Лабораторная работа №6

Васильев А.Р. ИУ5-24М

Цель лабораторной работы: изучение методов классификации текстов.

Требования к отчету:

Отчет по лабораторной работе должен содержать:

- титульный лист;
- описание задания;
- текст программы;
- экранные формы с примерами выполнения программы.

Задание - для произвольного набора данных, предназначенного для классификации текстов, решите задачу классификации текста двумя способами:

- Способ 1. На основе CountVectorizer или TfidfVectorizer.
- Способ 2. На основе моделей word2vec или Glove или fastText.

In [1]:
```
import nltk
import spacy
import numpy as np
from sklearn.datasets import fetch_20newsgroups
nltk.download('punkt')
from nltk import tokenize
import re
import pandas as pd
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Будем использовать датасет 20 newsgroups

In [2]:
```
categories = ["rec.autos", "rec.sport.hockey", "sci.crypt", "sci.med", "talk.
newsgroups_train = fetch_20newsgroups(subset='train', categories=categories)
newsgroups_test = fetch_20newsgroups(subset='test', categories=categories)
```

In [3]:
```
unique, frequency = np.unique(newsgroups_train.target,
                              return_counts = True)
```

In [4]:
```
for l, f in zip(unique, frequency):
  print(f'value: {l}, count: {f}')
```

```
value: 0, count: 594
value: 1, count: 600
value: 2, count: 595
value: 3, count: 594
value: 4, count: 377
```

In [5]:
```
print('Tokenizers NLTK have')
for i in dir(tokenize)[:16]:
  print(i)
```

```
Tokenizers NLTK have
BlanklineTokenizer
LineTokenizer
MWETokenizer
PunktSentenceTokenizer
RegexpTokenizer
ReppTokenizer
SExprTokenizer
SpaceTokenizer
StanfordSegmenter
TabTokenizer
TextTilingTokenizer
ToktokTokenizer
TreebankWordTokenizer
TweetTokenizer
WhitespaceTokenizer
WordPunctTokenizer
```

Подготовка текстов

In [6]:
```
from spacy.lang.en import English
import spacy
from nltk.corpus import stopwords

nlp = spacy.load("en_core_web_sm", disable=["parser", "ner"])
nltk.download('stopwords')
stopwords_eng = set(stopwords.words('english'))
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [7]:
```
def prepare(t):
  # t = ' '.join([i.strip().lower() for i in t.split(' ')])
  t = re.sub(r'[^a-zA-Z0-9 \n]', '', t)
  t = re.sub('\s+', ' ', t)
  t = ' '.join([token.lemma_.lower() for token in nlp(t) if token not in stop
  return t

texts = newsgroups_train.data

texts_array = []

for text in texts:
  prepared_text = prepare(text)
  texts_array.append(prepared_text)
```

In [8]:
```
len(texts_array), texts_array[-1]
```

Out[8]:
```
(2760,
 'from amolitormoinknmsuedu andrew molitor subject what the clipper naysayer
sound like to -pron- organization department of mathematical sciences lines 5
5 distribution na nntppostinghost moinknmsuedu originator amolitormoinknmsued
u the follow be available in some ftp archive somewhere -pron- insert -pron-
comment liberally throughout this demonic memo of big brotherdom white house
announcement on screw thread standards this be to announce that the american
national standards institute or whatever -pron- be have be give the authority
to define standard dimension for screw thread look this be clearly the first
```

```
step toward outlaw -pron- own screw thread specification if this madness be n
ot fight tooth and nail every step of the way -pron- will be a crime to use s
crew thread other than those -pron- fearless leader so graciously define for
-pron- the purpose of this be to permit industry to draw upon a standard pool
of specification and designation to ensure interoperability of various thread
object across vendor rubbish -pron- say ansi standard screw thread will have
subtle weakness allow -pron- agent to disassemble -pron- automobile more easi
ly cause -pron- muffler to fall off at inoppurtune moment questions and answe
rs on the ansi screw thread standard q will the screw thread define by ansi b
e as good as other screw thread design available elsewhere a yes hah trust -p
ron- q will -pron- be able to use -pron- own screw thread if -pron- desire a
of course but this will make -pron- thread object unlikely to interoperate co
rrectly with other within the industry see see this be the first step -pron-
be clear -pron- must band together write -pron- congressman use pretty good s
crew thread not this devilinspire ansi trash protect -pron- constitutional ri
ght to use whatever screw thread -pron- desire guerilla screw thread activism
must become the order of the day boycott gm and build -pron- own car use scre
w from stz screw thread associates screw -pron- bill clinton -pron- and -pron
- totalitarianist thug amolitornmsuedu finger for pgst personal screw thread
pitch or screw thread see the screw thread server must be free')
```

