Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №6
по дисциплине
«Методы машинного обучения»

Выполнил:
студент группы ИУ5-22М
ЧжаоЛян

Москва — 2022 г.

In [6]:
```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score, classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_squared_log_error, median_absolute_error, r2_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
import seaborn as sns
from collections import Counter
from sklearn.datasets import fetch_20newsgroups
import matplotlib.pyplot as plt

%matplotlib inline
sns.set(style="ticks")
```

In [7]:
```
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]))
```

Загрузка данных:

In [8]:
```
%cd /content/drive/MyDrive/dataset/spam/
```

```
/content/drive/MyDrive/dataset/spam
```

In [9]:
```
dataset = pd.read_csv("enron_spam_data.csv")
dataset.head()
```

Out[9]:

	Unnamed: 0	Subject	Message	Spam/Ham	Date
0	0	christmas tree farm pictures	NaN	ham	1999-12-10
1	1	vastar resources , inc .	gary , production from the high island larger ...	ham	1999-12-13
2	2	calpine daily gas nomination	- calpine daily gas nomination 1 . doc	ham	1999-12-14
3	3	re : issue	fyi - see note below - already done .\nstella\...	ham	1999-12-14
4	4	meter 7268 nov allocation	fyi .\n- - - - - - - - - - - - - - - - - - -...	ham	1999-12-14

In [10]:
```
dataset=dataset.drop(['Unnamed: 0','Subject','Date'],axis=1)
dataset.head()
```

Out[10]:

	Message	Spam/Ham
0	NaN	ham
1	gary , production from the high island larger ...	ham
2	- calpine daily gas nomination 1 . doc	ham
3	fyi - see note below - already done .\nstella\...	ham
4	fyi .\n- - - - - - - - - - - - - - - - - - -...	ham

In [11]:
```
dataset['Spam/Ham']=dataset['Spam/Ham'].replace(['ham','spam'],[0,1])
dataset.head()
```

Out[11]:

	Message	Spam/Ham
0	NaN	0
1	gary , production from the high island larger ...	0
2	- calpine daily gas nomination 1 . doc	0
3	fyi - see note below - already done .\nstella\...	0
4	fyi .\n- - - - - - - - - - - - - - - - - - -...	0

In [12]:
```
dataset=dataset.dropna()
dataset.head()
```

Out[12]:

	Message	Spam/Ham
1	gary , production from the high island larger ...	0
2	- calpine daily gas nomination 1 . doc	0
3	fyi - see note below - already done .\nstella\...	0
4	fyi .\n- - - - - - - - - - - - - - - - - - -...	0
5	jackie ,\nsince the inlet to 3 river plant is ...	0

In [17]:
```
dataset=dataset.sample(frac=1)
dataset.head()
```

Out[17]:

	Message	Spam/Ham
7683	gentleman ,\nkevin presto concurred on the pur...	0
434	daren or stacey : could you please extend deal...	0
19561	start date : 2 / 6 / 02 ; hourahead hour : 24 ...	1
31175	fyi , kim .\n- - - - original message - - - ...	1
27668	attached is the latest version of the cost cen...	1

In [18]:
```
dataset.describe()
```

Out[18]:

	Spam/Ham
count	33664.000000
mean	0.510070
std	0.499906
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

In [20]:
```
train_df=(dataset.iloc[0:26664,:])
test_df=(dataset.iloc[26664:33664,:])
```

In [23]:
```
train_df.describe()
```

Out[23]:

	Spam/Ham
count	26664.000000
mean	0.509338
std	0.499922
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

In [22]:
```
test_df.describe()
```

Out[22]:

	Spam/Ham
count	7000.000000
mean	0.512857
std	0.499870
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

