Предобработка и классификация текстовых данных.

1.Цель лабораторной работы

изучение методов предобработки и классификации текстовых данных.

2.Задание

1.Для произвольного предложения или текста решите следующие задачи:

Токенизация.

Частеречная разметка.

Лемматизация.

Выделение (распознавание) именованных сущностей.

Разбор предложения.

2.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.

Сравните качество полученных моделей.

Для поиска наборов данных в поисковой системе можно использовать ключевые слова "datasets for text classification".

Обработка текста

Токенизация

In [6]:
```
!pip install razdel
```

```
Collecting razdel
  Downloading razdel-0.5.0-py3-none-any.whl (21 kB)
Installing collected packages: razdel
Successfully installed razdel-0.5.0

WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (d:\software\anaconda\lib\site-packages)
```

In [7]:
```
from razdel import tokenize, sentenize
```

In [8]:
```
text = 'обеспечить возможность записи информации о клиенте (номер банковско
```

In [9]:
```
n_tok_text = list(tokenize(text))
n_tok_text
```

Out[9]:
```
[Substring(0, 10, 'о б е с п е ч и т ь'),
 Substring(11, 22, 'в о з м о ж н о с т ь'),
 Substring(23, 29, 'з а п и с и'),
 Substring(30, 40, 'и н ф о р м а ц и и'),
 Substring(41, 42, 'о'),
 Substring(43, 50, 'к л и е н т е'),
 Substring(51, 52, '('),
 Substring(52, 57, 'н о м е р'),
 Substring(58, 68, 'б а н к о в с к о й'),
 Substring(69, 74, 'к а р т ы'),
 Substring(74, 75, ','),
 Substring(76, 79, 'и м я'),
 Substring(79, 80, ','),
 Substring(81, 86, 'н о м е р'),
 Substring(87, 95, 'т е л е ф о н а'),
 Substring(96, 97, 'и'),
 Substring(98, 104, 'д р у г а я'),
 Substring(105, 113, 'о с н о в н а я'),
 Substring(114, 124, 'и н ф о р м а ц и я'),
 Substring(124, 125, ')'),
 Substring(125, 126, ','),
 Substring(127, 133, 'п о и с к а'),
 Substring(133, 134, ','),
 Substring(135, 143, 'к о н т р о л я'),
 Substring(144, 151, 'д о с т у п а'),
 Substring(152, 153, 'и'),
 Substring(154, 165, 'м о н и т о р и н г а'),
 Substring(165, 166, ','),
 Substring(167, 168, 'а'),
 Substring(169, 174, 'т а к ж е'),
 Substring(175, 185, 'у п р а в л е н и я'),
 Substring(186, 194, 'ж у р н а л о м'),
 Substring(195, 206, 'р е г и с т р а ц и и')]
```

In [10]:
```
[_.text for _ in n_tok_text]
```

Out[10]:
```
['о б е с п е ч и т ь',
 'в о з м о ж н о с т ь',
 'з а п и с и',
 'и н ф о р м а ц и и',
 'о',
 'к л и е н т е',
 '(',
 'н о м е р',
 'б а н к о в с к о й',
 'к а р т ы',
 ',',
 ',',
 'и м я',
 ',',
 ',',
 'н о м е р',
 'т е л е ф о н а',
 'и',
 'д р у г а я',
 'о с н о в н а я',
 'и н ф о р м а ц и я',
 ')',
 ',',
 ',',
 'п о и с к а',
 ',',
 ',',
 'к о н т р о л я',
 'д о с т у п а',
 'и',
 'м о н и т о р и н г а',
 ',',
 ',',
 'а',
 'т а к ж е',
 'у п р а в л е н и я',
 'ж у р н а л о м',
 'р е г и с т р а ц и и']
```

In [11]:
```
n_sen_text = list(sentenize(text))
n_sen_text
```

Out[11]:
```
[Substring(0,
           206,
           'обеспечить возможность записи информации о клиенте (номер банковской карты, имя, номер телефона и другая основная информация), поиска, контроля доступа и мониторинга, а также управления журналом регистрации')]
```

In [12]:
```
[_.text for _ in n_sen_text], len([_.text for _ in n_sen_text])
```

Out[12]:
```
(['обеспечить возможность записи информации о клиенте (номер банковской карты, имя, номер телефона и другая основная информация), поиска, контроля доступа и мониторинга, а также управления журналом регистрации'],
 1)
```

In [13]:
```
# Этот вариант токенизации нужен для последующей обработки
def n_sentenize(text):
    n_sen_chunk = []
    for sent in sentenize(text):
        tokens = [_.text for _ in tokenize(sent.text)]
        n_sen_chunk.append(tokens)
    return n_sen_chunk
```