In [9]:
```
test_texts_arr = []

test_texts = newsgroups_test.data

for text in test_texts:
  prepared_text = prepare(text)
  test_texts_arr.append(prepared_text)
```

Способ 1 На основе CountVectorizer и TfidfVectorizer

In [10]:
```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
```

In [11]:
```
tfidf_vectorizer = TfidfVectorizer()

train_feature_matrix_tfidf = tfidf_vectorizer.fit_transform(texts_array)
test_feature_matrix__tfidf = tfidf_vectorizer.transform(test_texts_arr)
```

In [12]:
```
count_vectorizer = CountVectorizer()

train_feature_matrix_count = count_vectorizer.fit_transform(texts_array)
test_feature_matrix_count = count_vectorizer.transform(test_texts_arr)
```

In [13]:
```
target_values_train = newsgroups_train.target
target_values_test = newsgroups_test.target
```

knn with count vectorizer

In [31]:
```
from sklearn.model_selection import GridSearchCV

knn_count = KNeighborsClassifier()


parameters = {'n_neighbors': [2, 3, 5, 7, 9, 11]}

knn_count_grid = GridSearchCV(knn_count, parameters, scoring='roc_auc_ovr_wei
```

```
                                   verbose=4, cv=5)

knn_count_grid.fit(train_feature_matrix_count, target_values_train)

# print(pd.DataFrame(knn_count_grid.cv_results_))
print('best param of n_neighbors', knn_count_grid.best_params_['n_neighbors'])
best_knn_count = KNeighborsClassifier(n_neighbors=knn_count_grid.best_params_
print(best_knn_count)
best_knn_count.fit(train_feature_matrix_count, target_values_train)
best_knn_pred_count = best_knn_count.predict(test_feature_matrix_count)

print(classification_report(target_values_test, pred_count))
```

```
Fitting 5 folds for each of 6 candidates, totalling 30 fits
[CV] n_neighbors=2 ...................................................
[CV] ....................... n_neighbors=2, score=0.844, total=   0.2s
[CV] n_neighbors=2 ...................................................
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.2s remaining:    0.
0s
[CV] ....................... n_neighbors=2, score=0.837, total=   0.2s
[CV] n_neighbors=2 ...................................................
[CV] ....................... n_neighbors=2, score=0.861, total=   0.2s
[CV] n_neighbors=2 ...................................................
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.3s remaining:    0.
0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.5s remaining:    0.
0s
[CV] ....................... n_neighbors=2, score=0.833, total=   0.2s
[CV] n_neighbors=2 ...................................................
[CV] ....................... n_neighbors=2, score=0.842, total=   0.2s
[CV] n_neighbors=3 ...................................................
[CV] ....................... n_neighbors=3, score=0.862, total=   0.2s
[CV] n_neighbors=3 ...................................................
[CV] ....................... n_neighbors=3, score=0.858, total=   0.2s
[CV] n_neighbors=3 ...................................................
[CV] ....................... n_neighbors=3, score=0.863, total=   0.2s
[CV] n_neighbors=3 ...................................................
[CV] ....................... n_neighbors=3, score=0.849, total=   0.2s
[CV] n_neighbors=3 ...................................................
[CV] ....................... n_neighbors=3, score=0.858, total=   0.2s
[CV] n_neighbors=5 ...................................................
[CV] ....................... n_neighbors=5, score=0.864, total=   0.2s
[CV] n_neighbors=5 ...................................................
[CV] ....................... n_neighbors=5, score=0.858, total=   0.2s
[CV] n_neighbors=5 ...................................................
[CV] ....................... n_neighbors=5, score=0.867, total=   0.2s
[CV] n_neighbors=5 ...................................................
[CV] ....................... n_neighbors=5, score=0.854, total=   0.2s
[CV] n_neighbors=5 ...................................................
[CV] ....................... n_neighbors=5, score=0.869, total=   0.2s
[CV] n_neighbors=7 ...................................................
[CV] ....................... n_neighbors=7, score=0.865, total=   0.2s
[CV] n_neighbors=7 ...................................................
[CV] ....................... n_neighbors=7, score=0.863, total=   0.2s
[CV] n_neighbors=7 ...................................................
[CV] ....................... n_neighbors=7, score=0.871, total=   0.2s
[CV] n_neighbors=7 ...................................................
[CV] ....................... n_neighbors=7, score=0.840, total=   0.2s
[CV] n_neighbors=7 ...................................................
[CV] ....................... n_neighbors=7, score=0.873, total=   0.2s
[CV] n_neighbors=9 ...................................................
[CV] ....................... n_neighbors=9, score=0.861, total=   0.2s
[CV] n_neighbors=9 ...................................................
[CV] ....................... n_neighbors=9, score=0.862, total=   0.2s
[CV] n_neighbors=9 ...................................................
[CV] ....................... n_neighbors=9, score=0.874, total=   0.2s
```

```
[CV] n_neighbors=9 ....................................................
[CV] ....................... n_neighbors=9, score=0.838, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ....................... n_neighbors=9, score=0.871, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] ...................... n_neighbors=11, score=0.861, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] ...................... n_neighbors=11, score=0.857, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] ...................... n_neighbors=11, score=0.872, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] ...................... n_neighbors=11, score=0.841, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] ...................... n_neighbors=11, score=0.870, total=   0.2s
best param of n_neighbors 5
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
[Parallel(n_jobs=1)]: Done  30 out of  30 | elapsed:    5.1s finished
              precision    recall  f1-score   support

           0       0.38      0.71      0.49       396
           1       0.64      0.55      0.59       399
           2       0.63      0.53      0.58       396
           3       0.53      0.36      0.43       396
           4       0.57      0.35      0.44       251

    accuracy                           0.51      1838
   macro avg       0.55      0.50      0.51      1838
weighted avg       0.55      0.51      0.51      1838
```