На основе CountVectorizer или TfidfVectorizer

Сформируем общий словарь для обучения моделей из обучающей и тестовой выборки

In [24]:
```
vocab_list = train_df['Message'].tolist()
vocab_list[1:10]
```

Out[24]:
```
['daren or stacey : could you please extend deal # 169625 for meter 1520 for\nfeb . 2000 ?\nthanks .\naimee\n- - - - - - - - - - - - - -
- - - - - - - - forwarded by aimee lannou / hou / ect on 03 / 11 / 2000 09 : 46\nam - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
\nanita luong\n03 / 10 / 2000 05 : 23 pm\nto : aimee lannou / hou / ect @ ect\ncc :\nsubject : re : allocation exceptions\nmeter 5191 tra
ck id is 89328 .\nmeter 1520 need to extend or deal or add new deal .\naimee lannou 03 / 10 / 2000 02 : 57 pm\nto : anita luong / hou / e
ct @ ect\ncc :\nsubject : allocation exceptions\nanita - i need some accounting arrangements created for two meters .\nmeter contract rec
eipt / delivery counterparty deal # last month allocated\n5191 gathering receipt tri - union development 100896\ncorp . 138661 jan 00\n15
20 hpl 215 delivery engage 169625 jan 00\nif youneed more information , please let e know .\n- aimee',
 "start date : 2 / 6 / 02 ; hourahead hour : 24 ; no ancillary schedules awarded . no variances detected .\nlog messages :\nparsing file
- - > > o : \\ portland \\ westdesk \\ california scheduling \\ iso final schedules \\ 2002020624 . txt\n! ! ! general sql error .\ncould
n ' t update ; currently locked by user ' admin ' on machine ' nahou - trdts 5 ' .\ntable\n- - - - energy import / export schedule - - -
-\n* * * final schedule not found for preferred schedule .\ndetails :\ntrans _ type : final\nsc _ id : ectstnw\nmkt _ type : 2\ntrans _ d
ate : 2 / 6 / 02\ntie _ point : malin _ 5 _ rndmtn\ninterchg _ id : enrj _ ciso _ 3000\nengy _ type : firm",
 'fyi , kim .\n- - - - - original message - - - - -\nfrom : frazier , perry\nsent : thursday , march 07 , 2002 2 : 25 pm\nto : lebeau , r
andy ; watson , kimberly ; abdmoulaie , mansoor\nsubject : re : re ivanhoe e . s . d\njust a couple of additional thoughts , the cost est
imate for esd mods of $ 130 , 000 will typically be , about 25 % more . the e abdmoulaie , mansoor ; frazier , perry\nsubject : fw : re i
vanhoe e . s . d\nhere is an estimate for the upgrade to the ivanhoe esd system . please keep in mind that the hp at the location has not
been in operation for several years and if it is ever required , we may be looking a major expense for that , too . ( $ 200 , 000 + or -
)\n- - - - - original message - - - - -\nfrom : jordan , fred\nsent : thursday , march 07 , 2002 7 : 24 am\nto : lebeau , randy\nsubject
: re ivanhoe e . s . d\nfyi\n- - - - - - - - - - - - - - - - - - - - - - forwarded by fred jordan / et & s / enron on 03 / 07 / 2002 07 :
```

In [25]:
```
def VectorizeAndClassify(vectorizers_list, classifiers_list):
    for v in vectorizers_list:
        for c in classifiers_list:
            pipeline1 = Pipeline([("vectorizer", v), ("classifier", c)])
            score = cross_val_score(pipeline1, train_df['Message'], train_df['Spam/Ham'], scoring='accuracy', cv=3).mean()
            print('Векторизация - {}'.format(v))
            print('Модель для классификации - {}'.format(c))
            print('Accuracy = {}'.format(score))
            print('===========================')
```

In [26]:
```
vocabVect = CountVectorizer()
vocabVect.fit(vocab_list)
corpusVocab = vocabVect.vocabulary_
print('Количество сформированных признаков - {}'.format(len(corpusVocab)))
```

```
Количество сформированных признаков - 54038
```