In [14]:
```
n_sen_chunk = n_sentenize(text)
n_sen_chunk
```

Out[14]:
```
[['о б е с п е ч и т ь',
  'в о з м о ж н о с т ь',
  'з а п и с и',
  'и н ф о р м а ц и и',
  'о',
  'к л и е н т е',
  '(',
  'н о м е р',
  'б а н к о в с к о й',
  'к а р т ы',
  ',',
  'и м я',
  ',',
  'н о м е р',
  'т е л е ф о н а',
  'и',
  'д р у г а я',
  'о с н о в н а я',
  'и н ф о р м а ц и я',
  ')',
  ',',
  'п о и с к а',
  ',',
  'к о н т р о л я',
  'д о с т у п а',
  'и',
  'м о н и т о р и н г а',
  ',',
  'а',
  'т а к ж е',
  'у п р а в л е н и я',
  'ж у р н а л о м',
  'р е г и с т р а ц и и']]
```

3.1 Для произвольного набора данных, предназначенного для классификации текстов, решите задачу классификации текста двумя

способами

Способ 1. На основе CountVectorizer

In [49]:
```
import re
import pandas as pd
import numpy as np
from typing import Dict, Tuple
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from nltk import WordPunctTokenizer
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\31139\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[49]: True

In [50]:
```
def accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray) -> Dict[int, float]:
    """
    Вычисление метрики accuracy для каждого класса
    y_true - истинные значения классов
    y_pred - предсказанные значения классов
    Возвращает словарь: ключ - метка класса,
    значение - Accuracy для данного класса
    """
    # Для удобства фильтрации сформируем Pandas DataFrame
    d = {'t': y_true, 'p': y_pred}
    df = pd.DataFrame(data=d)
    # Метки классов
    classes = np.unique(y_true)
    # Результирующий словарь
    res = dict()
    # Перебор меток классов
    for c in classes:
        # отфильтруем данные, которые соответствуют
        # текущей метке класса в истинных значениях
        temp_data_flt = df[df['t']==c]
        # расчет accuracy для заданной метки класса
        temp_acc = accuracy_score(
            temp_data_flt['t'].values,
            temp_data_flt['p'].values)
        # сохранение результата в словарь
        res[c] = temp_acc
    return res