knn with tfidf vectorizer

In [30]:

```
knn_tfidf = KNeighborsClassifier()


parameters = {'n_neighbors': [2, 3, 5, 7, 9, 11]}

knn_tfidf_grid = GridSearchCV(knn_tfidf, parameters,
                              scoring='roc_auc_ovr_weighted',
                              verbose=4, cv=5)

knn_tfidf_grid.fit(train_feature_matrix_count, target_values_train)

print('best param of n_neighbors', knn_count_grid.best_params_['n_neighbors']
best_knn_tfidf = KNeighborsClassifier(n_neighbors=knn_count_grid.best_params_

best_knn_tfidf.fit(train_feature_matrix_tfidf, target_values_train)
best_pred_knn = best_knn_tfidf.predict(test_feature_matrix__tfidf)

print(classification_report(target_values_test, best_pred_knn))
```

```
Fitting 5 folds for each of 6 candidates, totalling 30 fits
[CV] n_neighbors=2 ....................................................
[CV] ....................... n_neighbors=2, score=0.844, total=   0.2s
[CV] n_neighbors=2 ....................................................
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worke
rs.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.2s remaining:    0.
0s
[CV] ....................... n_neighbors=2, score=0.837, total=   0.2s
[CV] n_neighbors=2 ....................................................
[CV] ....................... n_neighbors=2, score=0.861, total=   0.2s
[CV] n_neighbors=2 ....................................................
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.3s remaining:    0.
```

```
0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.5s remaining:    0.
0s
[CV] ..................... n_neighbors=2, score=0.833, total=   0.2s
[CV] n_neighbors=2 ....................................................
[CV] ..................... n_neighbors=2, score=0.842, total=   0.2s
[CV] n_neighbors=3 ....................................................
[CV] ..................... n_neighbors=3, score=0.862, total=   0.2s
[CV] n_neighbors=3 ....................................................
[CV] ..................... n_neighbors=3, score=0.858, total=   0.2s
[CV] n_neighbors=3 ....................................................
[CV] ..................... n_neighbors=3, score=0.863, total=   0.2s
[CV] n_neighbors=3 ....................................................
[CV] ..................... n_neighbors=3, score=0.849, total=   0.2s
[CV] n_neighbors=3 ....................................................
[CV] ..................... n_neighbors=3, score=0.858, total=   0.2s
[CV] n_neighbors=5 ....................................................
[CV] ..................... n_neighbors=5, score=0.864, total=   0.2s
[CV] n_neighbors=5 ....................................................
[CV] ..................... n_neighbors=5, score=0.858, total=   0.2s
[CV] n_neighbors=5 ....................................................
[CV] ..................... n_neighbors=5, score=0.867, total=   0.2s
[CV] n_neighbors=5 ....................................................
[CV] ..................... n_neighbors=5, score=0.854, total=   0.2s
[CV] n_neighbors=5 ....................................................
[CV] ..................... n_neighbors=5, score=0.869, total=   0.2s
[CV] n_neighbors=7 ....................................................
[CV] ..................... n_neighbors=7, score=0.865, total=   0.2s
[CV] n_neighbors=7 ....................................................
[CV] ..................... n_neighbors=7, score=0.863, total=   0.2s
[CV] n_neighbors=7 ....................................................
[CV] ..................... n_neighbors=7, score=0.871, total=   0.2s
[CV] n_neighbors=7 ....................................................
[CV] ..................... n_neighbors=7, score=0.840, total=   0.2s
[CV] n_neighbors=7 ....................................................
[CV] ..................... n_neighbors=7, score=0.873, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ..................... n_neighbors=9, score=0.861, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ..................... n_neighbors=9, score=0.862, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ..................... n_neighbors=9, score=0.874, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ..................... n_neighbors=9, score=0.838, total=   0.2s
[CV] n_neighbors=9 ....................................................
[CV] ..................... n_neighbors=9, score=0.871, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] .................... n_neighbors=11, score=0.861, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] .................... n_neighbors=11, score=0.857, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] .................... n_neighbors=11, score=0.872, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] .................... n_neighbors=11, score=0.841, total=   0.2s
[CV] n_neighbors=11 ...................................................
[CV] .................... n_neighbors=11, score=0.870, total=   0.2s
best param of n_neighbors 3
[Parallel(n_jobs=1)]: Done  30 out of  30 | elapsed:    5.1s finished
              precision    recall  f1-score   support

           0       0.91      0.93      0.92       396
           1       0.97      0.89      0.93       399
           2       0.83      0.88      0.86       396
           3       0.95      0.64      0.76       396
           4       0.57      0.87      0.69       251

    accuracy                           0.84      1838
   macro avg       0.85      0.84      0.83      1838
```

```
weighted avg       0.87      0.84      0.84      1838
```