In [27]:
```
vectorizers_list = [CountVectorizer(vocabulary = corpusVocab), TfidfVectorizer(vocabulary = corpusVocab)]
classifiers_list = [LogisticRegression(), MultinomialNB()]
VectorizeAndClassify(vectorizers_list, classifiers_list)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: 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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: 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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

Лучшую точность показал CountVectorizer и LogisticRegression (99,93%)

In [31]:
```
X_train=train_df['Message']
y_train=train_df['Spam/Ham']
X_test=test_df['Message']
y_test=test_df['Spam/Ham']
```

In [32]:
```
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)
```

In [33]:
```
sentiment(CountVectorizer(), LogisticRegression(C=5.0))
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: 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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Метка	Accuracy
0	0.9994134897360704
1	1.0

На основе моделей word2vec

In [34]:
```
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.linear_model import LogisticRegression
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 /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[34]:
```
True
```

Подготовим корпус

In [53]:
```
corpus = []
stop_words = stopwords.words('english')
tok = WordPunctTokenizer()
for line in dataset['Message'].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 [54]:
```
corpus[:5]
```

Out[54]:
```
[['gentleman',
  'kevin',
  'presto',
  'concurred',
  'purchase',
  'site',
  'license',
  'recommended',
  'vince',
  'thoughts',
  'others',
  'available',
  'demo',
  'package',
  'others',
  'would',
  'like',
  'see',
  'thanks',
```

Количество текстов в корпусе не изменилось и соответствует целевому признаку

In [56]:
```
assert dataset.shape[0]==len(corpus)
```

In [57]:
```
import gensim
from gensim.models import word2vec
%time model = word2vec.Word2Vec(corpus, workers=4, min_count=10, window=10, sample=1e-3)
```

```
CPU times: user 1min 18s, sys: 442 ms, total: 1min 18s
Wall time: 42.3 s
```

Проверим, что модель обучилась

In [58]:
```
print(model.wv.most_similar(positive=['find'], topn=5))
```

```
[('contacts', 0.4571227431297302), ('complete', 0.4518755376338959), ('internally', 0.44660910964012146), ('see', 0.44049549102783203), ('samples', 0.4352661371231079)]
```

In [64]:
```
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)
```

In [65]:
```
class EmbeddingVectorizer(object):
    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 [67]:
```
boundary = 26664
X_train = corpus[:boundary]
X_test = corpus[boundary:]
y_train = dataset['Spam/Ham'][:boundary]
y_test = dataset['Spam/Ham'][boundary:]
```

In [68]:
```
sentiment(EmbeddingVectorizer(model.wv), LogisticRegression(C=5.0))
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: 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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Метка	Accuracy
0	0.9982404692082112
1	1.0

Лучшую точность показал CountVectorizer и LogisticRegression

Список литературы

[1] Гапанюк Ю. Е. Лабораторная работа «Подготовка обучающей и тестовой выборки,
кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей»
[Электронный ресурс] // GitHub. — 2019. — Режим доступа:
https://github.com/
ugapanyuk/ml_course/wiki/LAB_KNN (дата обращения: 05.04.2019).
[2] Team The IPython Development. IPython 7.3.0 Documentation [Electronic resource] //
Read the Docs. — 2019. — Access mode: https://ipython.readthedocs.io/en/
stable/ (online; accessed: 20.02.2019).

[3] Waskom M. seaborn 0.9.0 documentation [Electronic resource] // PyData. — 2018. —

Access mode: https://seaborn.pydata.org/ (online; accessed: 20.02.2019).
[4] pandas 0.24.1 documentation [Electronic resource] // PyData. — 2019. — Access mode:
http://pandas.pydata.org/pandas-docs/stable/ (online; accessed: 20.02.2019).
[5] dronio. Solar Radiation Prediction [Electronic resource] // Kaggle. — 2017. — Access
mode: https://www.kaggle.com/dronio/SolarEnergy (online; accessed: 18.02.2019).
[6] Chrétien M. Convert datetime.time to seconds [Electronic resource] // Stack Overflow.
— 2017. — Access mode: https://stackoverflow.com/a/44823381 (online; accessed:
20.02.2019).
[7] scikit-learn 0.20.3 documentation [Electronic resource]. — 2019. — Access mode: https:
//scikit-learn.org/ (online; accessed: 05.04.2019).