def print_accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray):
    """
    Вывод метрики accuracy для каждого класса
    """
    accs = accuracy_score_for_classes(y_true, y_pred)
    if len(accs)>0:
        print('Метка \t Accuracy')
    for i in accs:
        print('{} \t {}'.format(i, accs[i]))
```

In [53]:
```
df=pd.read_csv("youtube.csv")
df.head()
```

Out[53]:

	link	title	description	category
0	JLZICZ0	Ep 1| Travelling through North East India | Of...	Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nT...	travel
1	i9E_Blai8vk	Welcome to Bali | Travel Vlog | Priscilla Lee	Priscilla Lee\n45.6K subscribers\nSUBSCRIBE\n*...	travel
2	r284c-q8oY	My Solo Trip to ALASKA | Cruising From Vancouv...	Allison Anderson\n588K subscribers\nSUBSCRIBE\...	travel
3	Qmi-Xwq-ME	Traveling to the Happiest Country in the World!!	Yes Theory\n6.65M subscribers\nSUBSCRIBE\n*BLA...	travel
4	_IcOX55Ef70	Solo in Paro Bhutan | Tiger's Nest visit | Bhu...	Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nH...	travel

In [56]:
```
#Только держать колонки "verified_reviews" и "feedback".
df_new = pd.DataFrame(df,columns=['description','category'])
df_new.columns = ['text','value']
df_new.head()
```

Out[56]:

	text	value
0	Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nT...	travel
1	Priscilla Lee\n45.6K subscribers\nSUBSCRIBE\n*...	travel
2	Allison Anderson\n588K subscribers\nSUBSCRIBE\...	travel
3	Yes Theory\n6.65M subscribers\nSUBSCRIBE\n*BLA...	travel
4	Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nH...	travel

In [57]:
```
df_new.shape
```

Out[57]: (3599, 2)

In [58]:
```
#Сформируем общий словарь
vocab_list = df_new['text'].tolist()
vocab_list[1:10]
```

Out[58]:
```
["Priscilla Lee\n45.6K subscribers\nSUBSCRIBE\n*DISCLAIMER* Please do not ride elephants when visiting any country. At the time I d
idn't know (yes, I was dumb) so it is shown in the video, but I do not support the elephant riding business anymore. If I could tak
e it back I would, but instead I want to pass on the knowledge to anyone who isn't aware. Here's some info: \nSHOW MORE",
 'Allison Anderson\n588K subscribers\nSUBSCRIBE\nI spent 11 days cruising up the coast of Alaska and it was MAGICAL.\n•ALASKA BLOG
POST: https://allisonanderson.com/blog/crui...\n•Adventures (https://allisonanderson.com/blog/crui...\n•Adventures) on INSTAGRAM ht
tp://www.instagram.com/photoallison\nSHOW (http://www.instagram.com/photoallison\nSHOW) MORE',
 "Yes Theory\n6.65M subscribers\nSUBSCRIBE\n*BLACK FRIDAY DROP Out Now*: http://seek-discomfort.com/yes-theory (http://seek-discomf
ort.com/yes-theory) \nThis week only, with every purchase about $35, you'll get 2 free Seek Discomfort flags!\n\nCheck out our frie
nds from Beautiful Destinations!! Their videos are INCREDIBLE:\nSHOW MORE",
 'Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nHere's presenting the first part of the Bhutan Series Episode in Paro. I went strai
ght to Paro as first part of my road trip in the country. The drive from Phuntsholing took about 4 hours. \n \nThe entire budget of
my Bhutan trip was close to  INR 25k. You can carry cash everywhere in Indian currency in Bhutan as it is accepted. Some things abo
ut Paro below:\n\n1. The place where I stayed at in Paro is called Ama's Village Lodge. You can book the place here - \nSHOW MOR
E',
 'MOUNTAIN TREKKER\n1.42M subscribers\nJOIN\nSUBSCRIBE\nTHAILAND: FREE VISA ON ARRIVAL  \nCheck this- https://www.youtube.com/chann
el/UC15d...\nSHOW (https://www.youtube.com/channel/UC15d...\nSHOW) MORE',
 "XTREME MOTO ADVENTURE\n800K subscribers\nJOIN\nSUBSCRIBE\nAmazing Kerala story | Rainforest Athirapally | \nHere's is episode 37
 of my All India ride and you will be amazed by this episode. Kerala is really a god's own country..Check it out yourself.\nGet 15
 % Discount on Booking\nCoupon Code : EXPLORE \nRainforest link : \nSHOW MORE",
 'visa2explore\n1.21M subscribers\nJOIN\nSUBSCRIBE\nHow to plan your journey of Meghalaya, North east India Tour. This video has in
formation on Things to do in Meghalaya.\n\nNorth East India is known for its beauty, hills, waterfalls and so much more. After watc
hing this video you will know about North east India tourism ie tourist places.\n\nThis video can help you plan out your North East
India tour, starting from Meghalaya, then you can watch our series of Assam and Sikkim too.\n\nThis video on Meghalaya tourism has
 loads of knowledge on must visit Tourist destinations of Meghalaya. we spent 15 days in Meghalaya.\n\nWe traveled to all the 3 tri
be areas of Meghalaya - khasi hills, Jaintia hills and Garo hills. Basis my experience i have shared with you a plan for 4 nights a
nd 5 days in Meghalaya.\n\nImportant information below:\n\nPlaces to visit in Sohra ( also called Cherrapunji), basis time availabi
lity at your end your may prioritise, which places to visit and which ones to ignore.\n\nWah Kaba Falls\nDainthlen Falls\nWei Sawdo
ng Falls\nNohkalikai Falls\nThe Seven Sisters Falls\nKynrem Falls\nMawsmai Cave\nArwah Cave\nDouble decker Living Root Bridge\nRain
bow Falls\n\nPlaces to visit in Dawki\n\nUmngot River (You can enjoy boating here)\nShnongpdeng (You can enjoy water sports at this
place)\n\nLamin Guest House: we stayed here, there are not many hotels in Dawki\nRoom Tariff Rs 3000 for double occupancy (Room als
o starts from Rs 1500)\nWebsite: \nSHOW MORE',
 "Garima's Good Life\n1.76M subscribers\nSUBSCRIBE\nDo watch my video on how to prepare and pack for LADAKH trip if you plan to vis
it \nhttps://youtu.be/TRUTQ7fd_XA\nSHOW MORE",
 "Tanya Khanijow\n671K subscribers\nSUBSCRIBE\nKerala is one of the most beautiful states in India for traveling and a road trip. I
n this series  we are going to cover 5 main destinations by road - Wayanad, Athirapally falls, Vagamon, Varkala, and Alleppey. In t
his episode, I'm in Varkala, with some of the best beaches! It has places like black sand beach, and North Cliff, restaurants like
 Inda Cafe and Kerala God's Own Country Kitchen. \n\nI actually traveled to Kerala from Karnataka and took an RT-PCR negative repor
```