Способ 2 На основе моделей word2vec или Glove или fastText.

In [7]:
```
import tqdm
from gensim.models import Word2Vec
import gensim.downloader
# gensim.downloader.info()
# glove_vectors = gensim.downloader.load('glove-twitter-25')
glove_vectors = gensim.downloader.load('glove-wiki-gigaword-50')
```

In [8]:
```
class GloveTokenizer:
  def __init__(self, glove_tokenizer):
    self.glove = glove_tokenizer
    self.token_length = 800
    self.embedding_size = 50

  def __getitem__(self, word):
    try:
      vector = glove_vectors.get_vector(word).reshape(1, self.embedding_size)
    except KeyError as e:
      vector = np.zeros((1, self.embedding_size))
    return vector


  def __padd(self, sentence):
    padded_sentence = np.zeros((self.token_length, self.embedding_size))
    for i, token in enumerate(sentence):
        padded_sentence[i] = token
    return padded_sentence

  def tokenize(self, sentence):
    encoded_sentence = []
    sentence = sentence.strip(' ').split(' ')
    for i in sentence:
      token = self.__getitem__(i)
      encoded_sentence.append(token)
    return np.array(self.__padd(encoded_sentence), dtype=np.float16)

tokenizer = GloveTokenizer(glove_vectors)
```

