0.249997	0.399996	0.285712	0.0	
3	3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	
4	4	0.399992	0.199998	0.999950	0.666644	0.571420	0.307690	0.0	

5 rows × 29 columns

In [163]:

```
X_train, test_df, y_train, y_test = train_test_split(df, y, stratify=y, test_siz
e=0.2)
train_df, cv_df, y, y_cv = train_test_split(X_train, y_train, stratify=y_train,
test_size=0.2)
```

In [164]:

```
tfidf = TfidfVectorizer()
train_1 = tfidf.fit_transform(train_df['question1'])
test_1 = tfidf.transform(test_df['question1'])
cv_1 = tfidf.transform(cv_df['question1'])
```

In [165]:

```
tfidf = TfidfVectorizer()
train_2 = tfidf.fit_transform(train_df['question2'])
test_2 = tfidf.transform(test_df['question2'])
cv_2 = tfidf.transform(cv_df['question2'])
```

In [166]:

```
train_2.shape
```

Out[166]:

```
(64000, 27759)
```

In [167]:

```
d4 = hstack((train_1,train_2))
d5 = hstack((test_1,test_2))
d6 = hstack((cv_1,cv_2))
```

In [168]:

```
train_df = train_df.drop(['question1','question2'],axis=1)
cv_df = cv_df.drop(['question1','question2'],axis=1)
test_df = test_df.drop(['question1','question2'],axis=1)
```

In [169]:

```
train_data = train_df.values
test_data = test_df.values
cv_data = cv_df.values
```

In [170]:

```
train_f = hstack([train_data,d4])
test_f = hstack([test_data,d5])
cv_f = hstack([cv_data,d6])
```

In [180]:

```
do_pickling('y_train.pickle',y)
do_pickling('y_test.pickle',y_test)
do_pickling('y_cv.pickle',y_cv)
do_pickling('train_f.pickle',train_f)
do_pickling('test_f.pickle',test_f)
do_pickling('cv_f.pickle',cv_f)
```