```
t with me while travelling and in this episode, I stayed at Cliff Stories, which is a really nice place for relaxing and staying a
t. 😀\n\nThis video is sponsored by Skillshare!\nGive a new direction to your creative skills, first 5000 subscribers can avail a
 free trial on Skillshare using this link - \nSHOW MORE"]
```

In [59]:
```
vocabVect = CountVectorizer(
    stop_words='english',
    ngram_range=(1, 1),  #ngram_range=(1, 1) is the default
    dtype='double'
)
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusVocab)))
```

```
Количество сформированных признаков - 22038
```

In [60]:
```
for i in list(corpusVocab)[1:10]:
    print('{}={}'.format(i, corpusVocab[i]))
```

```
khanijow=10716
671k=1257
subscribers=18036
subscribe=18030
journey=10329
arunachal=2552
north=13431
east=6607
india=9566
```

In [61]:
```
test_features = vocabVect.transform(vocab_list)
```

In [62]:
```
test_features
```

Out[62]:
```
<3599x22038 sparse matrix of type '<class 'numpy.float64'>'
        with 114891 stored elements in Compressed Sparse Row format>
```

In [63]:
```
test_features.todense()
```

Out[63]:
```
matrix([[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.]])
```

In [64]:
```
# Размер нулевой строки
len(test_features.todense()[0].getA1())
```

Out[64]:
```
22038
```

In [65]:
```
# Непустые значения нулевой строки
[i for i in test_features.todense()[0].getA1() if i>0]
```

Out[65]:
```
[1.0,
 1.0,
 2.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 2.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0,
 1.0]
```

In [66]:
```
vocabVect.get_feature_names()[100:120]
```

```
D:\software\anaconda\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: Function get_feature_names is deprecated; ge
t_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[66]:
```
['11204',
 '11261',
 '112m',
 '113',
 '113k',
 '114',
 '114k',
 '115',
 '115k',
 '116',
 '1160',
 '116k',
 '117',
 '117k',
 '118',
 '1185',
 '119',
 '11k',
 '11m',
 '11th']
```

Разделим выборку на обучающую и тестовую

In [68]:
```
X_train, X_test, y_train, y_test = train_test_split(df_new['text'], df_new['value'], test_size=0.3, random_state=1)
```

In [69]:
```
def sentiment(v, c):
    model = Pipeline(
        [("vectorizer", v),
         ("classifier", c)])
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print_accuracy_score_for_classes(y_test, y_pred)
```

Используем классификатор "LogisticRegression"

In [70]:
```
sentiment(CountVectorizer(), LogisticRegression(C=3.0))
```

```
Метка       Accuracy
art_music       0.9253731343283582
food    0.8598484848484849
history         0.8531073446327684
travel  0.9191374663072777

D:\software\anaconda\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=
1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_m
odel.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Способ 2. На основе моделей word2vec

In [71]:
```
import gensim
from gensim.models import word2vec
```

In [73]:
```
# Подготовим корпус
corpus = []
stop_words = stopwords.words('english')
tok = WordPunctTokenizer()
for line in df_new['text'].values:
    line1 = line.strip().lower()
    line1 = re.sub("[^a-zA-Z]"," ", line1)
    text_tok = tok.tokenize(line1)
    text_tok1 = [w for w in text_tok if not w in stop_words]
    corpus.append(text_tok1)
```

In [74]:
```
corpus[:5]
```

Out[74]:
```
[['tanya',
  'khanijow',
  'k',
  'subscribers',
  'subscribe',
  'journey',
  'arunachal',
  'north',
  'east',
  'india',
  'begins',
  'train',
  'journey',
  'guwahati',
  'murkongselek',
  'head',
  'pasighat',
  'travel',
  'companions',
```

In [36]:
```
corpus[:5]
```

Out[36]:
```
[['tanya',
  'khanijow',
  'k',
  'subscribers',
  'subscribe',
  'journey',
  'arunachal',
  'north',
  'east',
  'india',
  'begins',
  'train',
  'journey',
  'guwahati',
  'murkongselek',
  'head',
  'pasighat',
  'travel',
  'companions',
```

In [76]:
```
assert df_new.shape[0]==len(corpus)
```

In [77]:
```
%time model_imdb = word2vec.Word2Vec(corpus, workers=4, min_count=10, window=10, sample=1e-3)
```