In [9]:
```
def prepare(t):
  # t = ' '.join([i.strip().lower() for i in t.split(' ')])
  t = re.sub(r'[^a-zA-Z0-9 ]', '', t)
  t = re.sub('\s+', ' ', t)
  lemmas = [token.lemma_.lower() for token in nlp(t) if token not in stopword
  t = ' '.join(lemmas)
  vectors = tokenizer.tokenize(t)
  return vectors, len(lemmas)

vectors_array_train = []
labels_train = []

for enum, text, label in zip(range(len(newsgroups_train.data)), newsgroups_tr
  try:
    vector, length = prepare(text)
    # print(vector, vector.shape)
    vectors_array_train.append(vector)
```

```
    labels_train.append(label)
  except IndexError as e:
    print(enum, e)
    continue


vectors_array_train = np.array(vectors_array_train)
print(vectors_array_train.shape)
train_data = vectors_array_train.reshape((-1, vectors_array_train.shape[1]*ve
train_data.shape
```

```
56 index 800 is out of bounds for axis 0 with size 800
58 index 800 is out of bounds for axis 0 with size 800
93 index 800 is out of bounds for axis 0 with size 800
112 index 800 is out of bounds for axis 0 with size 800
147 index 800 is out of bounds for axis 0 with size 800
159 index 800 is out of bounds for axis 0 with size 800
214 index 800 is out of bounds for axis 0 with size 800
215 index 800 is out of bounds for axis 0 with size 800
217 index 800 is out of bounds for axis 0 with size 800
222 index 800 is out of bounds for axis 0 with size 800
265 index 800 is out of bounds for axis 0 with size 800
268 index 800 is out of bounds for axis 0 with size 800
281 index 800 is out of bounds for axis 0 with size 800
336 index 800 is out of bounds for axis 0 with size 800
361 index 800 is out of bounds for axis 0 with size 800
395 index 800 is out of bounds for axis 0 with size 800
412 index 800 is out of bounds for axis 0 with size 800
416 index 800 is out of bounds for axis 0 with size 800
424 index 800 is out of bounds for axis 0 with size 800
462 index 800 is out of bounds for axis 0 with size 800
468 index 800 is out of bounds for axis 0 with size 800
480 index 800 is out of bounds for axis 0 with size 800
568 index 800 is out of bounds for axis 0 with size 800
574 index 800 is out of bounds for axis 0 with size 800
585 index 800 is out of bounds for axis 0 with size 800
587 index 800 is out of bounds for axis 0 with size 800
588 index 800 is out of bounds for axis 0 with size 800
591 index 800 is out of bounds for axis 0 with size 800
680 index 800 is out of bounds for axis 0 with size 800
696 index 800 is out of bounds for axis 0 with size 800
708 index 800 is out of bounds for axis 0 with size 800
714 index 800 is out of bounds for axis 0 with size 800
715 index 800 is out of bounds for axis 0 with size 800
733 index 800 is out of bounds for axis 0 with size 800
734 index 800 is out of bounds for axis 0 with size 800
743 index 800 is out of bounds for axis 0 with size 800
753 index 800 is out of bounds for axis 0 with size 800
816 index 800 is out of bounds for axis 0 with size 800
901 index 800 is out of bounds for axis 0 with size 800
911 index 800 is out of bounds for axis 0 with size 800
962 index 800 is out of bounds for axis 0 with size 800
965 index 800 is out of bounds for axis 0 with size 800
1015 index 800 is out of bounds for axis 0 with size 800
1022 index 800 is out of bounds for axis 0 with size 800
1112 index 800 is out of bounds for axis 0 with size 800
1116 index 800 is out of bounds for axis 0 with size 800
1155 index 800 is out of bounds for axis 0 with size 800
1167 index 800 is out of bounds for axis 0 with size 800
1170 index 800 is out of bounds for axis 0 with size 800
1173 index 800 is out of bounds for axis 0 with size 800
1181 index 800 is out of bounds for axis 0 with size 800
1265 index 800 is out of bounds for axis 0 with size 800
1269 index 800 is out of bounds for axis 0 with size 800
1292 index 800 is out of bounds for axis 0 with size 800
1294 index 800 is out of bounds for axis 0 with size 800
1357 index 800 is out of bounds for axis 0 with size 800
1372 index 800 is out of bounds for axis 0 with size 800
