

3.6 Featurizing text data with tfidf weighted word-vectors

```
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
```

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



C:\Users\brahm\Anaconda3\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning: "This module will be removed in 0.20.", DeprecationWarning)

```
# avoid decoding problems
df = pd.read_csv("train.csv")

# 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))

df.head()
```



	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	2	1	What is the story of Kohinoor	What would happen if the Indian	0

```

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# merge texts
questions = list(df['question1']) + list(df['question2'])

tfidf = TfidfVectorizer(lowercase=False, )
tfidf.fit_transform(questions)

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

```

- 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.

```

# en_vectors_web_lg, which includes over 1 million unique vectors.
nlp = spacy.load('en_core_web_sm')

vecs1 = []
# https://github.com/noamraph/tqdm
# tqdm is used to print the progress bar
for qu1 in tqdm(list(df['question1'])):
    doc1 = nlp(qu1)
    # 384 is the number of dimensions of vectors
    mean_vec1 = np.zeros([len(doc1), len(doc1[0].vector)])
    for word1 in doc1:
        # word2vec
        vec1 = word1.vector
        # fetch df score
        try:
            idf = word2tfidf[str(word1)]
        except:
            idf = 0
        # compute final vec
        mean_vec1 += vec1 * idf
    mean_vec1 = mean_vec1.mean(axis=0)
    vecs1.append(mean_vec1)
df['q1_feats_m'] = list(vecs1)

```



100%

404290/46

```
vecs2 = []
for qu2 in tqdm(list(df['question2'])):
    doc2 = nlp(qu2)
    mean_vec1 = np.zeros([len(doc1), len(doc2[0].vector)])
    for word2 in doc2:
        # word2vec
        vec2 = word2.vector
        # fetch df score
        try:
            idf = word2tfidf[str(word2)]
        except:
            #print word
            idf = 0
        # compute final vec
        mean_vec2 += vec2 * idf
    mean_vec2 = mean_vec2.mean(axis=0)
    vecs2.append(mean_vec2)
df['q2_feats_m'] = list(vecs2)
```



100%

404290/46

```
#prepro_features_train.csv (Simple Preprocessing Features)
#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 previous notebook")

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','question1','question2','is_duplicate'],axis=1)
df3_q1 = pd.DataFrame(df3.q1_feats_m.values.tolist(), index= df3.index)
df3_q2 = pd.DataFrame(df3.q2_feats_m.values.tolist(), index= df3.index)

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



	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	

data before preprocessing

df2.head()



	id	freq_qid1	freq_qid2	q1len	q2len	q1_n_words	q2_n_words	word_Common	word_Tot
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

Questions 1 tfidf weighted word2vec

df3_q1.head()



	0	1	2	3	4	5	6
0	121.929927	100.083900	72.497894	115.641800	-48.370870	34.619058	-172.057787
1	-78.070939	54.843781	82.738482	98.191872	-51.234859	55.013510	-39.140730
2	-5.355015	73.671810	14.376365	104.130241	1.433537	35.229116	-148.519385
3	5.778359	-34.712038	48.999631	59.699204	40.661263	-41.658731	-36.808594
4	51.138220	38.587312	123.639488	53.333041	-47.062739	37.356212	-298.722753

5 rows × 384 columns

Questions 2 tfidf weighted word2vec

df3_q2.head()



	0	1	2	3	4	5	6
0	125.983301	95.636485	42.114702	95.449980	-37.386295	39.400078	-148.116070

```
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 : 384
Number of features in question2 w2v dataframe : 384
Number of features in final dataframe : 794
```

```
# storing the final features to csv file
if not os.path.isfile('final_features.csv'):
    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')
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.