```
Wall time: 304 ms
```

In [79]:
```
# Проверим, что модель обучилась
print(model_imdb.wv.most_similar(positive=['adventure'], topn=10))
```

```
[('huge', 0.9954676628112793), ('rest', 0.9946334958076477), ('big', 0.9945908784866333), ('order', 0.9945046901702881), ('peace',
0.9945025444030762), ('quick', 0.9943442344665527), ('headed', 0.9942252039909363), ('eleven', 0.9940177202224731), ('dream', 0.993
9081072807312), ('chen', 0.9938148856163025)]
```

In [80]:
```
class EmbeddingVectorizer(object):
    '''
    Для текста усредним вектора входящих в него слов
    '''
    def __init__(self, model):
        self.model = model
        self.size = model.vector_size

    def fit(self, X, y):
        return self

    def transform(self, X):
        return np.array([np.mean(
            [self.model[w] for w in words if w in self.model]
            or [np.zeros(self.size)], axis=0)
            for words in X])
```

In [82]:
```
sentiment(EmbeddingVectorizer(model_imdb.wv), LogisticRegression(C=3.0))
```

```
D:\software\anaconda\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=
1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_m
odel.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Метка	Accuracy
art_music	0.4925373134328358
food	0.2803030303030303
history	0.13559322033898305
travel	0.5876010781671159

In [83]:
```
sentiment(EmbeddingVectorizer(model_imdb.wv), LogisticRegression(C=5.0))
```

```
D:\software\anaconda\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=
1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_m
odel.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Метка	Accuracy
art_music	0.48880597014925375
food	0.3712121212121212
history	0.1638418079096045
travel	0.522911051212938

In [85]:
```
from sklearn.neighbors import KNeighborsClassifier
sentiment(EmbeddingVectorizer(model_imdb.wv), KNeighborsClassifier(n_neighbors=5))
```

Метка	Accuracy
art_music	0.5522388059701493
food	0.4090909090909091
history	0.3446327683615819
travel	0.40431266846361186

In [43]:
```
def accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray) -> Dict[int, float]:
    """
    Вычисление метрики accuracy для каждого класса
    y_true - истинные значения классов
    y_pred - предсказанные значения классов
    Возвращает словарь: ключ - метка класса,
    значение - Accuracy для данного класса
    """
    # Для удобства фильтрации сформируем Pandas DataFrame
    d = {'t': y_true, 'p': y_pred}
    df = pd.DataFrame(data=d)
    # Метки классов
    classes = np.unique(y_true)
    # Результирующий словарь
    res = dict()
    # Перебор меток классов
    for c in classes:
        # отфильтруем данные, которые соответствуют
        # текущей метке класса в истинных значениях
        temp_data_flt = df[df['t']==c]
        # расчет accuracy для заданной метки класса
        temp_acc = accuracy_score(
            temp_data_flt['t'].values,
            temp_data_flt['p'].values)
        # сохранение результата в словарь
        res[c] = temp_acc
    return res

def print_accuracy_score_for_classes(
    y_true: np.ndarray,
    y_pred: np.ndarray):
    """
    Вывод метрики accuracy для каждого класса
    """
    accs = accuracy_score_for_classes(y_true, y_pred)
    if len(accs)>0:
        print('Метка \t Accuracy')
    for i in accs:
        print('{} \t {}'.format(i, accs[i]))
```

In [48]:
```
sentiment(EmbeddingVectorizer(model_imdb.wv), LogisticRegression(C=5.0))
```

Метка	Accuracy
art_music	0.6666666666666666
history	0.0

```
D:\software\anaconda\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=
1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_m
odel.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

In [87]:
```
df_new[df_new['value']=='history']
```

Out[87]:

	text	value
3006	Ranveer Allahbadia\n2.53M subscribers\nSUBSCRI...	history
3007	Knowledgia\n650K subscribers\nSUBSCRIBE\nTHE H...	history
3008	Let's Crack UPSC CSE\n4.71M subscribers\nSUBSC...	history
3009	Abhijit Chavda\n74.8K subscribers\nSUBSCRIBE\n...	history
3010	PDF visuals\n98.8K subscribers\nSUBSCRIBE\nHer...	history
...	...	...
3594	CrashCourse\n12.4M subscribers\nSUBSCRIBE\nThe...	history
3595	Publications Office of the European Union\n3.2...	history
3596	History Time\n619K subscribers\nSUBSCRIBE\n- W...	history
3597	Mr. Raymond's Civics and Social Studies Academ...	history
3598	Paul Sargent\n25.3K subscribers\nSUBSCRIBE\nIn...	history

593 rows × 2 columns