```

```
1398 index 800 is out of bounds for axis 0 with size 800
1412 index 800 is out of bounds for axis 0 with size 800
1413 index 800 is out of bounds for axis 0 with size 800
1422 index 800 is out of bounds for axis 0 with size 800
1426 index 800 is out of bounds for axis 0 with size 800
1429 index 800 is out of bounds for axis 0 with size 800
1442 index 800 is out of bounds for axis 0 with size 800
1462 index 800 is out of bounds for axis 0 with size 800
1481 index 800 is out of bounds for axis 0 with size 800
1501 index 800 is out of bounds for axis 0 with size 800
1527 index 800 is out of bounds for axis 0 with size 800
1551 index 800 is out of bounds for axis 0 with size 800
1584 index 800 is out of bounds for axis 0 with size 800
1586 index 800 is out of bounds for axis 0 with size 800
1592 index 800 is out of bounds for axis 0 with size 800
1605 index 800 is out of bounds for axis 0 with size 800
1620 index 800 is out of bounds for axis 0 with size 800
1625 index 800 is out of bounds for axis 0 with size 800
1672 index 800 is out of bounds for axis 0 with size 800
1728 index 800 is out of bounds for axis 0 with size 800
1766 index 800 is out of bounds for axis 0 with size 800
1781 index 800 is out of bounds for axis 0 with size 800
1804 index 800 is out of bounds for axis 0 with size 800
1811 index 800 is out of bounds for axis 0 with size 800
1812 index 800 is out of bounds for axis 0 with size 800
1819 index 800 is out of bounds for axis 0 with size 800
1825 index 800 is out of bounds for axis 0 with size 800
1832 index 800 is out of bounds for axis 0 with size 800
1854 index 800 is out of bounds for axis 0 with size 800
1883 index 800 is out of bounds for axis 0 with size 800
1891 index 800 is out of bounds for axis 0 with size 800
1908 index 800 is out of bounds for axis 0 with size 800
1924 index 800 is out of bounds for axis 0 with size 800
1958 index 800 is out of bounds for axis 0 with size 800
2005 index 800 is out of bounds for axis 0 with size 800
2061 index 800 is out of bounds for axis 0 with size 800
2086 index 800 is out of bounds for axis 0 with size 800
2105 index 800 is out of bounds for axis 0 with size 800
2138 index 800 is out of bounds for axis 0 with size 800
2149 index 800 is out of bounds for axis 0 with size 800
2167 index 800 is out of bounds for axis 0 with size 800
2175 index 800 is out of bounds for axis 0 with size 800
2180 index 800 is out of bounds for axis 0 with size 800
2220 index 800 is out of bounds for axis 0 with size 800
2243 index 800 is out of bounds for axis 0 with size 800
2260 index 800 is out of bounds for axis 0 with size 800
2262 index 800 is out of bounds for axis 0 with size 800
2304 index 800 is out of bounds for axis 0 with size 800
2328 index 800 is out of bounds for axis 0 with size 800
2356 index 800 is out of bounds for axis 0 with size 800
2373 index 800 is out of bounds for axis 0 with size 800
2391 index 800 is out of bounds for axis 0 with size 800
2419 index 800 is out of bounds for axis 0 with size 800
2428 index 800 is out of bounds for axis 0 with size 800
2462 index 800 is out of bounds for axis 0 with size 800
2466 index 800 is out of bounds for axis 0 with size 800
2469 index 800 is out of bounds for axis 0 with size 800
2487 index 800 is out of bounds for axis 0 with size 800
2500 index 800 is out of bounds for axis 0 with size 800
2517 index 800 is out of bounds for axis 0 with size 800
2559 index 800 is out of bounds for axis 0 with size 800
2603 index 800 is out of bounds for axis 0 with size 800
2616 index 800 is out of bounds for axis 0 with size 800
2628 index 800 is out of bounds for axis 0 with size 800
2652 index 800 is out of bounds for axis 0 with size 800
2669 index 800 is out of bounds for axis 0 with size 800
2678 index 800 is out of bounds for axis 0 with size 800
2683 index 800 is out of bounds for axis 0 with size 800
2743 index 800 is out of bounds for axis 0 with size 800
```

```
2754 index 800 is out of bounds for axis 0 with size 800
(2633, 800, 50)
```

Out[9]:
```
(2633, 40000)
```

In [10]:
```
vectors_array_test = []
labels_test= []

for enum, text, label in zip(range(len(newsgroups_test.data)), newsgroups_tes
  try:
    vector, length = prepare(text)
    vectors_array_test.append(vector)
    labels_test.append(label)
  except IndexError as e:
    print(enum, e)
    continue
```

```
67 index 800 is out of bounds for axis 0 with size 800
76 index 800 is out of bounds for axis 0 with size 800
124 index 800 is out of bounds for axis 0 with size 800
137 index 800 is out of bounds for axis 0 with size 800
292 index 800 is out of bounds for axis 0 with size 800
298 index 800 is out of bounds for axis 0 with size 800
350 index 800 is out of bounds for axis 0 with size 800
432 index 800 is out of bounds for axis 0 with size 800
435 index 800 is out of bounds for axis 0 with size 800
476 index 800 is out of bounds for axis 0 with size 800
484 index 800 is out of bounds for axis 0 with size 800
525 index 800 is out of bounds for axis 0 with size 800
558 index 800 is out of bounds for axis 0 with size 800
618 index 800 is out of bounds for axis 0 with size 800
680 index 800 is out of bounds for axis 0 with size 800
710 index 800 is out of bounds for axis 0 with size 800
720 index 800 is out of bounds for axis 0 with size 800
778 index 800 is out of bounds for axis 0 with size 800
780 index 800 is out of bounds for axis 0 with size 800
802 index 800 is out of bounds for axis 0 with size 800
819 index 800 is out of bounds for axis 0 with size 800
825 index 800 is out of bounds for axis 0 with size 800
836 index 800 is out of bounds for axis 0 with size 800
862 index 800 is out of bounds for axis 0 with size 800
882 index 800 is out of bounds for axis 0 with size 800
956 index 800 is out of bounds for axis 0 with size 800
960 index 800 is out of bounds for axis 0 with size 800
989 index 800 is out of bounds for axis 0 with size 800
1064 index 800 is out of bounds for axis 0 with size 800
1101 index 800 is out of bounds for axis 0 with size 800
1108 index 800 is out of bounds for axis 0 with size 800
1152 index 800 is out of bounds for axis 0 with size 800
1187 index 800 is out of bounds for axis 0 with size 800
1193 index 800 is out of bounds for axis 0 with size 800
1293 index 800 is out of bounds for axis 0 with size 800
1313 index 800 is out of bounds for axis 0 with size 800
1386 index 800 is out of bounds for axis 0 with size 800
1443 index 800 is out of bounds for axis 0 with size 800
1455 index 800 is out of bounds for axis 0 with size 800
1463 index 800 is out of bounds for axis 0 with size 800
1477 index 800 is out of bounds for axis 0 with size 800
1482 index 800 is out of bounds for axis 0 with size 800
1529 index 800 is out of bounds for axis 0 with size 800
1552 index 800 is out of bounds for axis 0 with size 800
1560 index 800 is out of bounds for axis 0 with size 800
1561 index 800 is out of bounds for axis 0 with size 800
1629 index 800 is out of bounds for axis 0 with size 800
1631 index 800 is out of bounds for axis 0 with size 800
1639 index 800 is out of bounds for axis 0 with size 800
1664 index 800 is out of bounds for axis 0 with size 800
1709 index 800 is out of bounds for axis 0 with size 800
```

```
1717 index 800 is out of bounds for axis 0 with size 800
1770 index 800 is out of bounds for axis 0 with size 800
1837 index 800 is out of bounds for axis 0 with size 800
```

In [11]:
```
vectors_array_test = np.array(vectors_array_test)
test_data = vectors_array_test.reshape((-1, vectors_array_test.shape[1]*vecto
test_data.shape
```

Out[11]:
```
(1784, 40000)
```

In [17]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV


glove_knn_clf = KNeighborsClassifier()
parameters = {'n_neighbors': [2, 3, 5, 7, 9]}

glove_clf_grid = GridSearchCV(glove_knn_clf, parameters, verbose=4, cv=3,
                              scoring='roc_auc_ovr_weighted', n_jobs=-1)


glove_clf_grid.fit(train_data, labels_train)

print('best param of n_neighbors', glove_clf_grid.best_params_['n_neighbors']
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  15 out of  15 | elapsed: 22.2min finished
best param of n_neighbors 9
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-17-8b00f2d61c86> in <module>()
     20
     21
---> 22 print(classification_report(labels_test[:800], best_pred_glove_knn))

NameError: name 'classification_report' is not defined
```

In [18]:
```
from sklearn.metrics import classification_report

best_glove_knn = KNeighborsClassifier(n_neighbors=glove_clf_grid.best_params_

best_glove_knn.fit(train_data, labels_train)

best_pred_glove_knn = best_glove_knn.predict(test_data[:800])


print(classification_report(labels_test[:800], best_pred_glove_knn))
```

```
              precision    recall  f1-score   support

           0       0.28      0.83      0.42       169
           1       0.49      0.23      0.32       171
           2       0.51      0.19      0.28       180
           3       0.42      0.17      0.24       164
           4       0.46      0.32      0.38       116

    accuracy                           0.35       800
   macro avg       0.43      0.35      0.33       800
weighted avg       0.43      0.35      0.32       800
```

In [ ]: